Spatial Augmented Reality

SIGGRAPH 2006 Course 19 Notes

Lecturers

Oliver Bimber
Bauhaus University
Bauhausstr. 11
99423 Weimar, Germany
Email: oliver.bimber@medien.uni-weimar.de
http://www.uni-weimar.de/~bimber

Ramesh Raskar

Senior Research Scientist
MERL - Mitsubishi Electric Research Labs
201 Broadway
Cambridge, MA 02139
Email: raskar@merl.com
http://www.merl.com/people/raskar/

Course Web Page

http://www.SpatialAR.com

Contents

- **Cover**
  - **Course Description**
  - **Prerequisites**
  - **Course Schedule**
  - **Lecturer Biographies**
  - **Course Syllabus**

- **Course Notes**
- **Course Slides**

- **Papers**
- **Code**
- **Bibliography**

Course Description

This course provides a survey of modern techniques for augmented reality (AR) systems. Novel approaches have taken AR beyond traditional eye-worn or hand-held displays, enabling new application areas. We discuss conventional AR displays and present alternative augmented reality approaches that are overcoming some of their limitations. These practical alternatives are spatially augmented displays that exploit video-projectors, optical elements, holograms, radio frequency tags and tracking technologies. Due to the fall in the cost of these devices and graphics resources, there has been a considerable interest in exploiting such augmented reality systems in universities, labs, museums and in the art community.

We present state-of-the-art concepts, details about hardware and software implementations, and current areas of application. We plan to draw parallels between displays techniques used for virtual reality and augmented reality and stimulate thinking about the alternative approaches for AR. The goal of AR is to create high level of consistency between real and virtual environments. We describe techniques for optical combination of virtual and real using mirror-beam splitters, transparent screens and holograms. We present projector-based augmentation of geometrically complex and textured display surfaces, which along with optical combiners achieve consistent illumination and occlusion effects.

We will present many spatial display examples, such as Shader Lamps, Virtual Showcases, Extended Virtual Tables, Interactive Holograms, Apparent Motion, Augmented Paintings and art installations that have been developed, applied and evaluated by the course presenters throughout the years. Finally, we will share our experiences made with such technologies within art, museums, edutainment, research and industrial areas. By handing out construction drawings, algorithms, and source code we aim to enable participants to realize such systems on their own.

Prerequisites

This course is appropriate for beginners in digital art and media. No programming or specific mathematical background is required. General knowledge of basic computer graphics techniques, 3D tools and optics is helpful but not necessary.

Designers of AR systems along with programmers and artists working on interactive applications will learn about modern hardware and software techniques to combine the real and virtual world with several case studies.

Course Schedule

A	Introduction and Review of Current AR Approaches	(Bimber, 15 minutes)
B	Concepts in Projector-based Augmentation	(Raskar, 35 minutes)
C	Augmentation on Everyday Surfaces	(Bimber, 35 minutes)
D	Prototypes and Experiences	(Bimber and Raskar, 15 minutes)
	Q&A	(5 minutes)

Course Web Page http://www.SpatialAR.com

Lecturer Biography

Oliver Bimber

Oliver Bimber is currently a Junior Professor for Augmented Reality at the Bauhaus University Weimar, Germany. He serves on the editorial board of the IEEE Computer magazine (graphics and multimedia editor). Bimber invented several AR displays, such as the Virtual Showcase, Interactive Holograms and Augmented Paintings. He published in several journals, such as IEEE Computer, IEEE CG&A, Presence, Computers and Graphics, and at many graphics, AR and VR conferences. Bimber is member of IEEE, Eurographics, ACM and ACM Siggraph. He has exhibited in the Emerging Technology Laboratory of Siggraph'02 and was a speaker in the Siggraph'02 course "Building Interactive Workspaces". Together with Ramesh Raskar, Bimber organized the proposed course at Eurographics 2003 and 2004.

http://www.uni-weimar.de/~bimber

Ramesh Raskar

Ramesh Raskar is a Senior Research Scientist at MERL-Cambridge Research. His research interests include projector-based graphics, projective geometry and non-photorealistic rendering. During his doctoral research at U. of North Carolina at Chapel Hill, he developed a framework for projector based 3D graphics, which can simplify the constraints on conventional immersive displays, and enable new projector-assisted applications. He has published several articles on immersive projector-based displays, spatially augmented reality and has introduced Shader Lamps, a new approach for projector-based augmentation. His technical papers have appeared in SIGGRAPH, EuroGraphics, IEEE VR, IEEE Visualization, CVPR and many other graphics and vision conferences. He was a course organizer and speaker for Siggraph 2002, 2003 and 2004. He is a member of the ACM and IEEE.

http://www.merl.com/people/raskar/

Course Syllabus

A. Introduction to Current Approaches (Bimber, 15 minutes)

- Overview: VR displays vs. AR displays and new opportunities
- Current Augmented Reality Displays
  - Head-mounted, eye-worn, hand-held displays
  - Spatial projection, hand-held and head-mounted projectors
- Fundamental AR problems: authoring, recognition and registration
- Pros & Cons of Head-mounted Augmentation vs. Spatial Augmentation
- Future Display Technologies

B. Concepts in Projector Based Augmentation (Raskar, 35 minutes)

- Spatially Augmented Reality
- Projector-based Augmentation
  - Planar, non-planar and quadric curved projection in art installations and industry
- Surface adaptive projection and Shader Lamps
  - Illuminating objects and complex shapes: image warping and registration
  - Virtual reflectance by changing surface appearance of real objects
  - Virtual illumination and virtual motion effects, Applications
- Optical and Radio Frequency Tags
  - Interaction with tracked PDAs, projectors and cameras
  - Tag based recognition and image registration

C. Augmentation on Everyday Surfaces (Bimber, 35 minutes)

Mirrors, special screens, holograms and sometime even smoke is used in art installations and industrial displays for augmentation. This module given an overview of the possibilities, explains mathematics and gives pointers on rendering techniques.

- Principles of Optical Combination
- Creating Graphical Overlays with Mirrors and Beam-Splitters
  - Planar and curved mirrors
- Multi-User vs. Single-User Displays

D. Prototypes and Experiences (Bimber, Raskar, 15 minutes)

This module discusses case studies and covers existing modern AR projects. These installations and projects have appeared in museums, art shows, they are being used in the industry and all of them have received media coverage.

- The Virtual Showcase: Mixing mirrors with head tracking and 3D graphics
- Interactive Holograms: Holograms, LCD panels and projectors
- Augmented Paintings: Painting and special projection screens
- AR-CAVE: Cave without Cave
- Shader Lamps: Physical models and projectors
- ILamps: Handheld projectors with image warping
- RFID for Augmented Reality: Location sensing RFID and automatic authoring

Questions and Answers (All, 5 minutes)

Copyright Information

The following copyright information appears on the first page of each corresponding paper.

© 2006 IEEE. Reprinted, with permission, from (…).

© 2006 ACM Reprinted, with permission, from (…).

All trademarks are the property of their respective owners.
Papers and Photos are copyrighted and shall not be reproduced without permission.

Modern Approaches to Augmented Reality

Oliver Bimber[1] and Ramesh Raskar[2]

[1]Bauhaus University, Weimar, Germany
T +49-3643-583724, F +49-3643-583709, Email: bimber@ieee.org, URL: http://www.uni-weimar.de/medien/AR

[2] MERL - Mitsubishi Electric Research Lab, Cambridge, USA
T +1-617-621-7533, F +1-617-621-7550, Email: raskar@merl.com, URL : http://www.merl.com/people/raskar/ , http://www.raskar.com

Abstract

This tutorial discusses the Spatial Augmented Reality (SAR) concept, its advantages and limitations. It will present examples of state-of-the-art display configurations, appropriate real-time rendering techniques, details about hardware and software implementations, and current areas of application. Specifically, it will describe techniques for optical combination using single/multiple spatially aligned mirror-beam splitters, image sources, transparent screens and optical holograms. Furthermore, it presents techniques for projector-based augmentation of geometrically complex and textured display surfaces, and (along with optical combination) methods for achieving consistent illumination and occlusion effects. Emerging technologies that have the potential of enhancing future augmented reality displays will be surveyed.

Categories and Subject Descriptors (according to ACM CSS): H.5.1 [Multimedia Information Systems]: Artificial, Augmented and Virtual Realities; I.3.1 [Hardware Architecture]: Three-dimensional Displays; I.3.2 [Computer Graphics]: Graphics Systems; I.3.3 [Computer Graphics]: Picture/Image Generation – Display Algorithms, Viewing Algorithms; I.3.7 [Computer Graphics]: Color, Shading, Shadowing, and Texture

1. Introduction and overview

Video see-through and optical see-through head-mounted displays have been the traditional output technologies for augmented reality (AR) applications for more than forty years. However, they still suffer from several technological and ergonomic drawbacks which prevent them from being used effectively in all application areas.

Novel approaches have taken augmented reality beyond traditional eye-worn or hand-held displays - enabling additional application areas. New display paradigms exploit large spatially aligned optical elements, such as mirror beam-splitters, transparent screens or holograms, as well as video-projectors. Thus, we call this technological variation "Spatial Augmented Reality (SAR)". In many situations, SAR displays are able to overcome technological and ergonomic limitations of conventional AR systems. Due to the fall in cost and availability of projection technology, personal computers and graphics hardware, there has been a considerable interest in exploiting SAR systems in universities, research laboratories, museums, industry and in the art community. Parallels to the development of virtual environments from head-attached displays to spatial projection screens can be clearly drawn. We believe that an analog evolution of augmented reality has the potential to yield a similar successful factor in many application domains. Thereby, SAR and body-attached AR are not competitive, but complementary.

Chapter 2 gives an overview over different augmented reality display techniques, from head-attached, over hand-held to spatial approaches. It will enable readers to identify parallels between virtual reality and augmented reality display technology, and stimulate them to think about alternative display approaches for AR.

Chapter 3 explains interactive rendering techniques that use fixed function and programmable rendering pipelines to support spatial optical see-through displays. They aim at neutralizing optical effects, such as reflection and refraction on planar or curved spatial optical combiners, such as transparent screen or mirror beam-splitter configurations.

Chapter 4 focuses on rendering methods for projector-based augmentation and illumination. It will be explained how a correct projection onto geometrically complex and textured screen surfaces is performed. Furthermore, it discusses projector-based illumination, and outlines

examples of how it can be used together with optical combination to create consistent illumination and occlusion effects.

In chapter 5 we summarize our tutorial and give an outlook to enabling technologies that might influence augmented reality technology in the future. The possibilities and limitations of technologies, such as autostereoscopy, video projectors, organic light emitting diodes, light emitting polymers, electronic paper, particular solid state volumetric and parallax display approaches, and holography will be outlined.

2. Augmented Reality Displays

Displays are image-forming systems that apply a set of optical, electronic and mechanical components to generate images somewhere on the *optical path* in-between the observer's eyes and the physical object to be augmented. Depending on the optics being used, the image can be formed on a plane or on a more complex non-planar surface.

Figure 2.1 illustrates the different possibilities of where the image can be formed, where the displays are located with respect to the observer and the real object, and what type of image is produced (i.e., planar or curved).

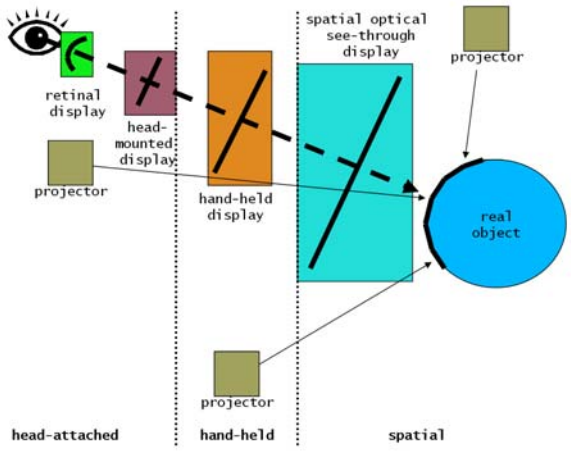


Figure 2.1: *Image-generation for augmented reality displays.*

Head-attached displays, such as retinal displays, head-mounted displays, and head-mounted projectors have to be worn by the observer. While some displays are *hand-held*, others are *spatially aligned* and completely detached from the users. Retinal displays and several projector-based approaches form curved images – either on the observer's retina or directly on the physical object. Most of the displays, however, form images on planes – called *image-planes* – that can be either head-attached or spatially aligned. Images behind real objects cannot be formed by a display that is located in front of real objects. In addition, if images are formed behind a real object, this object will occlude the image portion that is required to support augmentation.

Several pros and cons can be found by comparing the different types of displays. Most of them will be discussed within the following sections.

If *stereoscopic rendering* is used to present mixed (real and virtual) worlds, two basic fusion technologies are currently being used: video-mixing and optical combination.

While *video-mixing* merges live record video streams with computer generated graphics and displays the result on the screen, *optical combination* generates an optical image of the real screen (displaying computer graphics) which appears within the real environment (or within the viewer's visual field while observing the real environment). Both technologies entail a number of advantages and disadvantages which influence the type of application they can address.

Today, most of the stereoscopic AR displays require to wear some sort of goggles to provide stereo separation. Auto-stereoscopic approaches, however, might play a dominant role in the future of AR.

In this chapter, we discuss several types of augmented reality displays. Note that we rather present examples in each display category, than to provide a complete list of individual devices.

2.1. Head-Attached Displays

Head-attached displays require the user to wear the display system on his/her head. Depending on the image generation technology, three main types exist: Retinal displays that apply low power lasers to project images directly onto the retina of the eye, head-mounted displays that use miniature displays in front of the eyes, and head-mounted projectors that make use of miniature projectors or miniature LCD panels with backlighting and project images on the surfaces of the real environment.

2.1.1. Retinal Displays

Retinal displays [Kol93, Pry98, Lew04] utilize low-power semiconductor lasers (or –in future– special light-emitting diods) to scan modulated light directly onto the retina of the human eye, instead of providing screens in front of the eyes. This produces a much brighter and higher resolution image with a potentially wider field of view than a screen-based display.

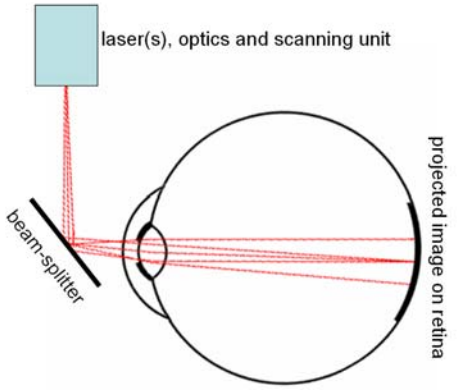


Fig 2.2: *Simplified diagram of a retinal Display.*

Current retinal displays share many shortcomings with head-munted displays (see section 2.1.2). However, some additional disadvantages can be identified for existing versions:

- Only monochrome (red) images are presented since cheap low-power blue and green lasers do not yet exist;
- The sense of ocular accommodation is not supported due to the complete bypass of the ocular motor-system by scanning directly onto the retina. Consequently, the focal length is fixed;
- Stereoscopic versions do not exist.

The main advantages of retinal displays are the high brightness and contrast, and low power consumption – which make them well suited for mobile outdoor applications. Future generations also hold the potential to provide dynamic re-focus, full-color stereoscopic images, and an extremely high resolution and large field-of-view.

2.1.2. Head-Mounted Displays

Head-mounted displays (HMDs) are currently the display devices which are mainly used for augmented reality applications.

Two different head-mounted display-technologies exist to superimpose graphics onto the user's view of the real world: *Video see-through head-mounted displays* that make use of video-mixing and display the merged images within a closed-view head-mounted display, or *optical see-through head-mounted displays* that make use of optical combiners (essentially half-silvered mirrors or transparent LCD displays). A comparison between these two general technologies can be found in [Rol94].

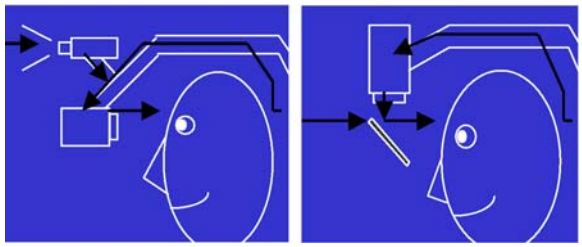


Figure 2.3: *Video see-through (left) and optical see-through (right). Courtesy: Azuma [Azu97].*

Several disadvantages can be related to the application of head-mounted displays as an augmented reality device. Note that most of these shortcomings are inherited from the general limitations of head-attached display technology:

- Lack in resolution that is due to limitations of the applied miniature displays. In the optical see-through case, only the graphical overlays suffer from a relatively low resolution, while the real environment can be perceived in the resolution of the human visual system. For video see-through devices, both – the real environment and the graphical overlays – are perceived in the resolution of the video-source or display;
- Limited field of view that is due to limitations of the applied optics;
- Imbalanced ratio between heavy optics (that results in cumbersome and uncomfortable devices) and ergonomic devices with a low image quality;
- Visual perception issues that is due to the constant image depth. For optical see-through: Since objects within the real environment and the image plane that is attached to the viewer's head are sensed at different depths, the eyes are forced to either continuously shift focus between the different depth levels, or perceive one depth level

unsharp. This is known as the *fixed focal length problem*, and is more critical for see-through than for closed-view head-mounted displays. For video see-through: Only one focal plane exists – the image-plane;

- Optical see-through devices require difficult (user and session dependent) calibration and precise head-tracking to ensure a correct graphical overlay. For video see-through, graphics can be integrated on a pixel-precise basis, but image processing for optical tracking increases the end-to-end system delay;
- Increased incidence of discomfort due to simulator sickness because of head-attached image plane (especially during fast head movements) [Pat00];
- Conventional optical see-through devices are incapable of providing consistent occlusion effects between real and virtual objects. This is due to the mirror beam splitters that reflect the light of the miniature displays which interferes with the transmitted light of the illuminated real environment. To solve this problem, Kiyokawa et al. [Kiy00] apply additional LCD panels to selectively block the incoming light with respect to the rendered graphics.

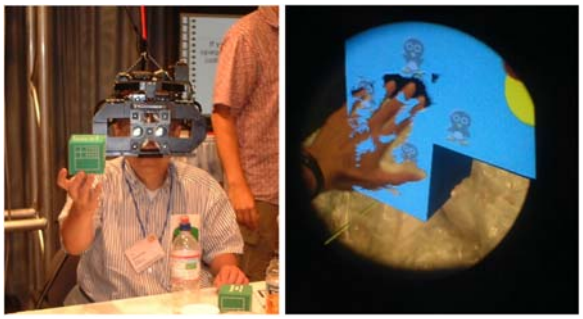


Figure 2.4: *Osaka University's ELMO – An optical see-through head-mounted display that provides mutual occlusion by using a see-through LCD panel in front of the HMD optics [Kiy00]. Courtesy: Kiyokawa.*

Head-mounted displays are currently the dominant display technology within the AR field. They support mobile applications and multi-user applications, if a large number of users need to be supported.

Some variations of head-mounted displays exist that are more attached to the real environment than to the user. Optical see-through boom-like displays (e.g., Osaka University's ELMO [Kiy00]) or video see-through, application-adopted devices (e.g., the head-mounted operating microscope [Fig02]) represent only two examples.

2.1.3. Head-Mounted Projectors

Head-mounted projective displays [Par98, Ina00, Hua01] redirect the projection frustum with a mirror beam-splitter so that the images are beamed onto *retro-reflective surfaces* that are located in front of the viewer. A retro-reflective surface is covered with many thousands of micro corner cubes. Since each micro corner cube has the unique optical property to reflect light back along its incident direction, such surfaces reflect brighter images than normal surfaces that diffuse light. Note that this is similar in spirit to the holographic films used for transparent projection screens. However, these films are back-projected while retro-reflective surfaces are front-projected.

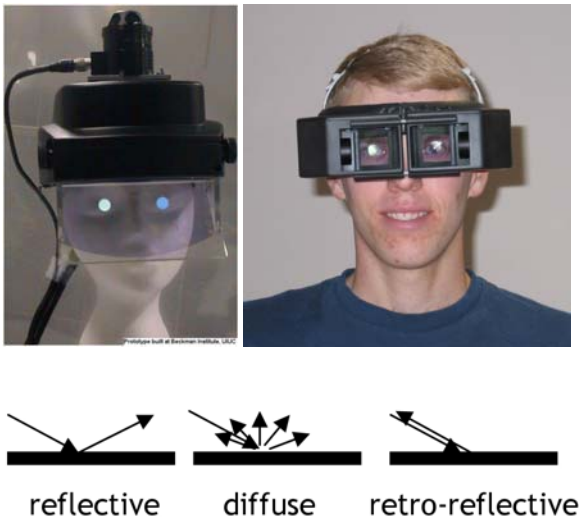


Figure 2.5: *Examples of head-mounted projector prototypes [Hua01] (top). Courtesy: Hua, Gao, Brown, Ahuja, and Rolland. Reflection-properties of retro-reflective material (bottom).*

Projective head-mounted displays [Kij97] beam the generated images onto regular ceilings, rather than onto special surfaces that face the viewer. Two half-silvered mirrors are used to integrate the projected image into the viewer's visual field so that the projectors' parameters match the viewer's parameters (i.e., projection/viewing center and frustum).

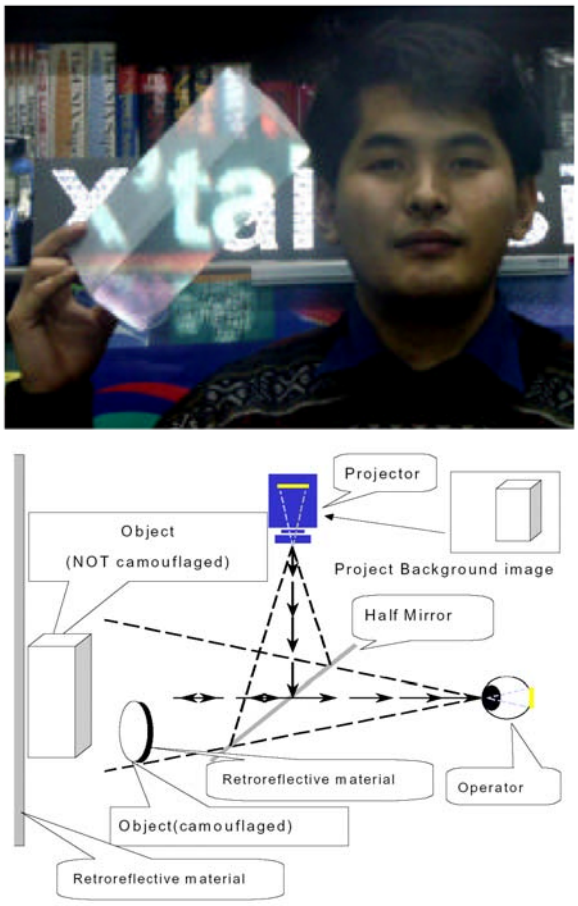


Figure 2.6: *Example of how HMPDs are used to make things transparent – Optical Camouflage [Ina01]. Courtesy: Inami, Kawakami, Sekiguchi, Yanagida, Maeda, and Tachi.*

Head-mounted projective displays decrease the effect of inconsistency of accommodation and convergence that is related to HMDs. Both, head-mounted projective displays and projective head-mounted displays also address other problems that are related to HMDs: They provide a larger field of view without the application of additional lenses that introduce distorting arbitrations. They also prevent incorrect parallax distortions caused by IPD (inter-pupil distance) mismatch that occurs if HMDs are worn incorrectly (e.g., if they slip slightly from their designed position). However, they also introduce several shortcomings:

- Both, head-mounted projective displays and projective head-mounted displays are currently cumbersome. However, new prototypes tend to be smaller and more ergonomically to wear;
- The integrated miniature projectors/LCDs offer limited resolution and brightness;
- Head-mounted projective displays might require special display surfaces (i.e., retro-reflective surfaces) to provide bright images;
- For projective head-mounted displays, the brightness of the images depends on the environmental light conditions;
- Projective head-mounted displays can only be used indoors, since they require the presence of a ceiling.

Although such displays technically tend to combine the advantages of projection displays with the advantages of traditional HMDs, their cumbersomeness currently prevents them from being applicable outside research laboratories. Like head-attached displays in general, they suffer from the imbalanced ratio between heavy optics (or projectors) that results in cumbersome and uncomfortable devices or ergonomic devices with a relatively poor image quality.

2.2. Hand-Held Displays

Conventional examples of *hand-held displays*, such as Tablet PCs, personal digital assistants (PDAs) [Fru01, Gei01, Gau03, Pas03, Wag03], or –more recently– cell-phones [Moe04] generate images at arm reach. All of these examples combine processor, memory, display, and interaction technology in one single device, and aim at supporting a wireless and unconstrained mobile handling. Video see-through is the preferred concept for such approaches. Integrated video cameras capture live video streams of the environment that are overlaid by graphical augmentations before displaying them.

Figure 2.7: *AR application running on a PocketPC [Wag03] (top, courtesy: TU-Vienna), and a first prototype on a conventional consumer cell-phone [Moe04] (bottom).*

However, optical see-through hand-held devices exist. Stetton et al [Ste01], for instance, has introduced device for overlaying real-time tomographic data over the patient. It consists of an ultrasound transducer that scans ultrasound slices of objects in front of it. The slices are displayed time-sequentially on a small flat-panel monitor and are then reflected by a planar half-silvered mirror in such a way that the virtual image is exactly aligned with the scanned slice area. Stereoscopic rendering is not required in this case, since the visualized data is two-dimensional and appears at its correct three-dimensional location.

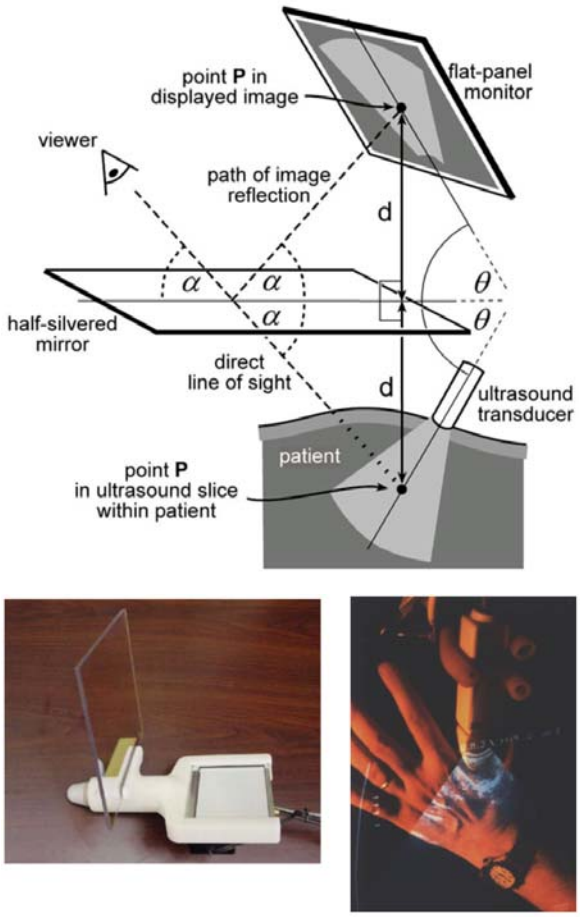


Figure 2.8: *Example of a hand-held mirror display - The Sonic Flashlight [Ste01]. Courtesy: Stetten, Chib, Hildebrand, and Bursee.*

Hand-held mirror beam splitters can be used in combination with large, semi-immersive or immersive screens to support augmented reality tasks with rear-projection systems [Bim00]. Tracked mirror beam-splitters act as optical combiners that merge the reflected graphics, which are displayed on the projection plane, with the transmitted image of the real environment.

Figure 2.9: *The Transflective Pad [Bim00] – a hand-held mirror beam-splitter in combination with a large rear-projection screen.*

Yet another interesting display concept was described in [Ras03] and proposes the application of a hand-held video projector as a real flashlight to interactively generate shadow effects of virtual objects on real surfaces. A combination of a hand-held video projector and a camera was also used by Foxlin and Naimark of Intersense to demonstrate the capabilities of their optical tracking system. This concept might represent an interesting application of AR to the fields of architecture and maintenance.

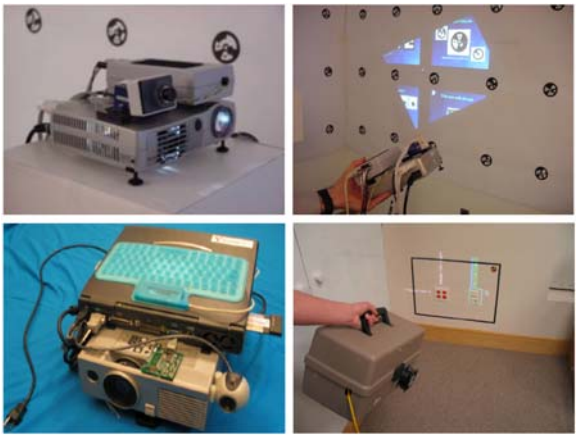


Figure 2.10: *AR Flashlight (top): augmenting the world with a tracked handheld projector. Courtesy: InterSense Inc., Foxlin and Naimark. Context aware iLamp (bottom): augmenting of an identified surface [Ras03].*

The following disadvantages can be related to the individual approaches:

- The image analysis and rendering components is processor and memory intensive. This is critical for low-end devices such as PDAs and cell-phones, and might result in a too high end-to-end system delay and low frame rates. Such devices often lack a floating point unit – which makes precise image processing and fast rendering even more difficult;
- The limited screen size of most hand-held devices restricts the covered field-of-view. However, moving the mirror to navigate3 through an information space that is essentially larger than the display device supports a visual perception phenomenon that is known as *Parks effect* [Par65]. That is, moving a scene on a fixed display is not the same as moving a display over a stationary scene because of the persistence of the image on the viewer's retina. Thus, if the display can be moved, the effective size of the virtual display can be larger than its physical size, and a larger image of the scene can be left on the retina. This effect was also pointed out by the early work of Fitzmaurice [Fiz93] on mobile VR devices;
- The optics and image sensor chips of integrated cameras in consumer hand-held devices is targeted to other applications and consequently provide a limited quality for image processing tasks (e.g., usually high barrel distortion). For example, they do not support an auto-focus. Fixed focus cameras can only be effective in a certain depth range. This also applies to the image output of hand-held projectors if they are used to augment surfaces with a certain depth variance;
- Compared to head-attached devices, hand-held devices do not provide a completely hands-free working.

Hand-held devices represent a real alternative to head-attached devices for mobile applications. Especially consumer devices, such as PDAs and cell-phones have the potential to bring AR to a mass market.

As it is the case for PCs, the graphics capabilities of cell-phones are clearly driven by the game industry. Vendors, such as ATI and nVidia, already offer the first 3D graphics acceleration chips for cell-phones. Autostereoscopic displays are available for off-the-shelf phones (e.g., Sharp) for viewing graphics in 3D. The processors of phones are becoming continuously faster and memory restrictions like today will become history.

2.3. Spatial Displays

In contrast to body-attached displays (head-attached or hand-held), *spatial displays* detach most of the technology from the user and integrate it into the environment. Three different approaches exist which mainly differ in the way they augment the environment – either using video see-through, optical see-through or direct augmentation.

2.3.1. Screen-Based Video See-Through Displays

Screen-based augmented reality has sometimes been referred to as window on the world [Fei93]. Such systems make use of video-mixing (video see-through) and display the merged images on a regular monitor.

Figure 2.11: *Example for a screen-based video see-through display.The locomotion of a dinosaur is simulated over a physical foot-print [Bim02b].*

As fish tank virtual reality systems which also apply monitors, window on the world setups provide a low degree of immersion. Within an augmented reality context the degree of immersion into an augmented real environment is frequently expressed by the amount of the observer's visual field (i.e., the field of view) that can be superimposed with graphics. In case of screen-based augmented reality, the field of view is limited and restricted to the monitor size, its spatial alignment relative to the observer, and its distance to the observer.

For screen-based augmented reality, the following disadvantages can be found:

- Small field of view that is due to relatively small monitor sizes. However, the screen-size is scalable if projection displays are applied;
- Limited resolution of the merged images (especially dissatisfying is the limited resolution of the real environment). This is a general disadvantage of video see-through [Rol94];
- Mostly provides a remote viewing, rather than supporting a see-through metaphor;
- Direct interaction with the real environment and the graphical augmentation is usually not supported. Only indirect / remote interaction techniques can be supported;

Screen-based augmentation is a quite common technique if mobile applications or optical see-through do not have to be supported. It represents probably the most cost efficient AR approach, since only off-the-shelf hardware components and standard PC equipment is required.

2.3.2. Spatial Optical See-Through Displays

In contrast to head-attached or hand-held optical see-through displays, *spatial optical see-through displays* generate images that are aligned within the physical environment. Spatial optical combiners, such as planar [Bim01a, Bim01b] or curved [Bim01a, Bim03] mirror beam splitters, transparent screens [Ogi01, Bim04b], or optical holograms [Bim04a] are essential components of such displays. This class of augmented reality displays is described in much more detail in chapter 3.

Figure 2.12: *A monitor-based Virtual Showcase variation [Bim01a] (top) and the Extended Virtual Table [Bim01b] (bottom) using a large beam-splitter and projection screen.*

The following shortcomings are related to spatial optical see-through configurations:

- They do not support mobile applications because of the spatially aligned optics and display technology;
- In most cases, the applied optics prevents a direct manipulative interaction with virtual and real objects that are located behind the optics. Exceptions are reach-in configurations – either realized with see-through LCD panels [Sch02] or mirror-beam splitters [www.arsys-tricorder.de];
- The number of observers that can be supported simultaneously is restricted by the applied optics. Multi-user displays such as the Virtual Showcase [Bim01a] and its variations support four and more users;
- For the same reasons as for optical see-through head mounted displays, a mutual occlusion between real and virtual environment is not supported. Projector-based illumination techniques which solve this problem [Bim02a] are discussed in chapter 4;
- Due to the limited size of screens and optical combiners, virtual objects outside the display area are unnaturally cropped. This effect is known as window violation and is also present for fish-tank and semi-immersive virtual reality displays.

Figure 2.13: *Transparent screen (top, courtesy: LaserMagic) and Ogi's invisible interface (bottom) [Ogi01]. Courtesy: Ogi, Yamada, Yamamoto, and Hirose.*

The general advantages of such displays are an easier eye accommodation and vergence, a higher and scalable resolution, a larger and scalable field of view, improved ergonomic factors, an easier and more stable calibration, and a better controllable environment. Chapter 3 will also discuss these issues in more detail.

Figure 2.14: *Example of a screen-based AR display using a see-through LCD panel – The AR window (upper-left and -right) [Sch02]. Courtesy: ZGDV, Schwald. Example of a mirror-based AR display using optical see-through beam splitters (lower-left and -right). Courtesy: Fraunhofer IMK (www.arsys-tricoder.de).*

2.3.3. Projection-Based Spatial Displays

Projector-based spatial displays apply front-projection to seamlessly project images directly on physical objects' surfaces instead of displaying them on an image plane (or surface) somewhere within the viewer's visual field. Single static [Ras98, Und99, Bim02b] or steerable [Pin01], and multiple [Ras99, Ras01, Ras02] projectors are applied to increase the potential display area. Chapter 4 will discuss this concept in more detail.

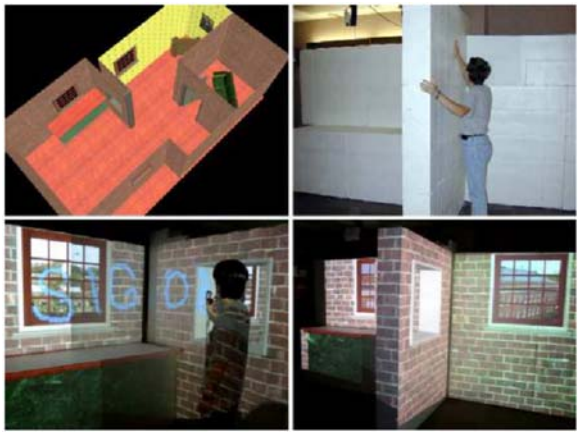


Figure 2.15: *Projector-based augmentation of a large environment [Low01]. Virtual model (upper left). Physical display environment constructed using Styrofoam blocks (upper right). Augmented display (bottom). Note the view dependent nature of the display, the perspectively correct view through the hole in the wall and the windows. Courtesy: Low.*

A stereoscopic projection and consequently the technology to separate stereo images is not necessarily required if only the surface properties (e.g., its color, illumination or texture) of the real objects are changed by overlaying images. In this case a correct depth perception is still provided by the physical depth of the objects' surfaces.

However, if 3D graphics are displayed in front of the object's surfaces, a view-dependent, stereoscopic projection is required as for other oblique screen displays [Ras98].

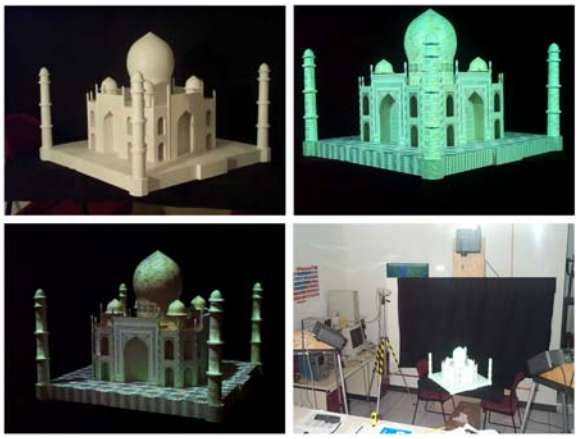


Figure 2.16: *ShaderLamps [Ras01] with TajMahal: The wooden white model (upper left) is illuminated. The scanned geometry of the TajMahal is authored to add texture and material properties. The geometry is registered to the real TajMahal and is displayed from projector's viewpoint.*

On the one hand projector-based spatial displays introduce several new problems:

- Shadow-casting of the physical objects and of interacting users that is due to the utilized front-projection;
- Restrictions of the display area that is constrained to the size, shape, and color of the physical objects' surfaces (for example, no graphics can be displayed beside the objects' surfaces);
- Restricted to a single user in case virtual objects are displayed with non-zero parallax;
- Conventional projectors can only focus on a single focal plane located at a constant distance. Projecting images onto non-planar surfaces causes blur. Exceptions are laser-projectors which do not suffer from this effect;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

On the other hand, they overcome some of the shortcomings that are related to head-attached displays: an improved ergonomics, a theoretically unlimited field of view, a scalable resolution, and an easier eye accommodation (because the virtual objects are typically rendered near their real world location).

Figure 2.17: *The Everywhere Display concepts [Pin01]. Courtesy: IBM, Pinhanez.*

References

[Azu97] Azuma, R. T., A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments, vol. 6, no. 4, pp. 355-385, 1997.

[Bim00] Bimber, O., Encarnação, L.M., and Schmalstieg, D. Augmented Reality with Back-Projection Systems using Transflective Surfaces. Computer Graphics Forum (proceedings of EUROGRAPHICS 2000 - EG'2000), vol. 19, no. 3, pp.161-168, 2000.

[Bim01a] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55, 2001.

[Bim01b] Bimber, O., Encarnação, L.M. and Branco, P. The Extended Virtual Table: An Optical Extension for Table-Like Projection Systems. Presence: Teleoperators and Virtual Environments, vol.10, no. 6, pp. 613-631, 2001.

[Bim02a] Bimber, O. and Fröhlich, B. Occlusion Shadows: Using Projected Light to Generate Realistic Occlusion Effects for View-Dependent Optical See-Through Displays. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 186-195, 2002.

[Bim02b] Bimber, O., Gatesy, S.M., Witmer, L.M., Raskar, R. and Encarnação, L.M. Merging Fossil Specimens with Computer-Generated Information. IEEE Computer, September, pp. 45-50, 2002.

[Bim03] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. Real-Time View-Dependent Image Warping to correct Non-Linear Distortion for Curved Virtual Showcase Displays. To appear in Computers and Graphics - The international Journal of Systems and Applications in Computer Graphics, vol. 27, no. 4, 2003.

[Bim04a] Bimber, O. Combining Optical Holograms with Interactive Computer Graphics. In IEEE Computer, January issue, pp. 85-91, 2004.

[Bim04b] Bimber, O., Coriand, F., Kleppe, A., Bruns, E., Zollmann, S., and Langlotz, T. Superimposing Pictorial Artwork with Projected Imagery. To appear in IEEE MultiMedia, 2004.

[Fei93] Feiner, S., MacIntyre, B., et al. Windows on the World: 2D Windows for 3D Augmented Reality. In proceedings of ACM Symposium on User Interface Software and Technology, pp. 145-155, 1993.

[Fig02] Figl, M., Birkfellner, W., Ede, C., Hummel, J., Hanel, R. Watzinger, F., Wanschitz, F., Ewers, R., and Bergmann H. The Control Unit for a Head Mounted Operating Microscope used for Augmented Reality Visualization in Computer Aided Surgery. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 69-76, 2002.

[Fiz93] Fitzmaurice, G.W. Situated Information Spaces and Spatially Aware Palmtop Computer. CACM, vol. 35(7), pp. 38-49, 1993.

[Fru01] Fründ, J.; Geiger, C.; Grafe, M.; Kleinjohann, B.: The augmented reality personal digital assistant. In proceedings of International Symposium on Mixed Reality, 2001.

[Gau03] Gausemeier, J., Fruend, J., Matysczok, C., Bruederlin, B., and Beier, D. Development of a real time image based object recognition method for mobile AR-devices. In proceedings of International Conference on Computer graphics, Virtual Reality, Visualisation and Interaction in Africa, pp. 133-139, 2003.

[Gei01] Geiger, C., Kleinjohann, B., Reimann, C. Stichling, D. Mobile Ar4All. In proceedings of IEEE and ACM International Symposium on Augmented Reality, pp. 181-182, 2001.

[Hua01] Hua, H., Gao, C., Brown, L., Ahuja, N., and Rolland, J.P. Using a head-mounted projective display in interactive augmented environments. In Proceedings of

IEEE and ACM International Symposium on Augmented Reality 2001, pp. 217-223, 2001.

[Ina00] Inami, M., Kawakami, N., Sekiguchi, D., Yanagida, Y., Maeda, T. and Tachi, S., Visuo-Haptic Display Using Head-Mounted Projector, Proceedings of IEEE Virtual Reality 2000, pp.233-240, 2000.

[Kij97] Kijima, R. and Ojika, T. Transition between virtual environment and workstation environment with projective head-mounted display, In proceedings of IEEE Virtual Reality Annual International Symposium, pp.130-137, 1997.

[Kiy00] Kiyokawa, K., Kurata, Y. and Ohno, H. An Optical See-through Display for Mutual Occlusion of Real and Virtual Environments. In proceedings of IEEE & ACM ISAR 2000, pp. 60-67, 2000.

[Kol93] Kollin, J. A Retinal Display For Virtual-Environment Applications. In proceedings of SID International Symposium, Digest Of Technical Papers, pp. 827, 1993.

[Lew04] Lewis, J.R. In the eye of the beholder. IEEE Spectrum, May issue, pp. 16-20, 2004

[Low01] Low, K., Welch, G., Lastra, A., and Fuchs, H. Life-Sized Projector-Based Dioramas. Symposium on Virtual Reality Software and Technology, 2001.

[Moe04] Moehring, M., Lessig, C., and Bimber, O. Video see-through AR on consumer cell-phones. Submitted to IEEE/ACM ISMAR'04, 2004.

[Ogi01] Ogi, T., Yamada, T., Yamamoto, K. and Hirose, M. Invisible Interface for Immersive Virtual World. In proceedings of the Immersive Projection Technology Workshop (IPT'01), pp. 237-246, Stuttgart, Germany, 2001.

[Par98] Parsons, J., and Rolland, J.P., A non-intrusive display technique for providing real-time data within a surgeons critical area of interest, In proceedings of Medicine Meets Virtual Reality'98, pp. 246-251, 1998.

[Par65] Parks, T.E. Post Retinal Visual Storage. American Journal of Psychology, vol. 78, pp. 145-147, 1965.

[Pas03] Pasman, W. and Woodward, C. Implementation of an augmented reality system on a PDA. In proceedings of IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 276-277, 2003.

[Pat00] Patrick, E., Cosgrove, D., Slavkovic, A., Rode, J.A., Verratti, T., and Chiselko, G. Using a Large Projection Screen as an Alternative to Head-Mounted Displays for Virtual Environments. In proceedings of CHI' 2000, vol. 2, no. 1, pp. 479-485, 2000.

[Pin01] Pinhanez, C. The everywhere displays projector: A device to create ubiquitous graphical interfaces, In proceedings of Ubiquitous Computing, pp. 315-331, 2001.

[Pry98] Pryor, Homer L., Furness, Thomas A. and Viirre, E. The Virtual Retinal Display: A New Display Technology Using Scanned Laser Light. In proceedings of Human Factors and Ergonomics Society, 42nd Annual Meeting, pp. 1570-1574, 1998.

[Ras98] Raskar, R., Welch, G., and Fuchs, H. Spatially Augmented Reality. In proceedings of First IEEE Workshop on Augmented Reality (IWAR'98), San Francisco, CA, pp. 63-72, 1998.

[Ras99] Raskar, R., Welch, G., and Chen, W-C. Table-Top Spatially Augmented Reality: Bringing Physical Models to Life with Projected Imagery. In proceedings of Second International IEEE Workshop on Augmented Reality (IWAR'99), San Francisco, CA, pp. 64-71, 1999.

[Ras01] Raskar, R., Welch, G., Low, K.L. and Bandyopadhyay, D. Shader Lamps: Animating real objects with image-based illumination, In proceedings of Eurographics Rendering Workshop, pp. 89-102, 2001.

[Ras02] Raskar, R., Ziegler, R. and Willwacher, T. Cartoon Dioramas in Motion, In proceedings of Int. Symp. on Non-photorealistic Animation and Rendering, pp. 7-ff, 2002.

[Ras03] Raskar, R., van Baar, J., Beardsly, P., Willwacher, T., Rao, S. and Forlines, C. iLamps: Geometrically Aware and Self-Configuring Projectors, In proceedings of ACM Siggraph, pp. 809-818, 2003.

[Rol94] Rolland, J., Rich, H. and Fuchs, H. A Comparison of Optical and Video See-Through Head-Mounted Displays. In proceedings of SPIE: Telemanipulator and Telepresence Technologies, pp. 293-307, 1994.

[Sch02] Schwald, B., Seibert, H., Weller, T. A Flexible Tracking Concept Applied to Medical Scenarios Using an AR Window. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 261-262, 2002.

[Ste01] Stetten, G., Chib, V., Hildebrand, D., and Bursee, J. Real Time Tomographic Reflection: Phantoms for Calibration and Biopsy, In proceedings of IEEE/ACM International Symposium on Augmented Reality (ISMAR'01), pp. 11-19, 2001.

[Und99] Underkoffler, J., Ullmer, B. and Ishii, H. Emancipated pixels: real-world graphics in the luminous room, In proceedings of ACM Siggraph, pp. 385-392, 1999.

[Wag03] Wagner, D., and Schmalstieg, D. First steps towards handheld augmented reality. In proceedings of International Conference on Wearable Computers, pp. 127-136, 2003.

3. Generating Graphical Overlays with Spatial Optical See-Through Displays

Spatial optical see-through displays overlay the real environment with computer graphics in such a way that the graphical images and the image of the real environment are visible at the same time. In contrast to head-attached or body attached optical see-through displays, spatial displays generate images that are aligned within the physical environment. They do not follow the users' movements but rather support moving around them. Consequently they are comparable with spatial projection displays – but do not share the opaque characteristic of such displays.

An essential component of optical see-through display is the *optical combiner* – an optical element that mixes the light emitted by the illuminated real environment with the light produced with an *image source* that displays the rendered graphics. Creating graphical overlays with spatial optical see-through displays is similar to rendering images for spatial projection screens for some optical combiners. For others, however, it is more complex and requires additional steps before the rendered graphics is displayed and optically combined.

While monitors, diffuse projection screens, or video projectors usually serve as light emitting image sources, two different types of optical combiners are normally being used for such displays: *transparent screens* and *half-silvered mirror beam combiners*. Rendering techniques that support creating correct graphical overlays with both types of optical combiners and with different images sources will be discussed below. Before going into details, we make the convention that all following elements, such as optical combiners, image sources, observers, virtual and real environments, etc. are defined within the same *Cartesian coordinate system*. This keeps the explanations particularly simple and allows a straight forward implementation of the techniques.

3.1. Transparent Screens

Transparent screens have two main properties: they are transparent up to a certain degree to transmit the image of the real environment, and they emit the light of the rendered graphics. Observers see directly through the screen (and through the image displayed on it) at the real environment.

In some cases, such screens are *active* and contain light emitting elements. Examples are liquid crystal displays that are modified (by removing the opaque back light source) to enable their see-through capabilities. In other cases external image sources, such as video or laser projectors

are applied to generate light that is diffused by a *transparent projection screen*. Examples include *holographic projection screens* that diffuse the light in a narrow angle to achieve an enhanced brightness for restricted viewing angles, or *transparent film screens* that diffuse the light in a wide angle to support a more flexible viewing and a higher degree of transparency.

On the one hand, the use of video projectors allows building large screens that do not necessarily have to be planar. Active screens, on the other hand, provide more flexibility since they are not constrained to a stationary configuration. In future new materials, such as *light emitting polymers* may allow building active transparent screens that are both – flexible and scalable.

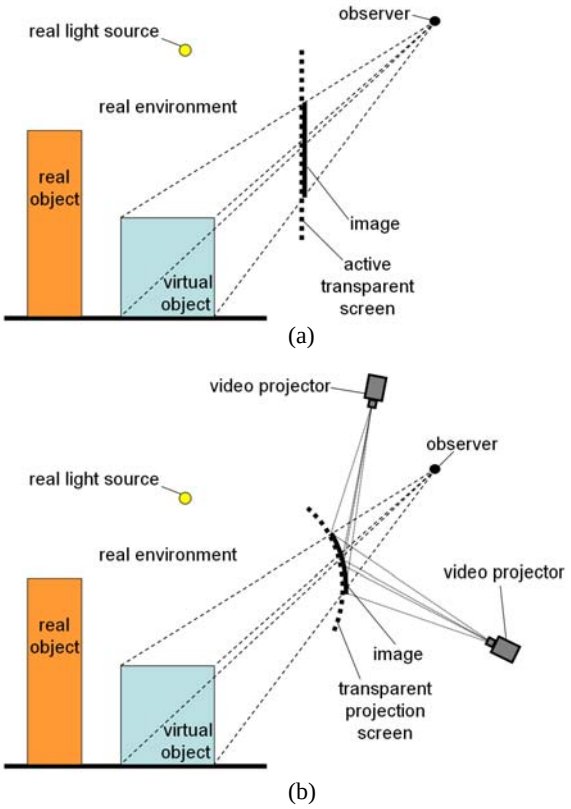


Figure 3.1: *Planar active transparent screen (a) and curved transparent projection screen with multiple projectors (b).*

Rendering for transparent screens is essentially equivalent to rendering for regular projection screens or monitors. While for planar screens an affine *off-axis projection* transformation is sufficient, *curvilinear image warping* is required for curved screens. For large or extremely curved screens the image has to be composed of the contribution displayed by multiple projectors. The different pieces have to be geometrically aligned and blended to result in a consistent final image.

Sections 3.3 through 3.5 describe a rendering framework spatial optical see-through displays that apply mirror beam combiners. Most of this framework, such as refraction correction, multi-screen and multi-plane beam combiner transformations is exactly the same for transparent screens. The main difference to mirror beam combiner is that reflection transformations do not apply in this case. They have to be ignored if the framework is used for displays with transparent screens. In addition, the outlined screen transformation is equivalent to the transformation of video projectors.

An important fact that needs to be mentioned is that the rendered image appears directly on the surface of the transparent screen. This cause *focus problems* if the real environment to be augmented is not located at the same place as the image. For transparent screens, this can never be the case since the screen itself (and therefore the image) can never take up exactly the same space within the environment. It is not possible for our eyes to focus on multiple distances simultaneously. Thus in extreme cases we can only continuously shift focus between the image and the real environment, or perceive either one unfocused.

Attention has to be paid if stereoscopic graphics needs to be displayed on transparent projection screens. Not all materials preserve the polarization of light that is produced by *passive stereoscopic projection* setups – especially when materials are bended to form curved screens. Transparent film screens, for instance, have the property to preserve polarization in any case – also if bended. Active stereo projection systems do not cause such problems, but are more expensive.

3.2. Mirror Beam Combiners

Mirror beam combiners are the most common optical combiners because of their availability and low cost. If not obtained from an optics store, they can be home made in almost any size and shape. For instance, regular float glass or Plexiglas can be coated with a *half-silvered film*, such as 3M's Scotchtint sun protection film. These materials are easily available in well-sorted hardware stores. The advantage of half-silvered film is that it can be coated onto

a flexible carrier, such as thin Plexiglas, that can easily be bended to build curved displays. The drawback of such a material is usually its non-optimal optical properties. Instead of having transmission and reflection factors of 50%, they normally provide factors of approximately 70% reflection and 30% transmission, due to their sun blocking functionality. An alternative is the so-called *spyglass* that offers better and varying transmission/reflection factors without sun blocking layers. However, the reflective film is usually impregnated into float glass – thus it is not well suited for building curved mirror displays.

Spatial optical see-through displays that apply mirror beam combiners as optical combiner have to display the rendered graphics on a *secondary screen* that is reflected by the mirror optics. The observer sees through the optical combiner and through the reflected image at the real environment. The image that appears on the secondary screen should usually not be visible. To hide it the secondary screen is often being coated with *light directing film*, such as 3M's Light Control Film. Such a film can be used to direct the displayed image only towards the mirror optics – and not to the observer. Thus only the reflection of the displayed image can be seen, and not the image itself.

In contrast to transparent screens, mirror beam combiners create an optical reflection of the secondary screen and the displayed image. There are no physical constraints in aligning the image closer to the real environment. The reflected image can even intersect with the real environment. Consequently, the focus problem of transparent screens can be improved – although not completely solved.

3.3. Planar Mirror Beam Combiners

To render a three-dimensional graphical scene on spatial optical see-through displays applying planar mirror beam combiners requires neutralizing the optical transformations that are caused by half-silvered mirrors: reflection and refraction. The goal is to render the graphical content and display it on the secondary screen in such a way that its reflection appears perspectively correct and aligned with the real environment.

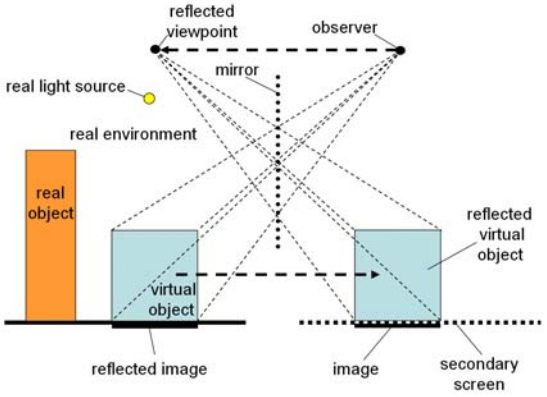
3.3.1. Reflection

A planar mirror beam combiner divides the environment into two subspaces: the one that contains the observer and the secondary screen, and the one that contains the real environment to be augmented and physical light sources that illuminate it.

Note that from a geometric optics point of view, the real environment behind the mirror equals the *mirror's image space* (i.e., the reflection that appears behind the mirror by looking at it).

Virtual objects that consist of graphical elements (such as geometry, normal vectors, textures, clipping planes, virtual light sources, etc.) are defined within same global world coordinate system in which the real environment, the observer, the secondary screen and the mirror beam combiner are located. In contrast to conventional Virtual Reality scenarios, this coordinate system actually exceeds the boundaries of the display screen and extends into the surrounding real environment.

The virtual objects are either defined directly within the real environment, or they are transformed to it during an *object registration process*.



(a)

(b) (c)

Figure 3.2: *Affine reflection transformation for planar mirror beam combiner . A virtual baby observed through a large beam combiner reflecting a horizontal projection screen (b). Due to parallax effects virtual objects (e.g., the baby's arm) can appear in front of the mirror (c).*

We now consider the mirror and compute the reflection of the observer's physical eye locations (as well as possible virtual headlights). We then apply the inverse reflection to every graphical element that is located within the real environment. In this manner these graphical elements are transformed and can be projected and displayed on the secondary screen. The displayed image is optically reflected back by the mirror into the real environment (or optically: into the mirror's image space).

When the setup is sufficiently calibrated, the real environment and the mirror's image space overlay exactly. The virtual objects appear in the same position within the image space as they would within the real environment without the mirror (if a direct display possibility was given within the real environment).

Note that the *reflection transformation* of planar mirrors is a *rigid-body transformation*, and preserves all properties of the transformed geometry.

With known plane parameters of the mirror beam combiner within the world coordinate system, a point in 3D space can be reflected with:

$$p' = p - 2(np + d)n$$

Where p' is the reflection of p over the mirror plane $[n,d] = [a,b,c,d]$.

This is equivalent to multiplying p with the 4x4 reflection matrix:

$$R = \begin{bmatrix} 1-2a^2 & -2ab & -2ac & -2ad \\ -2ab & 1-2b^2 & -2bc & -2bd \\ -2ac & -2bc & 1-2c^2 & -2cd \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note that $R = R^{-1}$.

The reflected viewpoint e' of the observer (which, for instance, is head-tracked) can be computed with the equation above or by multiplying the original viewpoint e with the reflection matrix.

The inverse reflection of the virtual scene that is located within the real environment is simply computed from its reflection with respect to the mirror plane. Since we assume that the real environment and the mirror's image space exactly overlay, we can also assume that the reflection of the graphical elements located within the real environment results in the inverse reflection of the image space, that is, they are transformed to their corresponding positions on the observer's side of the mirror and can be displayed on the secondary screen. Consequently, the additional model transformation (i.e., the inverse reflection of the scene) is achieved by multiplying the reflection matrix onto the current model-view matrix of the *transformation pipeline* (between scene transformation and view transformation).

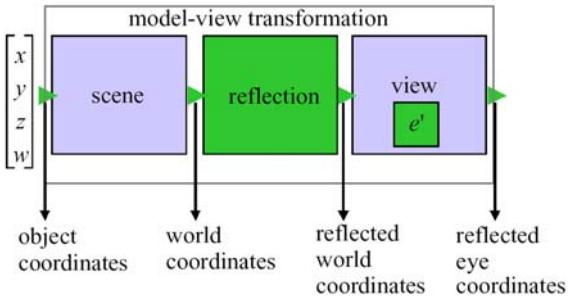


Figure 3.3: *The integration of reflection transformations into the rendering pipeline.*

Consequently, geometry that is registered to the real environment is first transformed from object coordinates into the coordinates of the world coordinate system, then into *reflected world coordinates*, and finally into *reflected eye coordinates*. After the model-view transformation has been applied, the rendering pipeline is continued in the normal way: the reflected eye coordinates are off-axis projected into clip coordinates, then –after the perspective division– transformed into normalized device coordinates, and finally (via the viewport transformation) mapped into window coordinates.

By applying the reflection matrix, every graphical element is reflected with respect to the mirror plane. A side effect of this is that the order of reflected polygons is also reversed (e.g., from counterclockwise to clockwise) which, due to the wrong front-face determination, results in a wrong rendering (e.g., lighting, culling, etc.). This can easily be solved by explicitly reversing the *polygon order*. Note that transformations and rendering have to be done for both viewpoints (left and right) if stereoscopic rendering is activated.

The reflection transformation can be entirely executed by accelerated graphics hardware that provides a *fixed function rendering pipeline*. The following OpenGL code fragment can be used to configure the rendering pipeline with reflection transformation:

```
...
// mirror plane = a,b,c,d
// viewpoint = e[0..2]
// world coordinate system = x/y-axes
// horizontal plane,
// z-axis points up
```

```
float e[3], e_[3];
float NP;
float R[16];

// compute reflected viewpoint e_ from
// original viewpoint e
NP=a*e[0]+b*e[1]+c*e[2];
e_[0]= e[0]-2.0*(NP+d)*a;
e_[1]= e[1]-2.0*(NP+d)*b;
e_[2]= e[2]-2.0*(NP+d)*c;

//set up reflection matrix
R[0]=1-2*a*a;       R[1]=-2*a*b;
R[2]=-2*a*c;        R[3]=0;
R[4]=-2*a*b;        R[5]=1-2*b*b;
R[6]=-2*b*c;        R[7]=0;
R[8]=-2*a*c;        R[9]=-2*b*c;
R[10]=1-2*c*c;      R[11]=0;
R[12]=-2*a*d;       R[13]=-2*b*d;
R[14]=-2*c*d;       R[15]=1;

//configure rendering pipeline
glViewport(…);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glFrustum(…);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(e_[0],e_[1],e_[2],e_[0],e_[1],0
,0,1,0);
glMultMatrixf(R);

//reverse polygon order
glFrontFace(GL_CW+GL_CCW-
glGetIntegerv(GL_FRONT_FACE));

//draw scene with scene transformation
…
```

An alternative to a reflection of scene geometry and viewpoint is to set up a viewing frustum that is defined by the reflected image plane instead of the image plane on the physical secondary screen. In figure 3.2a the secondary screen is reflected from the right sides of the beam combiner to its left side. The unreflected viewing frustum (right side) can be used if defined relative to the reflected image plane (left side). For this, the exact position and orientation of the reflected image plane has to be known. They can also be derived from the parameters of the mirror plane.

In addition, the displayed image has to be reflected over the on-screen axis that is perpendicular to the intersection vector (i_x, i_y) of the mirror plane and the secondary screen. In a simple example, this intersection equals the X- axis (or the Y-axis) of the screen coordinate system. In this case an additional scaling transformation can be applied after the projection transformation that causes the fragments to be reflected within the normalized device space. For instance, glScale$(i_x, i_y, 0)$ with $i_x = -1, i_y = 0$ causes a reflection over the Y-axis for the case that the intersection of the mirror with the screen is on the X-axis (all in normalized screen/device coordinates). For an arbitrary intersection vector, however, a scaling transformation alone is not sufficient. An additional rotation transformation is required that first aligns the intersection vector with either the X- or the Y-axis in the screen coordinate system. Then the scaling is transformed along this principle axis (for example over the X-axis as explained above). Finally the reverse rotation transformation has to be applied to produce the yield the correct effect.

It is important to apply this reflection transformation after the projection (e.g., before glFrustum() in OpenGL's reversed matrix stack notation) since it has to be the final transformation, and it must not influence other computations, such as lighting and depth culling.

An interesting optical effect can be observed by applying mirrors in combination with stereoscopic secondary screens: Convex or planar mirrors can optically only generate virtual images. However, in combination with a stereoscopic graphics and the effects caused by stereopsis, virtual objects can appear in front of the mirror optics (cf. figure 3.2c). We can refer to this effect as *pseudo real images*. In nature, real images of reflected real objects can only be generated with concave mirrors. Note that a restricted direct manipulative interaction with pseudo real images in front of the mirror optics is supported.

3.3.2. Refraction

From an optics point of view, the glass or Plexiglas carriers used for optical combiners (i.e., mirror beam combiner or transparent screens) are lenses that cause *refraction* distortion. A homogeneous medium that is bound by two plane-parallel panels is referred to as *planar lens*. The refraction distortion is small and can be neglected for thin planar lenses, but has to be corrected for thick plates. Note that the techniques described below are also applicable for transparent screens that suffer also from refraction distortion.

The following problem occurs: All virtual objects that are registered to the real environment are virtual points that are not physically located behind the optical combiner. They are images that are created by the optical combiner. Mirror beam combiners are usually *front surface mirrors* (i.e., the mirror film coated on the side of the carrier that

faces the image source and the observer) while transparent screens can be front projected – causing the same registration problem: the displayed image is not physically refracted by the optical combiner. However, the transmitted light which is emitted by the real environment and perceived by the observer is refracted. Consequently, the transmitted image of the real environment cannot be precisely registered to the reflected virtual objects, even if their geometry and alignment match exactly within our world coordinate system.

Unfortunately refraction cannot be undistorted by a rigid-body transformation, but approximations exist that are sufficient for augmented reality display types.

All optical systems that use any kind of see-through element have to deal with similar problems. For head-mounted displays, *aberrations* (optical distortion) caused by refraction of the integrated lenses are mostly assumed to be static [Azu97]. Due to the lack of eye-tracking technology as component of head-mounted displays, the rotation of the eye-balls and the exact position as well as the movement of the optics in front of the observer's eyes is not taken into account. Thus a static refraction distortion is pre-computed for a *centered on-axis* optical constellation with methods of *paraxial analysis*. The result is stored in a *two-dimensional look-up-table*. During rendering, this look-up-table is referenced to transform rendered vertices on the image plane before they are displayed. Rolland and Hopkins [Rol93], for instance, describe a polygon-warping technique that uses the lookup table to map projected vertices of the virtual objects' polygons to their pre-distorted location on the image plane. This approach requires subdividing polygons that cover large areas on the image plane. Instead of pre-distorting the polygons of projected virtual objects, the projected image itself can be pre-distorted, as described by Watson and Hodges [Wat95], to achieve a higher rendering performance.

For spatial see-through displays, however, aberrations caused by refraction are dynamic, since the optical distortion changes with a moving viewpoint that is normally off-axis and off-centered with respect to the optical combiner.

For some *near-field displays*, such as reach-in displays, the displacement caused by refraction can be estimated [Wei99]. An estimation of a constant refraction might be sufficient for near-field displays with a fixed viewpoint that apply relatively thin beam combiner. For larger spatial optical see-through setups that consider a head-tracked observer and apply a relatively thick beam combiner require, however, require more precise methods. In the following explanation, we want to consider such a display constellation:

Since we cannot pre-distort the refracted transmitted image of the real environment, we artificially refract the virtual scene before it is displayed, in order to make both images match. In contrast the static transformation that is assumed for head-mounted displays, refraction is dynamic for spatial optical see-through displays and cannot be pre-computed. In general, refraction is a complex curvilinear transformation and does not yield a *stigmatic mapping* (the intersection of multiple light rays in a single image point) in any case, we can only approximate it.

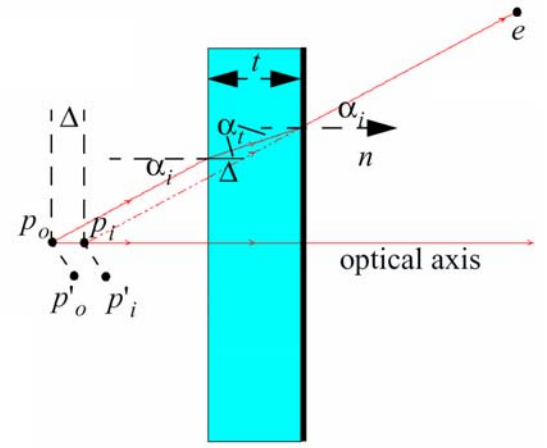


Figure 3.4: *Off-axis refraction transformation for planar lenses.*

In case of planar lenses, light rays are refracted twice – at their entrance points and at their exit points. This is referred to as in-out refraction. In case of planar lenses, the resulting out-refracted light rays have the same direction as the corresponding original rays, but they are shifted by the amount Δ parallel to their original counterparts. Due to refraction, an object that is physically located at the position p_o appears at the position p_i. To ´maintain registration between a virtual object (that will appear at position p_o) and the corresponding real object (that will appear at position p_i), the virtual object has to be re-positioned to p_i.

The offset Δ can be computed with:

$$\Delta = t\left(1 - \frac{\tan \alpha_t}{\tan \alpha_i}\right)$$

where t is the thickness of the planar lens, α_i the angle between the plane normal of the plate and the line of sight, and α_t is given by *Snellius' law of refraction*:

$$\eta_1 \sin \alpha_i = \eta_2 \sin \alpha_t$$

It is constrained to the following boundaries:

$$\lim\left(\alpha_i \to \frac{\pi}{2}\right) \Rightarrow \Delta = t \text{ and}$$

$$\lim(\alpha_i \to 0) \Rightarrow \Delta = t\left(1 - \frac{\sin \alpha_t}{\sin \alpha_i}\right) = t\left(1 - \frac{1}{\eta_2}\right) = const.$$

A special case of the above transformation is an *on-axis* situation where the incidence angle is perpendicular to the lens and parallel with the optical axis (i.e., $\alpha_i = 0$). In this situation the offset equation can be simplified to $\Delta = t(1 - 1/\eta_2)$.

In case of an optical combiner that is located in air, the refraction index η_1 is equal to one. The refraction index η_2 is the one of the carrier's base material (e.g., glass or Plexiglas).

A simple solution to simulate refraction is the assumption that, against its optical nature, it can be expressed as a rigid-body transformation. This is a common approximation of the 3D computer graphics community to render realistic-looking refraction effects in real time. In beam-tracing [Hec84], for instance, it was assumed that, considering only *paraxial rays* (entering light rays that are exactly or nearly perpendicular to the refracting plane), objects seen through a polygon with a refraction index of η appear to be η times their actual distance. This is because light travels slower in denser materials by precisely this factor. For this approximation, the incidence angles of the optical line of sight were not taken into account but. Instead, a constant incidence angle of $\alpha_i = 0$ to the optical axis was assumed. This, however, covers on-axis situations only. For off-axis situations that occur with spatial optical see-through displays the incident angle has to be considered. As a rigid-body transformation, refraction can be expressed by a homogenous 4x4 matrix:

$$F = \begin{bmatrix} 1 & 0 & 0 & \Delta a \\ 0 & 1 & 0 & \Delta b \\ 0 & 0 & 1 & \Delta c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The advantage of this simple translation along the plate's optical axis is that it can be integrated into a fixed function rendering pipeline and can be carried out by graphics hardware. The drawback, however, is that this is only a rough approximation for refraction distortion, since every vertex is translated by the same amount Δ which is computed from a common incident angle (such as the angle between the viewing direction and the optical axis, for instance). The curvilinear characteristics of optical refraction and the individual incident angles of the viewing vectors to each scene vertex are not taken into account.

Programmable rendering pipelines allow per-vertex transformations directly on the graphics hardware. Thus the correct *curvilinear refraction transformation* can be implemented as *vertex shader*, rather than a rigid-body transformation expressed by a homogenous matrix.

The following CG shader fragment can be used to configure a programmable rendering pipeline with the refraction transformation:

```
…
//viewpoint = e[0..2]
//vertex = v[0..3]
//plane normal = n[0..2]
//plane thickness = t
//refraction index of material = r

float3 n,d;
float alpha_i, alpha_t, delta;

// compute viewing vector to vertex
d=e-v;
d=normalize(d);
n = normalize(n);

// compute angle between normal and
viewing vector
alpha_i = acos(dot(d,n));

// compute delta
alpha_t = asin(sin(alpha_i)/r);
delta = t*(1-
(tan(alpha_t))/tan(alpha_i));

//compute refracted vertex
v.xyz = v.xyz+n*delta;
…
```

No matter if refraction is implemented as rigid-body transformation or as per-vertex transformation, the sequence of the transformations carried out by the rendering pipeline is important for spatial optical see-through displays.

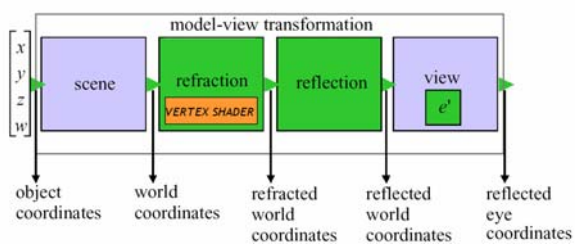


Figure 3.5: *The integration of reflection transformations into the rendering pipeline.*

Figure 3.5 illustrates the extension of figure 3.3. Refraction transformation has to be carried out after the scene transformation (either a rigid-body or as per-vertex transformation). Vertices are transformed from world coordinates to *refracted world coordinates* first. If the optical combiner is a mirror beam combiner, the refracted vertices are then reflected. If the optical combiner is a transparent screen, reflection transformation is not applied. Finally, the vertices are mapped into reflected eye coordinates (either with the reflected or with the un-reflected viewpoint – depending on the optical combiner), projected, converted into window coordinates, rastered and displayed.

The pseudo code below summarizes the rendering steps for spatial optical see-through displays that apply one planar screen and one planar beam combiner. Only one user is supported.

```
for left and right viewpoints i
     initialize transformation pipeline
     and polygon order
     compute reflected viewpoint e_i' = R e_i
     compute refraction offset Δ_i for i
     set transformation pipeline: RMF_i V_i P
     reverse polygon order
     render scene from e_i'
endfor
```

First, the polygon order and the transformation pipeline have to be set to an initial state. Then the reflected viewpoint and the view-dependent refraction offset (Δ_i) are computed. The transformation pipeline is then set to the corresponding concatenation of transformation matrices: reflection transformation (R), model transformation (M), refraction transformation (F_i, with e_i), view transformation (V_i, with e_i') and the projection transformation (P). Finally, the polygon order is reversed and the scene is rendered. Note that we assume that the screen coordinate system is equivalent to the world coordinate system. Note also that R might not be static but has to be re-computed continuously (e.g., if moving components have to be supported – see section 3.5).

3.4. Screen Transformation and Curved Screens

If the coordinate system of the secondary screen does not match with the world-coordinate system, an additional *screen transformation* has to be added to the rendering pipeline. It expresses the screen's transformation within the world coordinate system and is usually a composition of multiple translation and rotation transformations. Similar to the reflection transformation, not only the reflected geometry but also the reflected viewpoint has to be mapped into the *screen coordinate system*.

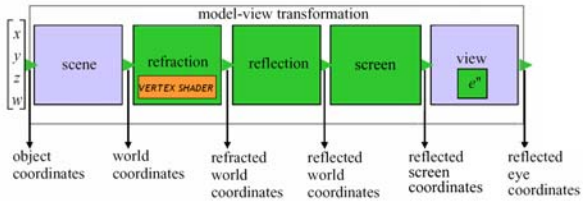


Figure 3.6: *The integration of screen transformations into the rendering pipeline.*

Important is once again the sequence in which all transformations are carried out. The screen transformation is applied after the reflection transformation and before the view transformation – mapping reflected world coordinates into *reflect screen coordinates*. The view transformation has to be performed with the reflected viewpoint in screen coordinates e'', that can be computed by simply applying the screen transformation to the reflected viewpoint e'.

To give an example, consider a 30cm x 40cm screen that is not aligned with the world coordinate system. In particular assume it is rotated by 180 degrees around the z-axis and translated in such a way that the screen's -y edge is aligned with the edge of a mirror beam combiner, located 20cm away from the world origin. In addition, the screen is tilted by 30 degrees about the edge that faces the mirror to achieve a higher contrast and a better position of the reflected image.

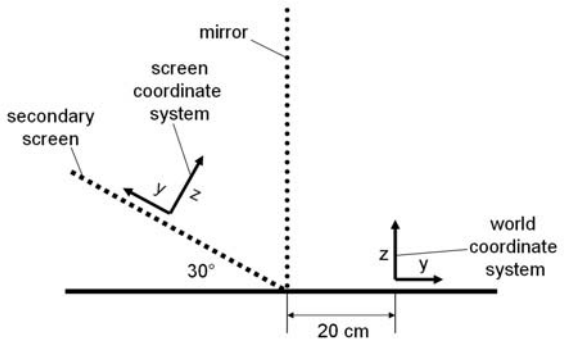


Figure 3.7: *Example of a screen transformation.*

The corresponding OpenGL screen transformation matrix can be calculated as follows:

```
…
glRotate(180,0,0,1);       //rotate
around z-axis by 180 deg
glTranlatef(0,15,0);       //translate
screen edge to origin
glRotatef(30,1,0,0);       //rotate
around screen edge by 30 deg
glTranslatef(0,20,0);      //translate
screen edge to mirror edge
…
```

Remember that the OpenGL notation has to be read from bottom-up (but it has to be implemented this way). To understand the transformation, it is helpful to think that the misaligned screen coordinate system has to be transformed back into the world coordinate system – since this corresponds to the transformation that needs to be applied to the rendered geometry: Reading the OpenGL fraction bottom-up, the screen is first translated by 20cm along the positive y-axis of the world coordinate system – aligning the adjacent edge with the world's origin. Then it is rotated around this edge by 30 degrees – fitting it to the x/y plane of the world coordinate system. Then the screen's origin position is matched with the world's origin position by translating the screen further up the positive y-axis (by exactly 15cm – half of the screen height). Finally, the screen rotated by 180 degrees to fully align both coordinate systems. A read-back from OpenGL's model-view stack (e.g., with glGetFloatv(GL_MODELVIEW_MATRIX,S);) allows to obtain this matrix and multiply it to the reflected viewpoint before the view transformation is added.

Projective texture-mapping [Seg92] can be applied in combination with *two-pass rendering* to support single or multiple front/back projections onto a multi-plane or curved display surface.

Projective textures utilize a *perspective texture matrix* to map projection-surface vertices into texture coordinates of those pixels that project onto these vertices. A first rendering pass generates an image of the virtual scene that will look perspectively correct to the user. During the second pass, this image is projected out from the user's current point of view onto a registered virtual model of the display surface -using projective texture-mapping. Finally, the textured model is rendered from the projector's point of view and the result is beamed onto the real display surface. If multiple projectors are used, the second pass have to be repeated for each projector individually. The generated images have to be geometrically aligned and color and edge blended appropriately to realize a seamless transition between them.

To support planar mirror beam combiners that require affine model and view transformations in combination with curved secondary screens, the above outlined method can be slightly modified: Instead of rendering the original scene from the observer's actual viewpoint, the reflected scene has to be rendered from the reflected viewpoint. The reflection transformations that are applied to the scene and the viewpoint depend on the optics. All other rendering passes remain unchanged. Note that if multiple planar display surfaces are used, and if one projector is assigned to one projection plane, projective textures and two-pass rendering are unnessecary. Instead, regular multi-pipeline rendering can be applied (as it is done for surround-screen projection systems, such as CAVEs or multi-sided workbenches).

3.5. Moving Components

Some spatial see-through displays do not require a static constellation of screens and optical combiners. They rather allow moving them freely within the world coordinate system during run time. With respect to rendering, it makes no difference whether the optical combiner and the screen is fixed or movable. The only modification to the transformations that are described above is that the plane parameters of the optical combiner (which influences reflection and refraction transformations), and/or the parameters of the screen transformation change continuously. This means that the rendering pipeline has to be updated every time a component has been moved – every frame in the worst case.

3.6. Multi-Plane Beam Combiners

More that one planar beam combiner can be utilized to build displays that provide views on an augmented environment from multiple –very different– angles, or to support multiple users simultaneously. This applies for mirror beam combiners as well as for transparent screens. If multiple planar beam combiners need to be supported by an optical see-through display simultaneously, the rendering pipeline that drives this display can be configured from the basic elements that have been discussed above. Convex mirror assemblies unequivocally tessellate the surrounding space into mirror individual reflection zones which –for the same point of view– do not intersect or overlap and consequently provide a definite one-to-one mapping between screen space and reflection space. The constellation of basic transformations for these displays depends on the physical constellation of optical combiners (e.g., mirror beam combiners or transparent screens) and image sources (e.g., video projectors or secondary screens). Several scenarios are discussed below.

3.6.1. Single Viewer

For the following explanation, we want to assume a *multi-plane beam combiner* that provides different perspectives onto the same scene to a single observer. We also want to assume that mirror beam combiners are applied as optical combiners, and a single secondary screen is used to display the rendered images. For example, four half-silvered mirrors can be assembled in form of an up-side-down pyramid frustum that reflects a large, horizontal projection screen. The real environment that needs to be augmented would be located inside the pyramid frustum.

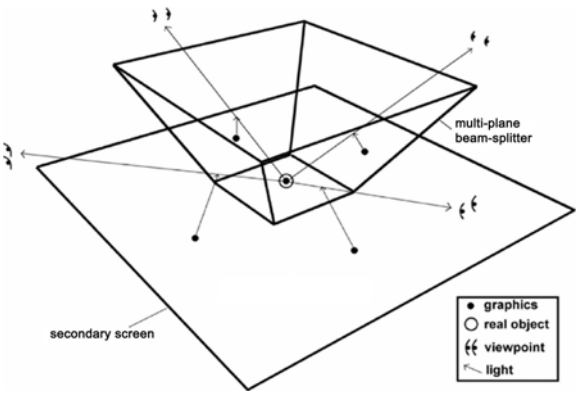


Figure 3.8: *Example for mulit-plane beam combiner constellation: four mirror beam combiners and one secondary screen that support observing the augmented real environment from 360 degrees.*

The step from single-plane to multi-plane optical see-through displays is similar to the step from single-plane projection displays (like walls or workbenches) to multi-plane projection displays (like caves or two-sided workbenches).

As outlined below, the rendering pipeline is split into several (four in our example) *sub-pipelines*. Each sub-pipeline is responsible for rendering a single image onto the secondary screen. Since only one screen is used in this example, four separate images have to be rendered into the same frame buffer that is displayed on the screen on each frame.

The images differ in their mirror-individual transformations (reflection and –optionally- refraction). The scene and the view transformation is the same for each sub-pipeline – except that individual reflections of the same viewpoint have to be applied together with the view transformation.

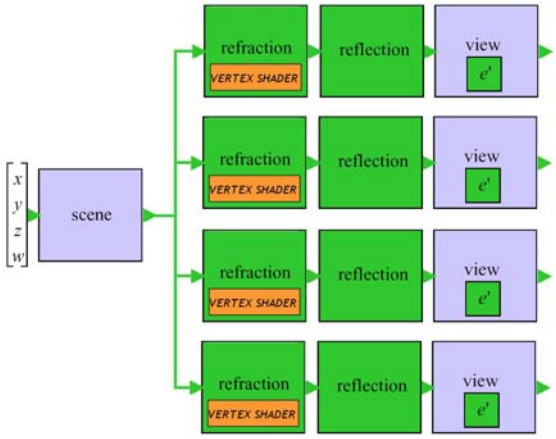


Figure 3.9: *Dependent rendering pipelines that support a multi-plane beam combiner in combination with a single screen and a single observer.*

Note that the number of sub-pipelines is doubled for stereoscopic viewing. In this case, the left and right stereo images may be rendered into two different frame buffers – the back-left and the back-right buffer if the graphics hardware provides a quad-buffer.

If the display is calibrated precisely enough (i.e., the parameters of the pipeline –such as mirror-plane

parameters, etc.– have been determined correctly), the different images merge into a single, consistent reflection from all points of view. The perpendicular mirror constellation that has been chosen in our example ensures that the different images never overlap in the frame buffer. Only two images and the corresponding two mirrors are visible from a single point of view at a time. The reflections of these images result in a single graphical scene.

Figure 3.10: *To images are reflected by two mirrors and merge into a single, consistent graphical scene.*

The sub-pipelines can be carried out sequentially on the same computer (i.e., by rendering each image into the frame buffer before swapping it). This divides the frame rate by a factor that equals the number of sub-pipelines.

Alternatively, hardware *image composition technology* (such as the Lightning-2 device [Stol01]) can be applied to distribute the sub-pipelines to multiple computers and merge the resulting images within a single frame buffer before the outcome is displayed on the screen. In this case multiple rendering nodes and the displays can be connected to a specific display subsystem. The subsystem allows the image data generated by the connected nodes to be mapped to any location of the connected displays without losing much time for transmission and composition.

To support configurations that apply multiple planar beam combiners and a single screen the following algorithm can be applied for a single user:

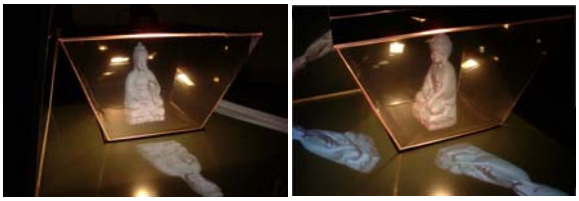
```
for left and right viewpoints i
   for each to i front-facing beam
   combiner j
      initialize transformation pipeline
      and polygon order
      compute reflected viewpoint
      e_i' = R_j  e_i
      compute refraction offset Δ_ij for
      i and j
      set transformation pipeline:
      R_j M F_ij V_i P
      reverse polygon order
      render scene from e_i'
   endfor
endfor
```

The scene has to be rendered for each viewpoint and each beam combiner. If the configuration of the optics unequivocally tessellates the surrounding space into mirror individual reflection zones which do not intersect or overlap, a single object that is displayed within the screen space appears exactly once within the reflection space.

3.6.2. Multiple Viewers

To support *multiple viewers*, each observer can be assigned to an individual beam combiner (and corresponding screen portion and rendering sub-pipeline).

Instead of using the same viewpoint for the view transformation of each sub-pipeline, different viewpoints of the assigned observe have to be applied. This means that the generated images are completely independent of each other and will no longer merge into a single, consistent three-dimensional scene. They rather represent the individual views on the augmented scene of each observer.



(a) (b)

Figure 3.11: *Individual views on the same scene generated for two different observers.*

If we use the above display example, the same scene is transformed and rendered for each observer individually, before all images are displayed on the same screen at the same time. In this case, each observer is restricted to limited viewing area that allows observing the scene through the assigned beam combiner only.

The following algorithm will support such kind of configurations:

```
for left and right viewpoints i of all
viewers
     initialize transformation pipeline
     and polygon order
     compute reflected viewpoint e_i' = R_i e_i
     compute refraction offset Δ_i for i
     set transformation pipeline: R_i M F_i V_i P
     reverse polygon order
     render scene from e_i'
endfor
```

Note that each user is restricted to the viewing zone that is given by the assigned beam combiner.

3.6.3. Multiple Screens

Using a single screen for multiple beam combiners has the disadvantage that each sub-pipeline renders its image into a portion of the common frame buffer. The resolution of a single image is only a fraction of the frame buffer's total resolution. To overcome this problem, an individual screen can be applied in combination with each beam combiner, sub-pipeline and –optionally– with each viewer.

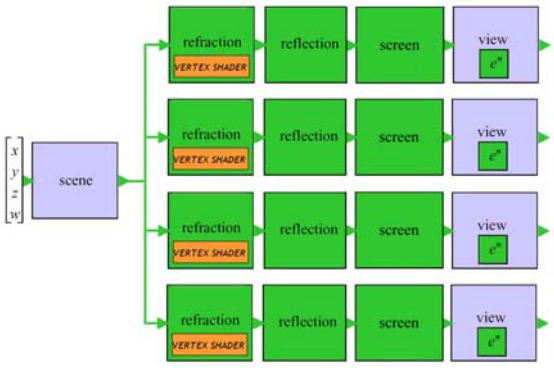


Figure 3.12: *Dependent rendering pipelines that support a multi-plane beam combiner in combination with multiple screens and a single or multiple observer(s).*

This implies that each image is rendered in full resolution into separate frame buffers that are displayed on different screens. Each sub-pipeline has to be extended by a screen transformation that copes with the screens' relative position and orientation within the global world coordinate system. Note, that single and multiple viewer scenarios are supported by this frame work – depending on the viewpoints that are passed to the view transformations.

- In general such a scenario can only be realized with *multi-channel rendering pipelines* that offer multiple frame buffers and the possibility to connect to more than one screen. One possibility is to use a large *graphics workstation* that provides these features. An alternative option that is becoming more and more popular is to apply multiple networked personal computers – so-called *PC clusters*. Nowadays, PC clusters are much more efficient than graphics workstations, both – from a performance and from a financial point of view.

- If PC clusters are used, it is important that the entire rendering process is synchronized correctly. The generated images have to be displayed at exactly the same time. This means that not only scene information have to be updated on all PCs continuously and at an acceptable speed, but also the signal that causes swapping the rendered images from back buffers to front buffers might have to be distributed in addition. This swapping synchronization is called *GenLocking* and is an essential feature for synchronizing multiple active stereoscopic screens. While some newer graphics cards can be connected to distribute the GenLock signal directly, older graphics cards do not provide this feature. However, open source software solutions (called *SoftGenLocking*) exist [All03] that offer an acceptable workaround. The scene information is usually distributed over the network, using distributed scene graph frameworks or other distributed graphics concepts. This ensures that the same scene state is displayed on each screen at a time. This scene synchronization is referred to as *frame-locking*.

Display configurations that assign an individual screen and beam combiner to each user benefit from a higher resolution those configurations that split the screen (and its resolution) into different portions. To support such configurations, an additional screen transformation (S) has to be introduces (see section 3.4). This matrix represents a concatenation of several translations, rotations and scaling transformations, that map coordinates from the world coordinate system into an individual coordinate system of the corresponding screen. One can also think of the inverse transformation of the screen itself within the world coordinate system.

The following algorithm can be applied in combination with such configurations:

```
for left and right viewpoints i of all
viewers
     initialize transformation pipeline
     and polygon order
     compute reflected viewpoint e_i' = R_i e_i
     compute refraction offset Δ_i for i
     set transformation pipeline:
     R_i M F_i V_i S_i P
     reverse polygon order
     render scene from e_i'
endfor
```

Note that by keeping track of individual model transformations (M) and by rendering individual scenes for each user, the display configuration practically acts as multiple user-individual displays. Otherwise, multiple users share and interact with the same scene. To use individual model transformations but to render the same scene for each user has been proven to be an effective way of precisely registering virtual objects to real ones – cancelling out slight physical misalignments of the screens and mirrors (whose measured/calculated parameters that are stored in R_i and S_i reflect these errors) within each M. Thereby, the object transformation that is used to align real and virtual objects is stored in each M. This also applies for the algorithm that is presented in section 3.6.2 – although only one screen is used.

3.6.4. Individual Scenes

For some multi-user scenarios, a common scene is not required. Rather than that, each user observers and interacts with his or her own scene representation on the same display. The actions that are carried out by a user do not effect the scene state of the other users.

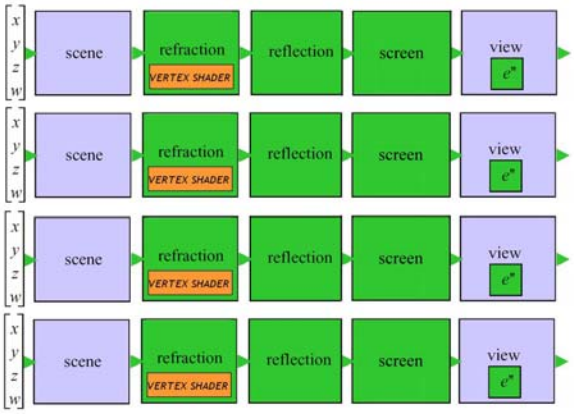


Figure 3.13: *Independent rendering pipelines that support individual scenes.*

In this case, the scene transformation becomes individual to each user and is carried out within each sub-pipeline. This makes all sub-pipelines completely independent from each other and allows distributing them (together with copies of the scene description) on separate rendering notes without the need of any synchronization mechanism. Now every user is assigned to completely independent components that are integrated into a single spatial optical see-through display.

3.7. Curved Mirror Beam-Splitters

Optical see-through displays can be assembled from many planar optical elements (mirror beam combiners or transparent screens). Every element requires its own rendering sub-pipeline. However, if many small optical elements are used to approximate curved surfaces, the above rendering concept becomes inefficient quickly.

Compared to calibrating multiple optical elements, the calibration effort for a single curved element can be reduced and optical aberrations that result from miscalibration can be decreased by several orders of magnitude. In addition, a curved optics can provide an *edge-free view* onto the augmented scene.

Like for curved projection screens the rendering for spatial optical see-through displays that apply *curved optics* (mirror beam combiners or transparent screens) has to be *warped* before being displayed. This transformation is view-dependent and curvilinear rather than a rigid-body transformation, and requires a *per-vertex viewpoint and model transformation*. A single accumulation of simple

homogenous transformation matrices (as described for planar optics) cannot express such a complex deformation for the entire scene. Consequently, fixed function rendering pipelines cannot be fully used for rendering graphics on curved optical see-through displays.

(a) (b)

Figure 3.14: *A warped and projected image reflected in a curved (conical) mirror beam combiner. A virtual cartridge is being placed in a physical printer.*

Note that since convex mirrors map a larger portion of the screen space into a smaller portion within the image space inside the optics, a high density of pixels can be compressed into a small reflection (cf. figure 3.14). Consequently, the *spatial resolution* within the reflection is regional higher than the spatial resolution of the display device!

Warping the scene geometry itself requires highly tessellated graphical models to approximate the *curvilinear optical transformation* well enough. Instead *multi-pass rendering* is frequently being applied. Rather that warping geometry, images of the geometry are warped and displayed. This causes the same effect, but the image transformation does not depend on the scene geometry (e.g., its type, its complexity, its shading, etc.).

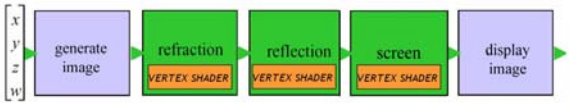
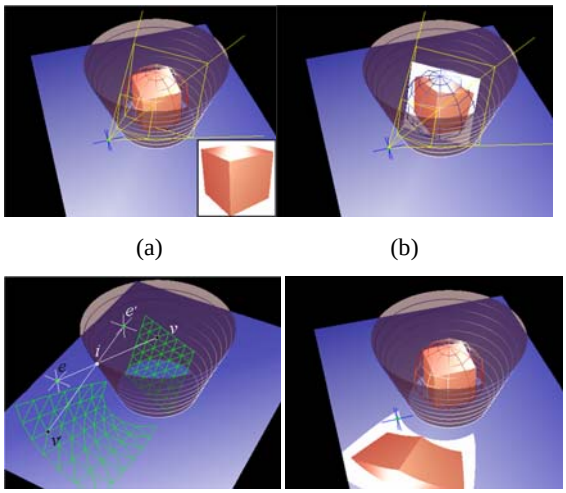


Figure 3.15: *Two-pass image warping pipeline for neutralizing curvilinear optical transformations.*

As outlined above, the *first rendering pass* generates an image of the graphical overlay – perspectively correct from the viewpoint of the observer. This image contains all visual information that is expected to appear within the real environment (e.g., shading, shadows, occlusion effects, etc.), and can be rendered completely in hardware. It is geometrically approximated by a *two-dimensional tessellated grid* that is transformed into the current viewing frustum in such a way that it is positioned perpendicular to the optical axis.

The grid vertices and the image's texture coordinates are transformed (warped) with respect to the viewpoint, the optics and the image source in the following steps. These transformations are applied on a *per-vertex level* and neutralize the image deformations that are caused by the optics (such as reflection and refraction of mirror beam combiners and transparent screens) and the image source (such as lens distortion of video projectors, or projections onto curved screens). As indicated, per-vertex transformations can be implemented as *vertex shaders* on programmable rendering pipelines.



(a) (b)

(c) (d)

Figure 3.16: *Different steps of image warping process: image generation, transformation of image grid, texturing and displaying deformed image grid.*

Finally, the image that was generated during the first pass is mapped onto the warped grid using *texture mapping* and *bi- or tri-linear texture filtering*, and is displayed during the *second pass*. This process is repeated for multiple individual viewpoints (e.g., for stereoscopic viewing and for multi-user scenarios).

The following sections will describe all steps of the *two-pass image warping pipeline* in more detail.

3.7.1. Generating Images

The first rendering pass generates the un-deformed graphical overlay with all information that is expected to be visible within the real environment. Conventional graphics techniques that are provided by fixed function rendering pipelines are sufficient for this step.

To capture the entire virtual scene in the image, the *scene's bounding sphere* is used to define an *on-axis viewing frustum* from the viewpoint of the observer. The *apex* of the frustum is chosen in such a way that it encloses the bounding sphere exactly. This ensures that the complete scene is visible in the image and that it covers a large portion of the image. Finally the scene is rendered through this frustum into a texture memory block – rather than directly into the frame buffer. Older graphics cards do not allow drawing an image into the on-board texture memory directly. In this case the scene has to be rendered into the back frame buffer, and then copied into the texture memory of the graphics card. This step is referred to as *read-back*.

New graphics cards do allow rendering the image directly into an auxiliary on-board memory block (usually called *P-buffer*) that can be referenced during texture mapping. This step is referred to as *render-to-texture* and avoids the time-consuming operation of transferring the image information from the frame buffer into the texture memory.

The following OpenGL code fragment can be used to configure the correct on-axis frustum for generating the un-deformed image of the virtual scene:

```
…
//scene's center = p[0..2]
//scene's bounding sphere radius = r
//viewpoint = e[0..2], up-vector =
u[0..2]
//texture height, texture width = th, tw

float l, d;
float left, right, bottom, top, near,
far;

//compute parameters of on-axis frustum
l=sqrt((p[0]-e[0])*(p[0]-e[0])+
       (p[1]-e[1])*(p[1]-e[1])+
       (p[2]-e[2])*(p[2]-e[2]));
d=r*(l-r)/l;
left=-d; right=d; bottom=-d; top=d;
near=l-r; far=l+r;

//configure rendering pipeline
glViewport(0,0,tw,th);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glFrustum(left,right,bottom,top,near,far)
;
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(e[0],e[1],e[2],p[0],p[1],p[2],u
[0],u[1],[u2]);

//draw scene with scene transformation
into texture memory
…
```

The big advantage of this image-based method is that it is completely independent on the scene's content and the way this content is rendered. Instead of using a geometric renderer, other techniques (such as image-based and non-photo-realistic rendering, interactive ray-tracing, volume rendering, or point-based rendering, etc.) can be employed to generate the image.

The image that has been generated during the first rendering pass has to be transformed in such as way that is perceived undistorted while observing it through or with the display's combiner optics. These transformations are described below in detail. To support these image deformations, a geometric representation of the image plane is generated initially. This *image geometry* consists of a *tessellated grid* (e.g., represented by an *indexed triangle mesh*) which is transformed into the current viewing frustum in such a way that, if the image is mapped onto the grid each line of sight intersects its corresponding pixel. Thus, the image grid is perpendicular to the optical axis and centered with the scene geometry. The following deformation steps transform the positions and the texture coordinates of every grid vertex individually.

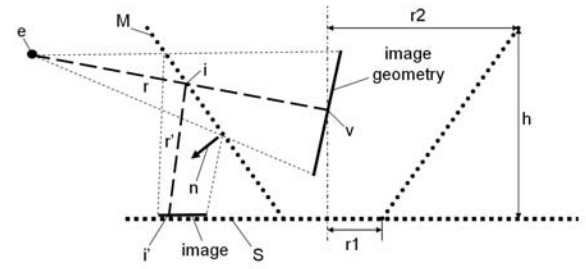
3.7.2. Reflection on Convex Mirrors

As discussed in section 3.6, convex multi-plane mirror assemblies unequivocally tessellate the surrounding space into mirror individual reflection zones which –for the same point of view– do not intersect or overlap. Consequently they provide a definite one-to-one mapping between screen space and reflection space. This is also true for curved convex mirrors. Curved convex mirrors cannot provide true stigmatism between all object-image pairs, but rather a close approximation which introduces small optical aberrations. Since due to the limited resolution of the eyes, human observers can usually not perceive these aberrations, we want to disregard them for the subsequent sections.

Images formed by convex mirrors appear to be reduced and deformed versions of the reflected objects. Hence, the image of the scene that results from the first rendering pass has to be stretched before displaying it on the secondary display. This results in the original (unscaled) image after the physical reflection.

Several curved mirror displays exist that generally don't pre-distort the graphics before they are displayed. Yet, some systems apply additional optics (such as lenses) to stretch or undistort the reflected image (e.g. [Mck99a, Mck99b]). But these devices constrain the observer to a single point of view or to very restricted viewing zones. However, if a view-dependent rendering is required to support freely moving observers, interactive rendering and real-time image warping techniques are needed which provide appropriate error metrics.

The reflection transformation for displays that apply convexly curved mirror beam combiners will be explained below based on an example of the conical mirror display illustrated in the figures above. Any other surface type can be used in combination with the described techniques.



(a)

(b) (c)

Figure 3.17: *Reflection transformation on conical mirror.*

Each grid point v of the image geometry has to be transformed with respect to the mirror surface M, the current viewpoint e, and the secondary screen S and texture each grid point with the image that we generated during the first rendering pass. For all grid vertices v, the intersection i of the geometric line of sight with the mirror surface has to be computed (that is, the ray r that's spanned by the eye e and the vertex v). Next, the normal vector n at the intersection i needs to be determined. The intersection point, together with the normal vector, gives the tangential plane at the intersection. Thus, they deliver the individual plane parameters for the per-vertex reflection. These information are then used to compute the reflection ray r' and the intersection i' of r' with the secondary screen surface S.

Having a geometric representation (e.g., a triangle mesh) to approximate the mirror's and the screen's surfaces, M and S, supports a flexible way of describing the dimensions of arbitrary shapes. However, the computational cost of the per-vertex transformations increases with a higher resolution surface geometry. A high resolution surface description, however, is required to neutralize the optical reflection distortion effectively. For triangle meshes, a fast ray-triangle intersection method (such as [Moe97]) is required that automatically delivers the barycentric coordinates of the intersection within a triangle. The barycentric coordinates can then be used to interpolate between the three vertex normals of a triangle to approximate the normal vector at the intersection.

A more efficient way of describing surface shapes for this purpose is to apply explicit functions in the form of $F(x,y,z)=0$. Explicit functions to describe mirror and screen surfaces ($M(x,y,z)=0$ and $S(x,y,z)=0$) can be used to calculate the intersections and the normal vectors (using its 1st order derivatives) with an unlimited resolution by solving simple equations. However, not all shapes can be expressed by explicit functions.

Obviously, we have the choice between a numerical and an analytical approach, for describing mirror and screen surfaces. If an analytical solution is given, it should be preferred over the numerical variant. Higher order curved surfaces, however, require the application of numerical approximations.

The explicit function for the conical mirror example and its 1st order derivatives are:

$$M(x,y,z)=\frac{x^2}{r1^2}+\frac{y^2}{r2^2}-\frac{z^2}{h^2}=0 \qquad \text{and}$$

$$n=\left[\frac{M}{\delta x},\frac{M}{\delta y},\frac{M}{\delta z}\right]=2\left[\frac{x}{r1^2},\frac{y}{r2^2},-\frac{z}{h^2}\right]$$

where $r1, r2$ are the cone's radii with its center located at the world-coordinate-system's origin, and h is the cone's height along the z-axis.

To intersect the ray r with the mirror surface, it has to be transformed from the world-coordinate-system into the coordinate-system that is used by the explicit function. Then it can be intersected easily with the surface by

solving a (in our example quadratic) equation created by inserting a parametric ray representation of $r(x,y,z)$ into the mirror equation $M(x,y,z)$.

Given the surface intersection i and the normal n at i, the specular reflection ray can be computed with:

$$r' = r - 2n(nr)$$

Finally, r' has to be transformed from the coordinate system of M into the coordinate system of S to compute the intersection i' with S by solving another equation created by inserting r' into S.

The following CG vertex shader fragment can be used to compute the reflection transformation for conical mirror surfaces (this example applies to the truncated conical mirror surface illustrated in figure 3.17):

```
…
// viewpoint = e[0..2]
// vertex = v[0..3]
// truncated cone parameters: lower
// radius = r1, upper radius = r2,

height = h

float3 i,i_,n,r,r_;
float  a,b,c,r,s,t,u,l1,l2,l3;

// compute cone parameters
a=r2;b=r2; c=(r2*h)/(r2-r1);

// transformation to cone-coordinat
system // (r=v-e)
v.z=v.z+(c-h);      e.z=e.z+(c-h);
//compute viewping direction (r)
r=v-e; r=normalize(r);

//compute cone intersection

s=(e.x*e.x)/(a*a)+(e.y*e.y)/(b*b)-
(e.z*e.z)/(c*c);

t=2.0*((e.x*r.x)/(a*a)+(e.y*r.y)/(b*b)-
(e.z*r.z)/(c*c));

u=(r.x*r.x)/(a*a)+(r.y*r.y)/(b*b)-
(r.z*r.z)/(c*c);

l1=(-t+sqrt(t*t-4.0*u*s))/(2.0*u);
l2=(-t-sqrt(t*t-4.0*u*s))/(2.0*u);
i=e+r*l2;

// back-transformation to world
// coordinate-system
i.z=i.z-(c-h);

// compute cone normal
n.x=i.x/(a*a);
n.y=i.y/(b*b);
n.z=i.z/(c*c);
n=normalize(n);

// compute incident ray (i-e)
r=i-e; r=normalize(r);

// compute reflector
r_=reflect(r,n);

// compute intersection with screen (x/y
// plane in this example)
l1=-dot(float3(0,0,1),i);

l2=dot(float3(0,0,1),r_);
l3=l1/l2;
i_=i+l3*r_;

// compute projected vertex
v.xyz=i_.xyz; v.w=1.0;
…
```

Note, that the same shader can be used in combination with any other explicit functions that describe the mirror and the screen surfaces. Only the surface-equation specific parts have to be replaced.

3.7.3. Refraction

A vertex transformation is applied to the rendered geometric model between scene and reflection transformation. To cope with the curvilinearity of refraction, vertex shaders are preferred over a rigid-body approximation if programmable function pipelines are available.

For the image-based method that is discussed above, a per-pixel transformation on the image plane would be more efficient than a per-vertex transformation of the image grid. Pixel shaders of programmable function pipelines, however, do not yet support the geometric transformation of pixels. But pixel transformations can be approximated with vertex shaders by computing new texture coordinates for each vertex that offsets the corresponding pixel positions on the image plane. The positions of pixels in between vertices (i.e., within patches of the image grid) are then approximated via linear interpolation of texture mapping. As for the reflection transformation, this introduces optical errors that can be high if the resolution of image grid is too low.

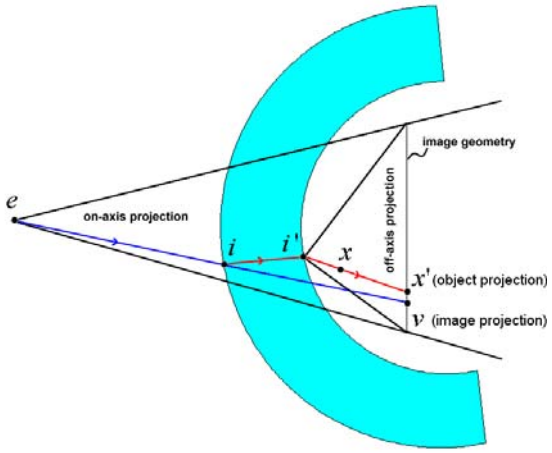


Figure 3.18: *Image-based in-/out-refraction transformation on curved lens.*

As illustrated above, two surface descriptions (M_o and M_i) are required to compute in- and out-refracted rays for such optical combiners since they represent curved, surface-parallel lenses. As for the reflection transformation, explicit functions in the form of $F(x,y,z)=0$ are again most efficient for computing the required intersections i, i' and the corresponding parameters of the tangent planes.

Similar to the reflection case, each ray from the viewpoint e to all grid vertices v have to be processed. Each ray has to be transformed from the world-coordinate-system into the coordinate-system that is used by the explicit function, and then intersected with the outer in-refracting surface M_i. The computed intersection i and its tangential plane parameters are used to determine the in-refractor, which is intersected again with the inner out-refracting surface M_o. The second intersection i' and its tangential plane parameters allow computing the out-refractor. Any point on this ray (x) can be projected onto the image plane from i'. This results in position x' in projected and normalized device coordinates (e.g., [-1,1]). After a conversion from device coordinates into normalized texture-coordinate (e.g.,[0,1]), the texture coordinates of the pixel that needs to appear at grid vertex v has been determined. To simulate refraction, these texture coordinates are simply assigned to v. Note, that x' can also be computed by intersecting the out-refractor directly with the image plane. This may requires a different conversion from world coordinate systems into normalized texture coordinates.

The composition of an appropriate texture matrix that computes new texture coordinates for the image vertex is summarized in the algorithm below:

compute texture normalization correction*:

$$S = scale(0.5,0.5,0.5), s = s \cdot translate(1,1,0)$$

compute off-axis projection transformation:

$$\phi = \frac{(i'_z - 1)}{i'_z},$$

$$left = -\phi(1 + i'_x),$$

$$right = \phi(1 - i'_x)$$

$$bottom = -\phi(1 + i'_y), \quad top = \phi(1 + i'_y)$$

$$near = i'_z - 1, \quad far = i'_z + 1$$

$$P = frustum(left, right, bottom, top, near, far)$$

compute view transformation:

$$V = lookat(i'_x, i'_y, i'_z, i'_x, i'_y, 0,0,1,0)$$

compute new texture coordinate x' for particular $x(v)$, including perspective division:

$$x' = S \cdot \frac{(P \cdot V \cdot x)}{w}$$

As illustrated in figure 3.18, an off-axis projection transformation is applied, where the center of projection is i'. Multiplying x by the resulting texture matrix and performing the perspective division projects x to the correct location within the normalized texture space of the image. Finally, the resulting texture coordinate x' has to be assigned to v'.

Note that if the image size is smaller than the size of the allocated texture memory, this difference has to be considered for the normalization correction. In this case, the image's texture coordinates are not bound by the value range of [0,1].

As mentioned above, the matrix operations can be replaced by an explicit ray-casting.

*This applies for OpenGL-like definitions of the texture and normalized device coordinates.

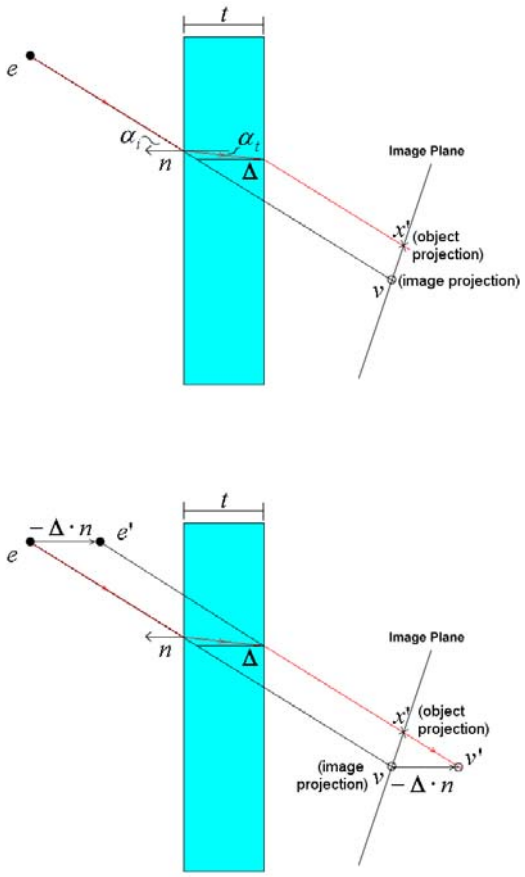


Figure 3.19: *Image-based in-/out-refraction transformation on planar lens.*

For plane parallel lenses, the in-refracting and out-refracting surfaces are the same – but offset by the thickness t of the lens. Explicit in- and out-refractors do not have to be computed in this case. Rather than that, the offset Δ can be computed for every geometric line of sigh $(v - e)$ as described in section 3.3.2. The new projection on the image plane can be computed by translating the viewpoint e and the corresponding grid vertex v by the amount Δ along the normal of the lens. The new texture coordinates can be computed by projecting v' onto the image plane from e' and converting the result into the normalized texture space.

Nevertheless, both refraction methods face the following problems for outer areas on the image:

- Given a geometric line of sight to an outer grid vertex, its corresponding optical line of sight does not intersect the image. Thus, a grid vertex exists but its new texture coordinate cannot be computed. This results in vertices with no, or wrong texture information;
- Given an optical line of sight to an outer pixel on the image, its corresponding geometric line of sight does not intersect the image. Thus, a texture coordinate can be found but an assignable grid vertex does not exist. Consequently, the portion surrounding this pixel cannot be transformed. This results in image portions that aren't mapped onto the image geometry.

A simple solution to address these problems does not avoid them, but ensures that they do not occur for image portions which contain visible information: As described in section 3.7.1 the image size depends on the radius of the scene's bounding sphere. Its radius can simply be increase by a constant amount before carrying out the first rendering pass. An enlarged image does not affect the image content, but subjoins additional outer image space that does not contain any visible information (i.e., just black pixels). In this way, we ensure that the above mentioned problems emerge only at the new (black) regions. Yet, these regions will not be visible in the optical combiner.

Note that in contrast to the image-based reflection transformation (section 3.7.2) which transforms grid vertices, the refracted image transform re-maps texture coordinates. However, all image transformations have to be applied before the final image is displayed during the second rendering pass.

3.7.4. Screen Transformation and Non-Planar Screens

In cases where the screen coordinate system is not equivalent to the world coordinate system in which beam combiners, user(s) and scene(s) are defined, an additional screen transformation has to be applied to project the reflected/refracted image-geometry vertices to their correct places. This can be achieved with additional affine transformation operations that are applied to the image-geometry itself, as described in section 3.4, or by intersecting the traced rays with the geometry of the transformed screen plane (e.g., if explicit ray-casting is used inside a vertex shader).

If the secondary screen is not planar, an explicit ray-casting approach still leads to a correct result. The traced

rays simply have to be intersected with a geometric representation of the curved screen.

Note that an approach that applies projective texture-mapping [Seg92] for planar mirror beam-spitters fails for a combination of curved mirror optics and curved screens. Although the first rendering pass is similar to the first pass that generates the image (see section 3.7.1), the second pass cannot be applied, because multiple centers of projection exist (one for each transformed image vertex).

3.7.5. Displaying Images

During the second rendering pass, the transformed image geometry is finally displayed within the screen space, mapping the outcome of the first rendering pass as texture onto its surface.

Note, that if the reflection/refraction transformations of the previous steps deliver device coordinates, but the secondary screen and the mirror optics are defined within the world-coordinate system, a second projection transformation (such as glFrustum) and the corresponding perspective divisions and viewpoint transformation (such as gluLookAt) aren't required. If a plane secondary screen is used, a simple scale transformation suffices to normalize the device coordinates – for example, glScale(1/device_width/2),1/device_height/2,1). A subsequent view-port transformation finally upscales them into the window-coordinate system – for example, glViewport(0,0,window_ width, window_height).

Time-consuming rendering operations that aren't required to display the 2D image (such as illumination computations, back-face culling, depth buffering, and so on) should be disabled to increase the rendering performance.

In this case, the polygon order doesn't need to be reversed before rendering, as we noted previously.

Obviously, one can choose between numerical and analytical approaches to represent curved mirror surfaces. Simple shapes can be expressed as parametric functions, but higher order curved mirrors require numerical approximations. In addition, the grid resolution that's required for the image geometry also depends on the mirror's shape. Pixels between the triangles of the deformed image mesh are linearly approximated during rasterization (that is, after the second rendering pass). Thus, some image portions stretch the texture while others compress it. This results in different regional image resolutions. However, because of the symmetry of simple mirror setups (such as cones and cylinders), a regular grid resolution and a uniform image resolution achieve acceptable image quality. In section 3.7.8 a selective refinement method is described that generates a non-uniform image geometry to minimize the displacement error of the image portions, the complexity of the image geometry and consequently the number of vertex transformations and triangles to be rendered.

Since a primitive-based (or fragment-based) antialiasing doesn't apply in deformed texture cases, bilinear or trilinear texture filters can be used instead. As with antialiasing, texture filtering is usually supported by the graphics hardware.

Note that the image's background and the empty area on the secondary screen must be rendered in black, because black doesn't emit light and therefore won't be reflected into the reflection space.

3.7.6. Multiple Viewers

To support multi-user applications, the viewer-individual images must be composed and the black background must be *color-blended* appropriately. Since for convex beam combiners the images are stretched within the screen space to appear correctly within the reflection space (cf. figure 3.20), multiple images for different observers might intersect.

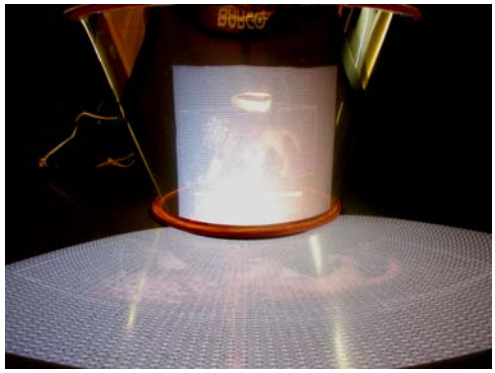


Figure 3.20: *Stretched image geometry projected on secondary screen and reflected in conical mirror beam combiner.*

In these cases, individual observers can perceive the (perspectively wrong) images of other users in addition to their own images. The amount of intersection depends on the size of the graphical scene, the positions of the observers and the parameters of the mirror optics and the

secondary screen. The larger the graphical scene, for instance, the more space on the secondary screen is required and conflicting situations become more likely. Such conflicts also arise for multi-plane beam combiner configurations if multiple users would be allowed to move freely around the display (as it has been described for a single user in section 3.6.1). Although this is technically possible, it has not been discussed in section 3.6.2.

3.7.7. Concave Beam-Splitters

Concave mirrors can generate both -real images and virtual images. Light rays which are reflected off convex mirror assemblies do not intersect. In contrast to this, light rays that are reflected off a *parabolic concave mirror* do intersect exactly within its *focal point*. For concave mirrors that deviate from a parabolic shape (e.g., *spherical mirrors*) the light rays do not intersect within a single point, but bundle within an *area of rejuvenation*.

The rendering techniques for curved mirror beam combiners explained above, are described on the basis of convex mirrors. However, they can be applied for concave mirrors without or with only minor modifications to the algorithms.

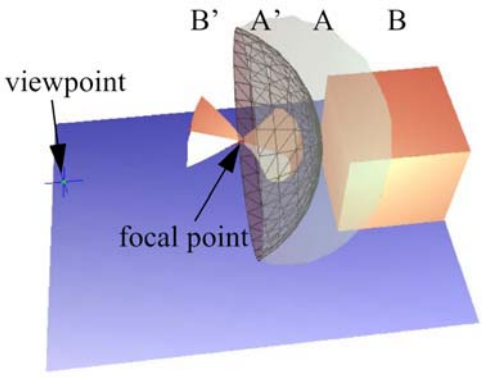


Figure 3.21: *Reflected geometry at a concave mirror.*

Figure 3.21 illustrates how, given the viewpoint and the location of the mirror, the geometry of a cube is transformed by the algorithm described in section 3.7.2 (without projection onto the screen plane). Note, that the vertices of a highly tessellated surface approximation of the cube are reflected, instead of the vertices of the image geometry.

The portion of the geometry that is located within area A is mapped to area A' (behind the mirror's focal point -as seen from the viewpoint). This mapping has a similar behavior as for convex mirrors. The portion of the geometry that is located within area B, is mapped to area B' (in front of the mirror's focal point -as seen from the viewpoint). In this case, the mapped geometry is flipped and the polygon order is changed. The geometry that is located on the intersecting surface of A and B is mapped to the focal point.

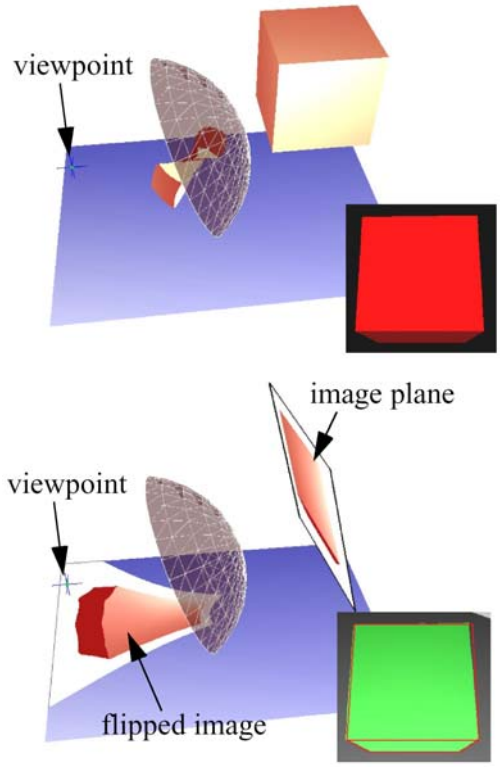


Figure 3.22: *Reflecting scene geometry (left) and reflecting image geometry (right) at a concave mirror.*

Figure 3.22 shows that concave mirrors can be treated just like convex mirrors. The left image indicates the reflection of the scene geometry itself (again, without projection onto the screen plane). The transformed geometry and the mirror model have been exported to a ray-tracer and the image that is seen from the viewpoint has been raytraced. The result is outlined at the lower right of the left image (red cube).

It can be seen, that the transformed geometry appears untransformed as reflection in the mirror. Note that due to

the different orders of polygons that are located in front of and behind the focal point, polygon order related options offered by the applied rendering package should be disabled. Otherwise, additional actions have to be taken to determine the location of each transformed polygon and its order.

The same experiment has been carried out reflecting the image geometry (including the projection onto the screen plane). The image geometry was transformed and the image that has been generated during the first rendering pass was mapped onto the transformed image grid. The right part of figure 3.22 illustrates that the projected image is a flipped version of the original image. If ray-traced from the given viewpoint, this deformed image appears as correct reflection in the mirror (see lower right of the right image). Note that the contours of the ray-traced result have been highlighted, since the planar projection of the deformed image does not provide sufficient normal or depth information to generate correct shading effects with the ray-tracer. Note also, that reflection the scene geometry and projecting the result onto the screen plane yields the same results as reflecting and projecting the image geometry.

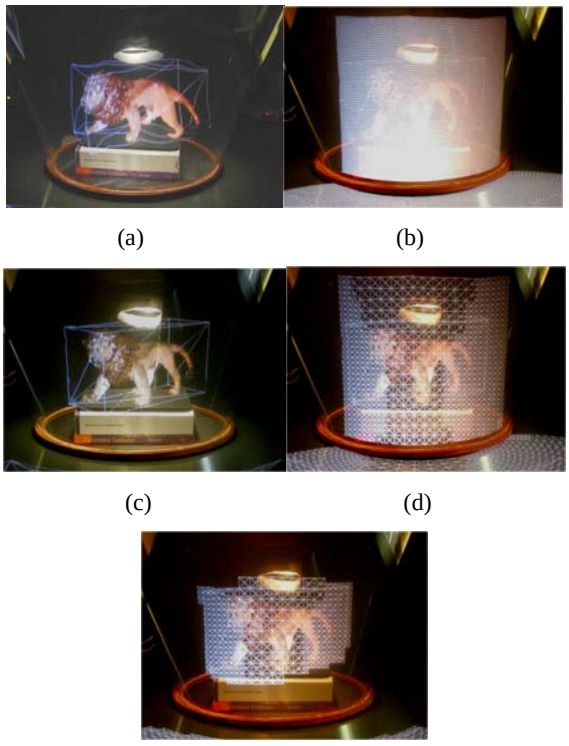
Mirrors with a *mixed convexity* (i.e., simultaneously convex and concave mirrors) can cause multiple images of the same object, or they can reflect multiple objects to the same location within the image space. In such cases, the transformations from screen space to reflection space and vice versa are not definite and are represented by possible many-to-many mappings. Such types of mirrors should be decomposed into convex and concave parts (as done by Ofek [Ofe98, Ofe99]) to ensure a correct functioning of our algorithms. Although for many surfaces this can be done fully atomically [Spa92], spatial optical see-through configurations do not make use of mirrors with mixed convexities. This is because one of the initial aim of such displays -namely to unequivocally overlay real environments with computer graphics in an AR manner- is physically not supported by such mirrors. Consequently, mirrors with mixed convexities are not considered for spatial augmented reality.

3.7.8. Non-Uniform Image Geometry

If image warping is applied for a *non-linear predistortion* (such as described above), the required grid resolution of the underlying image geometry depends on the *degree of curvilinearity* that is introduced by the display (e.g. caused by the properties of a mirror, a lens, or a projection surface).

Pixels within the triangles of the warped image mesh are linearly approximated during rasterization (i.e. after the second rendering pass). Thus, some image portions stretch the texture while others compress it. This results in different local image resolutions. If, on the one hand, the geometric resolution of an applied uniform grid is too coarse, texture artifacts are generated during rasterization (cf. figure 3.23a). This happens because a piecewise bi- or tri-linear interpolation within large triangles is only a crude approximation to neutralize the curvilinear optical transformations.

If on the other hand, the grid is oversampled to make the interpolation sections smaller (cf. figure 3.23b), interactive frame rates cannot be achieved. In addition to the speed of the first rendering pass, the performance of view-dependent multi-pass methods depends mainly on the complexity of the image geometry that influences geometric transformation times and on the image resolution that influences *texture transfer* and *filter times*.



(a) (b) (c) (d) (e)

Figure 3.23: *Selectively refined image geometry: (a) distorted image created with an undersampled uniform image grid, (b) oversampled uniform image grid, (c) undistored image, (d) selectively refined image grid, (e) grid portion that projects onto bounding container.*

This section describes a selective refinement method [Bim03] that generates image grids with appropriate local grid resolutions on the fly. This method avoids oversampling and the occurrence of artifacts within the final image.

The main difficulty for a selective refinement method that supports curved mirror beam combiners is that in contrast to simpler screens, a displacement error that defines the refinement criterion has to be computed within the image space of the mirror optics, rather than in the screen space of the display. For convexly curved mirror beam combiners, this requires fast and precise numerical methods. For displays that do not contain view-dependent optical components (e.g. curved screens), these computations are much simpler because analytical methods or look-up tables can be used to determine the error within the object space (e.g. the screen space).

Recent advances in level-of-detail (LOD) rendering take advantage of *temporal coherence* to *adaptively refine* geometry between subsequent frames. Especially, terrain-rendering algorithms locally enhance terrain models by considering viewing parameters.

Hoppe introduced *progressive meshes* [Hop96] and a later developed a view-dependent refinement algorithm for progressive meshes [Hop97, Hop98]. Given a complex triangle mesh, Hoppe first pre-generates a coarse representation called *base mesh* by applying a series of *edge collapse operations*. A sequence of pre-computed *vertex split operations* that are inverse to the corresponding edge collapse operations can then be applied to the base mesh's regions of interest to successively refine them.

The selection of the appropriate vertex split operations is based on his refinement criteria. Lindstrom [Lin96] describes a method that generates a view-dependent and continuous LOD of height fields dynamically in real-time, instead of precomputing a coarse base mesh and a sequence of refinement steps. He hierarchically subdivides a regular height field into a *quad-tree* of *discrete grid blocks* with individual LODs. Beginning with the highest LOD, Lindstrom locally applies a two-step surface simplification method: he first determines which discrete LOD is needed for a particular region by applying a coarse *block-based simplification*, and then performs a *fine-grained re-triangulation* of each LOD model in which vertices can be removed. To satisfy continuity among the different LODs, Lindstrom considers vertex dependencies at the block boundaries.

The main difference between both methods is that Lindstrom performs a dynamic simplification of highresolution height fields for domains in R^2 during rendering. Lindstrom's mesh definition provides an *implicit hierarchical LOD structure*. Hoppe applies refinement steps to low-resolution LODs of arbitrary meshes during rendering. His mesh definition is more general and does not require an implicit hierarchical LOD structure. Consequently, the refinement steps and the low resolution base mesh have to be precomputed. In addition, he applies *triangulated irregular networks* (TINs) for triangulation, rather than regular grids. Note that these two types of refinement methods may be representative for related techniques. Since the image geometry for spatial optical see-through displays can also be parameterized in R^2 and provides an implicit hierarchical LOD structure, multiple LODs or appropriate refinement steps do not need to be pre-computed but can be efficiently determined on the fly. This is similar to Lindstrom's approach. However, simplifying a high-resolution mesh instead of refining a low-resolution mesh would require to re-transform all grid vertices of the highest LOD after a change of the viewpoint occurred. This is very inefficient, since for the type of displays that we consider, viewpoint changes normally happen at each frame.

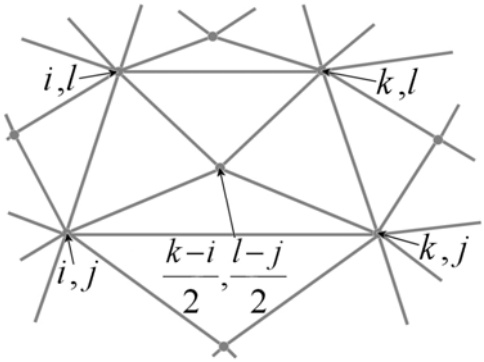
In contrast to the static geometry that is assumed in Lindstrom's and Hoppe's case the image geometry generated for spatial optical see-through displays is not constant but dynamically deforms with a moving viewpoint. Consequently, the geometry within all LODs dynamically changes as well.

Therefore, this section describes a method that dynamically deforms the image geometry within the required LODs while *selectively refining* the lowest-resolution base mesh during rendering. The method aims at minimizing the *displacement error* of the image portions, the complexity of the image geometry and consequently the number of vertex transformations and triangles to be rendered.

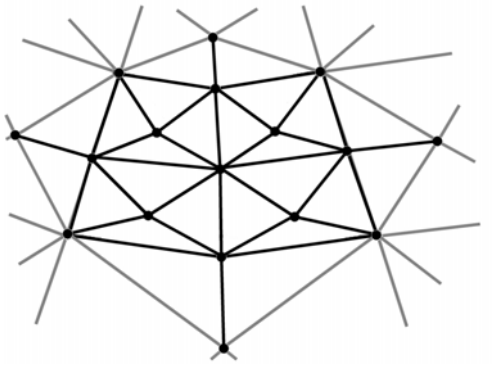
Note that this method cannot be implemented with the limited capabilities of vertex shaders, yet. Consequently it has to be executed on the CPU rather than on the GPU. This implies that rendering a high-resolution uniform image grid on a GPU might be faster that creating a selectively refined image grid on the CPU. Nevertheless, the LOD method is explained to impart knowledge about displacement errors that are generated by approximating a curvilinear optical transformation with discretized image grids.

3.7.8.1. Image Triangulation

Instead of transforming and rendering a uniform high-resolution mesh, one can start from the coarsest geometry representation and successively refine it locally until certain *refinement criteria* are satisfied. Due to the well-defined structure of the image grid, all possible *global or discrete LODs* can be computed at runtime – including the highest, which is the uniform high-resolution representation of the mesh itself.



(a)



(b)

Figure 3.24: *Triangulation of unrefined patch at LOD n (a), and triangulation of refined patch at LOD n+1 with resolution transitions (b).*

Figure 3.24 illustrates a quadtree-based image triangulation, which is similar to Lindstrom's triangulation method for height fields [Lin96]. While figure 3.24a shows an unrefined patch at LOD n, figure 3.24b shows the same patch at LOD $n+1$, with lower LOD neighbors. Given a highest LOD of m, an indexed $(2^m+1)\times(2^m+1)$ matrix structure can be chosen to store the grid vertices.

For illustration purposes the following types of patch vertices can be differentiated:

- *L-vertices* are vertices at the corners of a patch (e.g., at indices [*i,j*],[*i,l*],[*k,l*] and [*k,j*] in figure 3.24a);
- *X-vertices* are vertices at the center of a patch (e.g., at index [(*k-i*)/2,(*l-j*)/2] in figure 3.24a);
- *T-vertices* are vertices that split the patch edges after refinement (e.g., at indices [*i*,(*l-j*)/2], [*k*,(*l-j*)/2], [(*k-i*)/2,*l*], and [(*k-i*)/2,*j*] in figure 3.24b)

To refine a patch, it is divided into four sub-patches by computing the corresponding four T-vertices, as well as the four X-Vertices that lie inside the sub-patches. Note that the matrix structure is in this particular case equivalent to a quad-tree data structure. To ensure consistency during rasterization, the T-vertices have to be connected to their neighboring X-vertices wherever a LOD transition occurs (e.g., at all four neighbors of the refined patch, as shown in the example illustrated in figure 3.24b).

Due to the well-defined matrix structure that contains the image grid, the following conditions are given:

- A clear relationship between the X-vertices and T-vertices exists: X-vertices can never be T-vertices, and vice versa.
- Each patch has definite L-vertices, T-vertices and X-vertices, whose indices can always be computed.
- Each X-vertex can be explicitly assigned to a single patch at a specific LOD.
- Each T-Vertex can be explicitly assigned to exactly one or two adjacent patches at the same LOD.

The triangulation methods described above require *continuous* level-of-detail transitions [Mit92]. This implies that neighboring patches do not differ by more than one LOD.

3.7.8.2. Recursive Grid Refinement

The objective of this step is to generate an image grid that provides a sufficient local grid resolution (i.e. appropriate discrete LODs) to avoid artifacts within the rasterized texture that would result from undersampling, as well as oversampling.

The following pseudo code illustrates an approach to recursively refine a grid patch, which initially is equivalent to the lowest LOD (i.e. the patch at the lowest LOD is outlined by the L-vertices at the four corners of the image geometry):

```
RecursiveGridRefinement(i,j,k,l )
1: begin
2:   a=(k-i)/2 , b=(l-i)/2
3:    if
     GeneratePatch([i,j],[i,l],[k,l],[k,j
     ],[a,b])
4:     begin
5:
       TransformPatchVertices([i,j],[i,l]
       ,[k,l],[k,j],[a,b])
6:     P = P ∪ {[i, j, k, l]}
7:     if
       RefineFurther([i,j],[i,l],[k,l],[k
       ,j],[a,b])
8:      begin
9:       RecursiveGridRefinement(i,j,a,b)
10:      RecursiveGridRefinement(a,j,k,b)
11:      RecursiveGridRefinement(i,b,a,l)
12:      RecursiveGridRefinement(a,b,k,l)
13:      if  j < 2^m +1  TC[a,j]+=1
14:      if  k < 2^m +1  TC[k,b]+=1
15:      if  l > 1  TC[a,l]+=1
16:      if  i > 1  TC[i,b]+=1
17:     end
18:    end
19: end
```

The patch that has to be refined is stored at indices *i,j,k,l* within a matrix structure. Condition (ii) allows to locate the position of the patch-individual X-vertex at indices *a,b* (line 2). First, it is evaluated whether or not a patch has to be generated at all (line 3). The conditions that are implemented within the *GeneratePatch* function will be discussed below. The four L-vertices and the X-vertex are transformed from the image plane to the display surface (line 5) –as described in detail in sections 3.7.2 and 3.7.3. Normally the image plane is located within the image space of the mirror optics. In this case, these mappings composite the vertex individual model-view transformations to neutralize reflection and refraction, as well as the projection transformation that maps a vertex onto the display surface. Note that vertices are only transformed once – even if the recursive refinement function addresses them multiple times. This is realized by attaching a marker flag to each vertex. A reference to the transformed patch is stored in patch set *P* by adding the patch's indices to *P* (line 6). In line 7, a function is called that evaluates the transformed patch based on pre-defined refinement criteria and decides whether or not this patch has to be further refined. The main refinement criterion is described below. If this decision is positive, the patch is divided into four equal sub-patches and the refinement function is recursively called for all of these sub-patches (lines 9-12). Note that condition (ii) also allows to determine the indices of the patch's four T-vertices, which become L-vertices of the sub-patches in the next LOD. Consequently, the *GeneratePatch* and the *RefineFurther* functions represent the exit conditions for the recursion.

It was mentioned that T-vertices have to be connected to their neighboring X-vertices whenever an LOD transition occurs to ensure consistency during rasterization. To detect LOD transitions, a counter (TC) is attached to each T-vertex. This counter is incremented by 1, each time the corresponding T-vertex is addressed during the recursive refinement (lines 13-16). Note that the if-conditions ensure a correct behavior of the counter at the image geometry's boundaries. Due to condition (iv) each counter can have one of the following three values:

- *0*: indicates that the T-vertex is located at a boundary edge of the image geometry or it is contained by a discrete LOD that is higher than the required one for the corresponding region.
- *1*: indicates a LOD transition between the two neighboring patches that –with respect to condition (iv)– belong to the T-vertex.
- *2*: indicates no resolution transition between the two neighboring patches that belong to the T-vertex.

After the image grid has been completely generated, all patches that are referred to in *P* are rendered with appropriate texture coordinates during the second rendering pass. Thereby, the counters of the patch's four T-vertices are evaluated. Depending on their values, either one or two triangles are rendered for each counter. These triangles form the final patch. Counter values of 0 or 2 indicate no LOD transition between adjacent patches. Consequently, a single triangle can be rendered which is spanned by the T-vertex's neighboring two L-vertices and the patch's X-vertex (this is illustrated in figure 3.24b). A counter value of 1, however, indicates a LOD transition. Two triangles have to be rendered that are spanned by the T-vertex itself,

the two neighboring L-vertices and the X-vertex of the adjacent patch (this is illustrated in figure 3.24a). A selectively refined image geometry is shown in figure 3.23d.

3.7.8.3. Generation and Refinement Criteria

This section discusses the patch generation and refinement criteria that are implemented within the *GeneratePatch* and *RefineFurther* functions. The input for these functions is the four L-vertices, as well as the X-vertex of a patch. They deliver the Boolean value *true* if the patch has to be generated, transformed, rendered or further refined, or *false* if this is not the case.

The *GeneratePatch* function, that supports an appropriate image clipping, and an evaluation criterion that considers the scene's convex container is described in part 1.

In general, the *RefineFurther* function can represent a Boolean concatenation of multiple refinement criteria (such as maximal patch size, angular deformation, etc.). An important refinement criterion for LOD methods is the *screen space error*. Since the computations of this displacement error are display specific, an important variation of the screen space error that can be applied for convex mirror beam combiners is described -the *image space error*. This error is explained in greater detail in parts 2 through 4.

1.) Spatial Limits

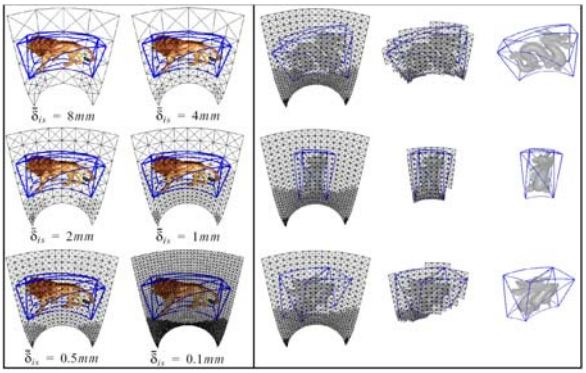
The multi-pass rendering method that is described in sections 3.7.1-3.7.7 uses the scene's *bounding sphere* to determine the parameters of the symmetric viewing frustum and the image size (cf. figure 3.16). Since all image generation methods assume a rectangular image shape that is adapted to today's screen shapes, the bounding sphere provides enough information to determine the rectangular image size.

Bounding spheres, however, are only rough approximations of the scene's extensions and consequently cover a fairly large amount of void space. This void space results in grid patches on the image plane whose texture does not contain visible color information.

To achieve a speed-up, these patches are avoided while creating the image grid. This implies that they are not transformed and refined during the *RecursiveGridRefinement* algorithm and that they are not rendered during the second rendering pass. As a result an image grid is generated and renderd that is not rectangular but dynamically approximates the silhouette of the scene as perceived from the observer's perspective (cf. Figure 3.23e). A condition that causes the recursion to exit in these cases is implemented within the *GeneratePatch* function:

A tighter convex container (such as an oriented convex hull or a bounding box) of the scene is evaluated that is generated either in a pre-process for static objects, or at runtime for animated scenes. For each untransformed patch that is passed recursively into the *RecursiveGridRefinement* algorithm, it has to be determined whether the container is visible on that patch – partially or as a whole. The approximation is twofold: First, the geometric lines of sight from e to all four L-vertices of the patch are intersected with the front-facing portion of the container. Second, the geometric lines of sight from e to front-facing container vertices are intersected with the patch. If at least one of the resulting rays causes an intersection, the patch might contain visible color information and it will be further processed. If, however, none of the rays cause intersections, the patch is not treated further (i.e. it won't be transformed nor refined or rendered). If the convex container is represented as a triangle mesh, a fast *ray-triangle intersection* method [MOE97] is applied together with the front-facing container triangles. Note that as for vertex transformations, the intersection information are buffered and looked up in the memory, rather than re-computing them multiple times while evaluating adjacent patches.

Oriented convex hulls can be used as containers. It is obvious that the complexity of the container influences the performance of this method. Although a precise container can eliminate a maximum number of patches, the number of intersection tests increases with the container's number of vertices and polygons. Experiments have shown that the highest speedups are reached if the container is as simple as an oriented bounding box or a very coarse but tighter convex hull (cf. figure 3.25). However, the complexity of the convex hull that maximizes the speedup and balances intersection tests with patch generations depends on the scene and the required rendering precision.



(a) (b)

Figure 3.25: *Selective grid refinement for different image space error thresholds (a). Spatially limited grids for different perspectives (b).*

2.) Image Space Error

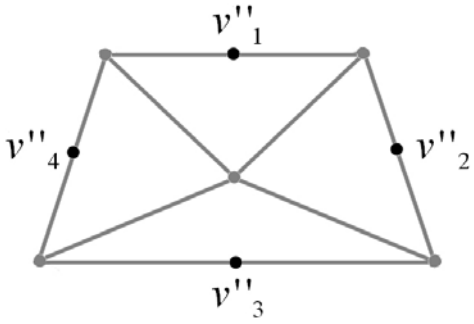
The consideration of the *screen space error* that is determined relative to a display surface is a common measure for many computer graphics methods (e.g., [Hop97, Hop98, Lin96]). In contrast to traditional displays, curved mirror beam combiners create a view-dependent, non-planar image surface which is located inside the mirror optics. Consequently, an appropriate error function has to consider this optical transformation. The error is determined by first warping the curved image surface into an image plane, and then computing the screen space error on this plane. This error is referred to as *image space error* (δ_{is}).

The image space error is a variation of a screen space error that can be computed for convex mirror beam compiners which present the image plane within the image space of their optics, rather than on a display surface. The image space error is defined as the geometric distance between the desired position (v_d) and the actual appearance (v_a) of a point on the image plane. Consequently, the image space error is given by:

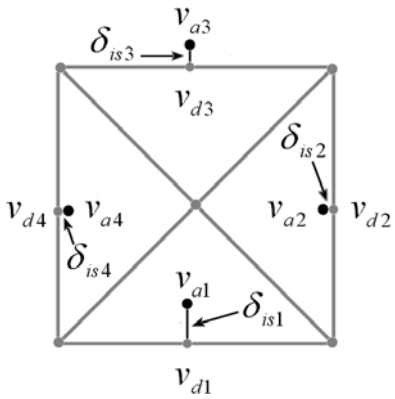
$$\delta_{is} = |v_d - v_a|$$

It delivers results in image space coordinates (e.g., *mm* in this case).

In case of convexly curved mirror optics, the *image space* is the reflection of the *object space* (i.e. the secondary screen in front of the mirror optics) that optically overlays the real environment inside the mirror optics. In addition, the optically deformed pixels do not maintain a uniform size within the image space. They are deformed in exactly the same way as the entire image after being reflected from the object space into the image space - although on a smaller scale. Consequently, the Euclidean distance between geometric points can be chosen as an error metric, rather than expressing the image space error with a uniform pixel size.



(a)



(b)

Figure 3.26: *Samples on the transformed patch (a) The distance between desired and actual appearance of samples near the untransformed patch results in the image space error (b).*

For any given pixel on the transformed patch with texture coordinates *u,v*, δ_{is} can be computed as follows (cf. figure 3.26):

First, the pixel's world coordinate v'' at *u,v* within the object space (i.e. on the secondary display surface) is determined. Note that the pixels, which are mapped onto the patch's transformed vertices, optically appear at their correct locations on the image plane inside the image space. This is because their exact mappings have been determined during the patch's transformation. This transformation considers the laws of geometric optics (i.e., reflection and refraction laws). Note also that the pixels that are displayed anywhere else (i.e. inside one of a patch's triangle) do not necessarily appear at their correct locations on the image plane. This is because their positions on the display surface are approximated by a *linear texture interpolation*, rather than by optical laws.

The second step is to determine the position of the optical image (v') of v'' within the image space of the mirror optics. The projection of v' onto the image plane results in v_a. The transformation from v'' to v_a will be discussed in detail in part 3.

In an ideal case, v_a is located at the position that also maps to the texture coordinates *u,v* within the untransformed patch. We can identify the location which does this as the desired image position v_d. However, if $v_a \neq v_d$, the image space error δ_{is} for this pixel is non-zero.

The image space errors for the four points on the transformed patch (figure 3.26a) can be computed that should map to the patch's T-vertices on the untransformed patch (figure 3.26b) as if the image space errors were zero for these positions. Obviously, this is not the case in the example, shown in figure figure 3.26b.

Since the untransformed patch is a *rectangular quad*, small image space errors suggest that the optical mapping between the transformed and untransformed patch is linear. Furthermore, it can then be concluded that a linear texture interpolation within the displayed transformed patch produces approximately correct results while being mapped (i.e., reflected and refracted) into image space. Consequently, it can be assumed that the resulting image space errors describe a patch's curvilinearity at the representative pixel locations.

To decide whether or not a patch has to be further refined, the largest of the four image space errors is determined. If it is above a pre-defined threshold value $\overline{\delta}_{is}$ the patch has to be further refined and the *RefineFurther* returns *true*.

3.) Computing Object-Image Reflections

To compute v_a from a given viewpoint e, the object point v'' and the optic's geometry is equivalent to finding the *extremal Fermat path* from v'' to e via the optics. In general, this would be a difficult problem of *variational calculus*.

Beside ray- and beam-tracing approaches, several methods have been proposed that approximate reflection on curved mirrors to simulate global illumination phenomena within rendered 3D scenes. All of these methods face the above mentioned problem in one or the other way.

Mitchell and Hanrahan [Mit92], for instance, solve a multidimensional non-linear optimization problem for explicitly defined mirror objects ($g(x) = 0$) with *interval techniques*. To compute reflected illumination from curved mirror surfaces, they seek the *osculation ellipsoid* that is tangent to the mirror surface, whereby its two focal points match the light source and the object point.

For a given viewpoint e, Ofek and Rappoport [Ofe98] spatially subdivide the object space into *truncated tri-pyramid shaped cells*. In contrast to solving an *optimization problem*, they apply *accelerated search techniques* to find the corresponding cell that contains the object v'' at interactive rates.

While Mitchell's *multidimensional optimization* approach is far from being applied at interactive rates, Ofek's search method offers a good approximation for rendering global illumination effects (such as reflections), but does not provide the precision required by an optical display.

In the following, a numerical minimization method to compute the object-image reflection for specific parametric mirror surfaces (such as cones and cylinders) is described. For such surfaces, the optimization problem be reduced to only one dimension. Consequently, the method provides an appropriate precision at interactive rates.

For the subsequent example the same cone-shaped mirror surface is chosen as used for the examples in sections 3.7.1-3.7.8.

Cones and similar *bodies of revolution* have the property that multiple surface points lie on a common plane. For example, all points on the straight line spanned

by a cone's peak and an arbitrary point on its bottom circle lie on the same plane. Consequently, individual plane parameters can be determined for all angles around the cone's principle axis.

To determine an object's (v''_d) image for a given viewpoint e and mirror surface $g(x)=0$, an arbitrary angle α around the cone's principle axis is assumed. Then the surface's tangent plane TP_1 is determined at α by computing a surface point and the surface normal at α. Since $g(x)$ is an explicit function, the surface normal can be computed by using its first-order derivatives $g/(\partial x)$. Next, v''_d is reflected over TP_1 to its corresponding position within the image space and it is projected onto the image plane. In figure 3.27, the projected image point is outlined by v.

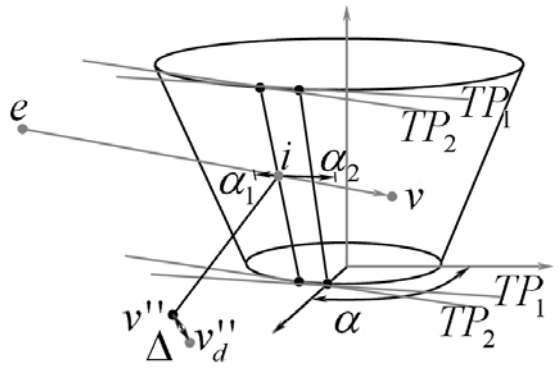


Figure 3.27: *Object-image reflection via nummerical minimization.*

To verify the quality of this assumption, v is reflected back into the object space and project it onto the display surface. For a given v, $g(x)$ and e, a simple *analytical solution* exists to determine this *image-object transformation*: The ray spanned by e and v is intersected with $g(x)$ by solving a simple quadratic equation. In the case of a conical beam combiner, the *quadratic equation* is given by inserting the linear ray equation into the quadratic equation of a cone, and solving for x. The surface intersection i, together with the normal vector at i (again determined using the surface's first-order derivatives) gives the tangent plane TP_2 at i. Reflecting v and e over TP_2 and projecting the reflection of v onto the display surface using the reflection of e as center of projection, results in point v''. The image-object transformation is illustrated in figure 3.16c. Note that for simplicity, the image-object transformation and the *object-image transformation* have been described as simple reflection/projection transformations. Normally, they incorporate refraction as well, as described in section 3.7.3.

If the tangent plane at α produces the *extremal Fermat path* between v''_d and e, the geometric distance Δ between v''_d and v'' is zero, $TP_1 = TP_2$, and $v_a = v$. Otherwise Δ is non-zero.

To approximate the correct α, the above-described function is minimized for Δ. Since this function depends only on α, a fast numerical optimizers for one dimension can be applied. However, because its first order derivatives cannot easily be derived but it appears to be nicely parabolic near its minimum, Brent's inverse parabolic interpolation [Bre73] with bracketing is applied.

To speed up the minimization process (i.e. to reduce the number of function iterations), the function range for a particular v''_d can be constrained. As illustrated in figure 3.27, the minimization is restricted to find α between α_1 and α_2. These boundaries can be determined as follows: Given e and the untransformed patch that belongs to v''_d, the projections on the mirror surface of the untransformed patch at the next lower LOD are evaluated. The two angles α_1 and α_2 at the horizontal extrema of these projections are the corresponding *boundary angles*. Note that these projections are determined while transforming the patches (i.e., within *TransformPatchVertices*). Thus, for efficiency reasons, they are stored and looked up in the memory, rather than re-computing them again.

Experiments showed that sufficiently precise solutions can be found after a small number of iterations. Typically, average errors of $\Delta = 0{,}002mm$ are achieved with an average of 3 iterations.

4.) Error Direction Propagation

Although the number of iterations is relatively small, the computational expenses of four minimization steps per patch result in a noticeable loss of performance. Especially while evaluating the large number of higher LOD patches, such an approach might not produce a speedup.

It can be noticed that the direction in which the highest image space error (i.e. the one of the four patch sides where δ_{is} is maximal) propagates is the same for higher LOD patches that are derived from the same parent patch.

However, from which level of detail on this is the case depends on the display's properties. If, for instance, the optics produce well-behaved image deformations, the propagation direction of the highest error is consistent for relatively high LODs. If, on the other hand, the optics produces noisy images, the error direction alternates as randomly.

To benefit from this situation, we specify a LOD Λ depending on the optic's and the display surface's properties.

For patches that are below Λ, the image space error is determined as described above: δ_{is} is compute at all four edges and the highest value is found. In addition, the error direction (i.e. the edge where δ_{is} is maximal) is recorded for each patch.

For patches that are above Λ the error direction of the corresponding parent patch can be reused and δ_{is} can be computed only for the edge in this direction, rather than at all four edges. By doing this, it is assumed that the largest error will occur in the same direction as for the parent patch. Consequently, the number of minimization steps is reduced from four to one for all patches that are above Λ - i.e., for the majority of all relevant grid patches.

Experiments have shown that this heuristic leads to a significant speedup while producing the same visual final output.

Note that although the examples above deal with a simple mirror shape (i.e., a cone), an algorithm that uses a pre-computed look-up table (e.g., such as the one described by Ofek [Ofe98]) instead of dynamic numerical minimizations would either result in fairly large data-structures that cannot be efficiently searched, or in a precision that is not adequate for an optical display.

3.7.8.4. Display Specific Components

While the described algorithm is valid for other displays that require non-linear pre-distortion, its display specific components have been explained based on a particular mirror display -a cone-shaped beam bombiner. The nature of the additional mirror optics makes the transformation of the grid patches and the computation of the resulting displacement error fairly complex. In fact, the implementation of the *TransformPatchVertices* and the *RefineFurther* functions have been explained with an emphasize on cone-shaped mirror surfaces. These two functions represent the display specific components of our algorithm. While for a mirror display, *TransformPatchVertices* and *RefineFurther* have to consider laws of geometric optics (such as reflection and refraction transformations) to map grid vertices from the image plane onto the projection surface and vice versa, they can be generalized to do the same for other displays without modifying the general algorithm.

If, for instance, the algorithm is used to project images onto curved screens (e.g., a cylindrical or spherical projection device), *TransformPatchVertices* would incorporate only projection transformations (i.e., it would only determine intersections of vertex-individual geometric lines of sight with the display surface). The resulting displacement error that is computed by *RefineFurther* can then be determined within the object space (e.g., the screen space), rather than within an image space. Compared to our numerical approach for convex mirror displays, this would be less complex since it involves only simple intersection computations for which analytical solutions exist. If a view-dependence is not required, *TransformPatchVertices* and *RefineFurther* could also retrieve pre-computed values from a look-up table.

For curved mirror beam combiners that require a correction of a non-linear distortion by applying multi-pass rendering, generating appropriate local levels of detail instead of applying uniform grids or constant image geometries allows to consider the error that is caused from a piecewise linear texture interpolation and to minimize it by adapting the underlying geometry.

The method described above prevents oversampling and texture artifacts that result from undersampling. If executed on CPUs, a significant speed up can be achieved by rendering a selectively refined image geometry instead of a high-resolution uniform image grid. However, the capability of per-vertex operations of today's GPUs beats the computational overhad that is required for recusions and minimizations with a faster transformation of high-resolution uniform image geometries. This is described in sections 3.7.2 nd 3.7.3.

3.8. Summary and Discussion

The basic principles of geometric optics represent the mathematical, physical and physiological foundations for this chapter. Planar mirror beam combiners that produce true stigmatic reflections between all object-image pairs, and convexly curved mirror beam combiners that represent non-absolute optical systems represent the main components for the creation of optical overlays in combination with spatial secondary screens.

If such beam combiners are integrated into the optical path between observer and secondary screen, the perceived graphical images are transformed and distorted by the optical elements. We have discussed real-time rendering techniques that are capable of neutralizing these effects for different display configurations in such a way that orthoscopic, stereoscopically and perspectively correct and undistorted images can be seen from any viewing position. Transparent primary screens, such as active or passive screens which can be utilized for spatial optical combination have been outlined briefly. Their rendering techniques are the same as for planar opaque projection screens.

Mirror beam combiners that consist of a half-silvered mirror layer coated or impregnated onto or into a transparent base carrier represent a combination of lens and mirror. Consequently, reflection transformation and refraction distortion have to be considered.

Since planar mirrors are absolute optical systems, affine model and viewpoint transformations can be applied for such elements. These transformations can be integrated into traditional fixed function rendering pipelines and consequently can be fully supported by hardware acceleration without causing additional computational costs.

In contrast to this, elements that cause significant refraction and warped reflection distortion require curvilinear image transformations. A multi-pass image-warping method avoids a direct access to the scene geometry, and consequently prevents the time-consuming transformations of many scene vertices. In addition, it is not restricted to a geometry-based first rendering pass, but rather supports any perspective image generating rendering method (such as point-based rendering, image-based rendering, interactive raytracing and radiosity, or non-photorealisitc rendering). It can be easily configured and extended to support new optics and display technology, and it is flexible enough to be smoothly integrated into existing software frameworks. Additionally, it takes as much advantage of hardware implemented rendering concepts (such as vertex-shaders, and render-to-texture capability) as currently possible. Such optics also cause aberrations of non-stigmatic mappings that cannot be corrected in software. However, due to the limited retinal resolution of the eyes small aberrations are not detected.

To efficiently support simultaneous rendering for multiple viewers on cost-effective rendering platforms (e.g., PCs), a networked cluster of rendering nodes could be applied. Each node would carry out the image generation and deformation for a single observer. However, in terms of displaying the final images on arbitrary locations of a single projection screen, they need to be composed within one image. One possibility for doing this is to utilize coding and streaming technology to transmit the images to a central node that is responsible for their composition and the presentation of the resulting image. However, the transmission of the images over an

all-purpose network (e.g., an Ethernet) as well as their explicit composition might represent bottlenecks that slow down the overall rendering performance significantly. Recent advances in hardware *image composition technology* (such as [Stol01]) solve this problem. Instead of connecting the rendering nodes via an all-purpose network, they can be connected to a specific display subsystem. These subsystems allow the image data generated by the connected nodes to be mapped to any location of the connected displays without losing much time for transmission and composition. Some devices (such as the Lighning-2 device [Stol01]) are able to scale in both dimensions -the number of rendering nodes and the number of displays. Such technology also supports to efficiently drive high-resolution displays that are built from a collection of multiple display units.

In general several advantages of spatial optical combination over head-attached optical see-through exist:

- *Easier eye accommodation and vergence*: using mirror beam combiners, for instance, the reflected image plane can be brought very close the physical location that has to be overlaid. Consequently multiple focal planes do not differ much. In addition, the reflected image remains at a static position in space if neither mirror combiner nor secondary screen is not moved. This position is invariant of the user's location (note: the presented content is not). Moving closer to or further away from the physical location and the reflected image yields the same natural accommodation and vergence effects. This is not the case for head-attached displays that present the image at a constant distance in front of the observer. In this case the image's position and it's focal plane do not chance for a moving user. Consequently multiple focal planes (i.e., in the physical world and on the image plane) make it impossible to focus at real and virtual content at the same time. Either real or virtual environment have to be perceived unfocused. This requires that the observer continuously switches focus;
- *High and scalable resolution*: spatial screens, such as projection display or monitors provide a high resolution. If, however, a single display does not offer the resolution that is required for a particular application, multiple image sources (such as video beamer) can be combined. Packing issues that are related to the miniature displays which have to be integrated into the limited space of for

head-attached devices are not a problem for spatial configurations. In addition, the compression characteristics of convex mirror that map a larger screen portion into a smaller reflection area causes a high density of pixels. Consequently the spatial resolution of such a display is higher than the resolution of the secondary screen;

- *Large and scalable field of view*: the same argumentation as for the resolution applies to the field of view. Screens and optics are theoretically not constrained in their dimensions. Spatial surround screen display, for instance, can fill out the entire field of view – a characteristic that cannot be achieved with today's head-attached devices;
- *Improved ergonomic factors*: today's head-attached displays suffer from an unbalanced ratio between cumbersome and high quality optics. Spatial displays do not require the user to wear much technology. While stereoscopic displays imply the application light-weight passive or active glasses, autostereoscopic displays detach any kind of technology form the user;
- *Easier and more stable calibration*: while some head-attached displays can have up to 12 degrees of freedom, spatial optical see-through configurations generally have significantly less. This implies an easier and user/session independent calibration, and consequently yields more precise overlays;
- *Better controllable environment*: a limited indoor space can be easier controlled than large scale outdoor environments. This applies to tracking and illumination conditions, and leads to higher precision and consistency, as well as to better quality.

Beside these advantages, several disadvantages can be found with respect to head-attached approaches:

- *Mutual occlusion*: Optical beam combiners cannot present mutual occlusion of real and virtual environments. Consequently, bright real surfaces overlaid with dark virtual objects optically emphasize the real environment and let the virtual objects disappear or appear semi-transparent. A general solution to this problem for spatial optical see-through displays is presented in chapter 4;
- *Window violation*: the window violation or clipping problem is linked to fish-tank-sized and semi-immersive screens. Graphics that –due to a disadvantageous user position– is projected outside the display area is unnaturally cropped. This effect also appears with the mirror beam combiners: displayed images that –due to a disadvantageous user position– cannot be reflected are cropped at he mirror edges as well.
- *Multi-user viewing*: the number of observers that can be supported simultaneously is restricted by the applied optics and screen configuration;
- *Support mobile of applications*: Spatial displays do not support mobile applications because of the spatially aligned optics and display technology;
- *Direct interaction*: In most cases, the applied optics prevents a direct manipulative interaction. Real objects might not be touchable because they are out of arm reach or because the additional optics represents a physical barrier. Indirect interaction methods have to be applied in these cases.

In general, it has to be annotated that upcoming and new technology will not only open new possibilities for the spatial optical see-through concept, but also for other display concepts, such as head-attached displays. *Organic Light Emitting Diodes* (OLEDs), for instance, may replace the crystalline LEDs that are currently being used to build the miniature displays for HMDs.

OLEDs promise to produce cheap and very high-resolution full-color matrix displays that can give head-attached displays a technological push. A variant of OLEDs are *Light Emitting Polymers* (LEPs) that provide the opportunity for the fabrication of large, flexible, full-color emissive displays with a high resolution, a wide viewing angle and a high durability. Large-scale transparent LEPs may present a future variation of optical combination with an active screen. However, LEPs have not yet left the basic research stages and will not be applicable to build stereoscopic AR displays in the near future. But in combination with autostereoscopic or holographic techniques this problem may be solved.

In the short run, especially high-resolution and bright display devices, high-performance and cost-efficient rendering hardware, reliable and precise tracking technology, and advanced interaction techniques and devices will pave the way for forthcoming spatial optical see-through configurations. However, the underlying technology must be robust, flexible and the technology that directly interfaces to users should adapt to humans, rather than forcing users to adapt to the technology. Therefore, human-centered and seamless technologies, devices and techniques will play a major role for augmented reality in future.

References

[All03] Allard, J., Gouranton, V., Lamarque, G., Melin, E., and Raffin, B. Softgenlock: active stereo and GenLock for PC clusters. In proceedings of Immersive Projection Technology and Eurographics Virtual Environments Workshop 2003 (IPT/EGVE'03), pp. 255-260, 2003.

[Azu97] Azuma, R. T., A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments, vol. 6, no. 4, pp. 355-385, 1997.

[Bim00] Bimber, O., Encarnação, L.M., and Schmalstieg, D. Augmented Reality with Back-Projection Systems using Transflective Surfaces. Computer Graphics Forum (proceedings of EUROGRAPHICS 2000 - EG'2000), vol. 19, no. 3, pp.161-168, 2000.

[Bim01a] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55, 2001.

[Bim01b] Bimber, O., Encarnação, L.M. and Branco, P. The Extended Virtual Table: An Optical Extension for Table-Like Projection Systems. Presence: Teleoperators and Virtual Environments, vol.10, no. 6, pp. 613-631, 2001.

[Bim02] Bimber, O. Interactive rendering for Projection-Based Augmented Reality Displays. Ph.D. Dissertation, University of Technology Darmstadt, http://elib.tu-darmstadt.de/diss/000270/, 2002.

[Bim03] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. Real-Time View-Dependent Image Warping to correct Non-Linear Distortion for Curved Virtual Showcase Displays. To appear in Computers and Graphics - The international Journal of Systems and Applications in Computer Graphics, vol. 27, no. 4, 2003.

[Bre73] Brent, R.P. Algorithms for minimization without derivatives. Prentice-Hall, Engelwood Cliffs, NJ, 1973.

[Hec84] Heckbert, P. and Hanrahan, P. Beam tracing polygonal objects. Compute Graphics (proceedings of SIGGRAPH'84), vol. 18, no. 3, pp. 119-127, 1984.

[Hop96] Hoppe H. Progressive meshes. Computer Graphics (Proceedings of SIGGRAPH'96), New Orleans, LA, p. 99–108, 1996.

[Hop97] Hoppe H. View-dependent refinement of progressive meshes. Computer Graphics (Proceedings of SIGGRAPH'97), Los Angeles, CA, p. 189–97, 1997.

[Hop98] Hoppe H. Smooth view-dependent level-of-detail control and its application to terrain rendering. Proceedings of IEEE Visualization'98, Research Triangle Park, NC, p. 35–42, 1998.

[Lin96] Lindstrom P, Koller D, Ribarsky W, Hughes L, Faust N, Turner G. Realtime, continuous level of detail rendering for height fields. Computer Graphics (Proceedings of SIGGRAPH'96), New Orleans, LA, p. 109–18,1996.

[Mck99a] McKay, S., Mason, S., Mair, L. S., Waddell, P., and Fraser, M. Membrane Mirror-Based Display For Viewing 2D and 3D Images. In proceedings of SPIE, vol. 3634, pp. 144-155, 1999.

[Mit92] Mitchell, D. and Hanrahan, P. Illumination from Curved Reflectors. Computer Graphics (proceedings of SIGGRAPH'92), pp. 283-291, 1992.

[Moe97] Möller, T., and Trumbore, B. Fast, Minimum Storage Ray-Triangle Intersection. Journal of graphics tools. vol. 2, no. 1, pp. 21-28, 1997.

[Mck99b] McKay, S., Mason, S., Mair, L. S., Waddell, P., and Fraser, M. Stereoscopic Display using a 1.2-M Diameter Stretchable Membrane Mirror. In proceedings of SPIE, vol. 3639, pp. 122-131, 1999.

[Ofe98] Ofek, E. and Rappoport A. Interactive reflections on curved objects. Computer Graphics (proceedings of SIGGRAPH'98), pp. 333-342, 1998.

[Ofe99] Ofek. E. Interactive Rendering of View-Dependent Global Lighting Phenomena. Ph.D. Dissertation, Hebrew University (Israel), 1999.

[Rol93] Rolland, J.P., and Hopkins, T. A Method of Computational Correction for Optical Distortion in Head-Mounted Displays. (Tech. Rep. No. TR93-045). UNC Chapel Hill, Department of Computer Science, 1993.

[Seg92] Segal, M., Korobkin, C., van Widenfelt, R. Foran, J., and Haeberli, P. Fast Shaddows and Lighting Effects Using Texture Mapping. Computer Graphics (proceedings of SIGGRAPH'92), vol. 26, no. 2, pp. 249-252, 1992.

[Spa92] Spanguolo, M. Polyhedral surface decomposition based on curvature analysis. Modern Geometric Computing for Visualization, Springer Verlag, 1992.

[Stol01] Stoll, G., Eldridge, M., Patterson, D., Webb, A., Berman, S., Levy, R., Caywood, C., Taveira, M., Hunt, S., Hanrahan, P. Lightning-2: A high-performance subsystem for PC clusters. Computer Graphics (proceedings of SIGGRAPH'01), 2001, pp. 141-148.

[Wat95] Watson, B., and Hodges, L. Using Texture Maps to Correct for Optical Distortion in Head-Mounted Displays. In proceedings of IEEE VRAIS'95, pp. 172-178, 1995.

[Wei99] Wiegand, T. E., von Schloerb, D. W., and Sachtler, W. L. Virtual Workbench: Near-Field Virtual Environment System with Applications. Presence: Teleoperators and Virtual Environments, vol. 8, no. 5, pp. 492-519, 1999.

4. Projector-Based Illumination and Augmentation

Their increasing capabilities and declining cost make video projectors become widespread and established presentation tools. Being able to generate images that are larger than the actual display device virtually anywhere is an interesting feature for many applications that cannot be provided by desktop screens. Several research groups discover this potential by applying projectors in unconventional ways to develop new and innovative information displays that go beyond simple screen presentations.

The *Luminous Room* [Und99] for instance, describes an early concept for providing graphical display and interaction at each of an interior architecture space's surface. Co-located two-way optical transducers –called *I/O bulbs*– that consist of projector-camera pairs capture the user interactions and display the corresponding output. With the *Everywhere Displays projector* [Pin01] this concept has been extended technically by allowing a steerable projection using a pan/tilt mirror. Recenty, it was demonstrates how context-aware hand-held projectors –so-called *iLamps*– can be used as mobile information displays and interaction devices [Ras03].

Another concept called *Shader Lamps* [Ras01] approaches to lift the visual properties of neutral diffuse objects that serve as projection screen. The computed radiance at a point of a non-trivial physical surface is mimicked by changing the BRDF and illuminating the point appropriately with projector pixels. Animating the projected images allows creating the perception of motion without physical displacement of the real object [Ras02]. This type of spatial augmentation is also possible for large, human-sized environments, as demonstrated in [Low01].

Projector-based illumination has become an effective technique in augmented reality to achieve *consistent occlusion* [Nod99, Bim02a] and *illumination* [Bim03] effects between real artifacts and optically overlaid graphics. Video projectors instead of simple light bulbs are used to illuminated physical objects with arbitrary diffuse reflectance. The per-pixel illumination is controllable and can be synchronized with the rendering of the graphical overlays. It also makes the combination of high-quality *optical holograms* with interactive graphical elements possible [Bim04a]. Using a video projector to produce a controlled reference wave allows reconstructing the hologram's object wave partially – not at those portions that are overlaid by integrated graphical elements.

New optical combiners together with real-time color correction methods allow to effectively *superimposing arbitrarily textured surfaces*, such as paintings [Bim04b]. This is enabled by thin, transparent film materials that

diffuse a fraction of the light projected onto them. Such a film can be seamlessly integrated into the frame that holds the artwork. Any kind of visual information, such as animations or interactive graphical elements can be overlaid in correct colors.

A virtual retouching technique that applies video projectors for reconstructing and enhancing the faded colors of paintings by projecting light directly onto the canvas is described in [Yos03]. An affine correlation between projection and the result captured by a camera is established manually for each pixel. Users are then able to retouch the original painting interactively via a desktop GUI.

4.1. Projector-based augmentation

In projector-based augmentation, the user's physical environment is augmented with images that are integrated directly in the user's environment, not simply in their visual field [Ras98c, Ras99]. For example, the images could be projected onto real objects using digital light projectors, or embedded directly in the environment with flat panel displays. For the purpose of this discussion we will focus on projector-based augmented reality. While the approach has certain restrictions, it offers an interesting new method to realizing compelling illusions of virtual objects coexisting with the real world. The images could appear in 2D, aligned on a flat display surface, or they could be 3D and floating above a planar or even a non-planar surface. In the most basic applications, projector-based augmentation combines the benefits of traditional spatially immersive displays and augmented reality displays.

The idea of projector-based augmentation is best used when virtual objects are close to the physical objects on which they are displayed. For example, an architect can augment a tabletop scaled model of a house or building using a projector. She can start with a very simple neutral colored cardboard model and its geometric CAD representation. Then it is easy to add virtual objects such as doors, windows, chimneys. She can also visualize underground water pipes or support structure inside the building. A compelling example of projector-based augmentation is the application aimed at walk-through of virtual human-sized environments built by Kok-lim Low et. al. in the Being There project at UNC [Low01]. Instead of building an exact detailed physical replica for projection, the display is made of simplified versions. For example, primary structures of building interiors and mid-sized architectural objects (walls, columns, cupboards, tables, etc.), can usually be approximated with simple components (boxes, cylinders, etc.). As seen in the Figure 2.1.5, the display is made of construction Styrofoam blocks. The main architectural features that match the simplified physical model retain 3D auto-stereo, but the other details must be presented by projecting view-dependent images. Nevertheless, the experiment to simulate a building interior is convincing and provides a stronger sense of immersion when compared to spatual immersive displays, as the user is allowed to really walk around in the virtual environment. However, strategic placement of projectors to allow complete illumination and avoiding user shadows is critical.

4.1.1. Shader Lamps

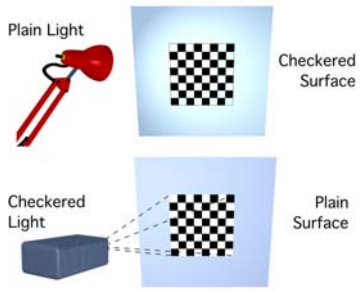


Figure 4.1: *The idea of ShaderLamps*

In this section, we describe a special case of projector-based augmentation. The idea is to replace a physical object with its inherent color, texture, and material properties with a neutral object and projected imagery, reproducing the original appearance directly on the object. Furthermore the projected imagery can be used to reproduce alternative appearances, including alternate shading, lighting, and even animation. The approach is to effectively lift the visual properties of the object into the projector and then re-project onto a neutral surface (Figure 4.1). We use the phrase shader lamps to describe this mode of operation for projectors [Ras01]. Consider the effect shown in Figure 4.2. The underlying physical object is a white diffuse vase. (The other objects such as the book and flowers are also real objects.) Can we make this white vase appear to look like it is made of marble, plastic or metal? Can we change the color or texture? The pictures show that the vase can be effectively 'painted' by projecting an image with view-independent diffuse shading, textures and intensity correction. The view-dependent effects such as specular highlights are generated for a given user location by modifying reflectance properties of the graphics model. The figure shows appearance of a red plastic and a green metallic material on the clay vase. Although, there have

been other attempts at augmenting appearances of objects by projecting color or texture, those effects are very limited and have been achieved for only specific applications. In this section, we show that the real challenges to realizing this as a new medium for computer graphics lies in addressing the problems related to complete illumination of non-trivial physical objects. The approach presented here offers a compelling method of visualization for a variety of applications including dynamic mechanical and architectural models, animated or living dioramas, artistic effects, entertainment, and even general visualization for problems that have meaningful physical shape representations. We present and demonstrate methods for using multiple shader lamps to animate physical objects of varying complexity, from a flower vase (Figure 4.2), to some wooden blocks, to a model of the Taj Mahal (Figure 2.16).

Figure 4.2: *(Left) The underlying physical object is a white diffuse vase. (Middle and right) View-dependent effects, such as specular highlights, can be generated by tracking the user's location and projecting images on the vase.*

4.1.1.1. Illumination Process

We introduce the idea of rearranging the terms in the relationship between illumination and reflectance to reproduce equivalent radiance at a surface. As shown in flatland in Figure 4.3, the radiance in a certain direction at point (x), which has a given BRDF in the physical world (left), can be mimicked by changing the BRDF and illuminating the point with a appropriately chosen light source, e.g. a projector pixel (right). Below we identify a radiance adjustment equation for determining the necessary intensity of a projector pixel, given the position and orientation of the viewer and the virtual scene. For a more systematic rendering scheme, we describe the notion of separating the *rendering view*—the traditional virtual camera view, from the *shading view*—the position of the viewer for lighting calculations.

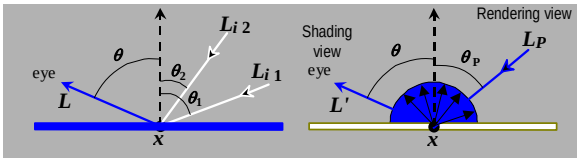


Figure 4.3: *(Left) The radiance at a point in the direction (θ, ϕ). (Right) The radiance as a result of illumination from a projector lamp. By rearranging the parameters in the optical path, the two can be made equal.*

First, let us consider the rendering equation, which is essentially a geometrical optics approximation as explained in [Kaj86]. The radiance at a visible surface point (x) in the direction (θ,ϕ) that would reach the observer of a physical realization of the scene is

$$L(x,\theta,\phi) = g(x,\theta,\phi)(L_e(x,\theta,\phi) + h(x,\theta,\phi))$$

where

$$h(x,\theta,\phi) = \int_i F_r(x,\theta,\phi,\theta_i,\phi_i) L_i(x,\theta_i,\phi_i)\cos(\theta_i) d\omega_i$$

and $g(x,\theta,\phi)$ is the geometry term (visibility and distance), $L_e(x,\theta,\phi)$ is the emitted radiance at the point (non-zero only for light sources), and $F_r(x,\theta,\phi,\theta_i,\phi_i)$ is the BRDF of the point. The integral in $h(x,\theta,\phi)$ accounts for all reflection of incident radiance $L_i(x,\theta_i,\phi_i)$ from solid angles $d\omega_i$. Radiance has dimensions of energy per unit time, area and solid angle.

Treating the projector lamp as a point emitter, the radiance due to direct projector illumination at the same surface point at distance $d(x)$ but with diffuse reflectance $k_u(x)$ is given by

$$L'(x,\theta,\phi) = g(x,\theta,\phi) k_u(x) I_p(x,\theta_p,\phi_p)\cos(\theta_p)/d(x)^2$$

where $I_p(x,\theta_p,\phi_p)$ = radiant intensity of projector in the direction (θ_p,ϕ_p) and is related to a discretized pixel value via filtering and tone representation.

We can reproduce radiance $L'(x,\theta,\phi)$ equivalent to $L(x,\theta,\phi)$ for a given viewer location, by solving Equation (3) for I_p:

$$I_p(x,\theta_p,\phi_p) = \frac{L(x,\theta,\phi)d(x)^2}{k_u(x)\cos(\theta_p)} \quad \text{for } k_u(x) > 0.$$

Thus, as long as the diffuse reflectance $k_u(x)$ is nonzero for all the wavelengths represented in $L(x,\theta,\phi)$, we can effectively represent the surface attribute with appropriate pixel intensities. In practice, however, the range of values we can display are limited by the brightness, dynamic range and pixel resolution of the projector.

The rendering process here involves two viewpoints: the user's and the projector's. A simple approach would be to first render the image as seen by the user, which is represented by $L(x,\theta,\phi)$, and then use traditional image-based rendering techniques to warp this image to generate the intensity-corrected projected image, represented by $I_p(x,\theta_p,\phi_p)$ [Che93, McM95]. For a changing viewer location, view-dependent shading under static lighting conditions can also be implemented [Deb98, Lev96, Gortler96]. However, the warping can be avoided in the case where the display medium is the same as the virtual object. For a single-pass rendering, we treat the moving user's viewpoint as the shading view. Then, the image synthesis process involves rendering the scene from the projector's view, by using a perspective projection matrix that matches the projector's intrinsic and extrinsic parameters, followed by radiance adjustment. The separation of the two views offers an interesting topic of study. For example, for a static projector, the visibility and view-independent shading calculations can be performed just once even when the user's viewpoint is changing.

To realize a real-time interactive implementation we use conventional 3D rendering APIs, which only approximate the general rendering equation. The BRDF computation is divided into view-dependent specular, and view-independent diffuse and ambient components. View-independent shading calculations can be performed by assuming the rendering and shading view are the same. (The virtual shadows, also view-independent, are computed using the traditional two-pass shadow-buffer technique.) For view-dependent shading, such as specular highlights (Figure 4.3), however, there is no existing support to separate the two views. A note in the appendix describes the required modification.

4.1.1.2. Secondary Scattering

Shader lamps are limited in the type of surface attributes that can be reproduced. In addition, since we are using neutral surfaces with (presumed) diffuse characteristics, secondary scattering is unavoidable and can potentially affect the quality of the results. When the underlying virtual object is purely diffuse, sometimes the secondary scattering can be used to our advantage. The geometric relationships, also known as form factors, among parts of the physical objects, are naturally the same as those among parts of the virtual object. Consider the radiosity solution for a patch *i* in a virtual scene with *m* light sources and *n* patches:

$$B_{i\text{-intended}} = k_{d_i}\sum_j B_j F_{i,j} = k_{d_i}\left(\sum_m B_m F_{i,m} + \sum_n B_n F_{i,n}\right).$$

Here k_d is the diffuse reflectance, B_j is the radiance of patch *j*, and $F_{i,j}$ is the form factor between patches. Using shader lamps to reproduce simply the effect of direct illumination (after radiance adjustment), we are able to generate the effect of *m* light sources:

$$B_{i\text{-direct}} = k_{d_i}\sum_m B_m F_{i,m}.$$

However, due to secondary scattering, if the neutral surfaces have diffuse reflectance k_u, the perceived radiance also includes the secondary scattering due to the *n* patches, and that gives us

$$B_{i\text{-actual}} = B_{i\text{-direct}} + B_{i\text{-secondary}} = k_{d_i}\sum_m B_m F_{i,m} + k_u\sum_n B_n F_{i,n}.$$

The difference between the desired and perceived radiance is

$$\left|(k_{d_i} - k_u)\sum_n B_n F_{i,n}\right|.$$

Thus, in scenarios where k_d and k_u are similar, we get approximate radiosity for "free"—projection of even a simple direct illumination rendering produces believable "spilling" of colors on neighboring parts of the physical objects. From the equation above, the secondary contribution from the neutral surfaces is certainly not accurate, even if we reproduce the first bounce exactly. The difference is even larger when the virtual object has non-lambertian reflectance properties. We are currently investigating *inverse global illumination* methods so that the projected image can more accurately deliver the desired global illumination effect. Figure 4.4 shows a green and a white paper with spill over from natural white and projected green illumination. In this special case, the secondary scattering off the horizontal white surface below is similar for both parts.

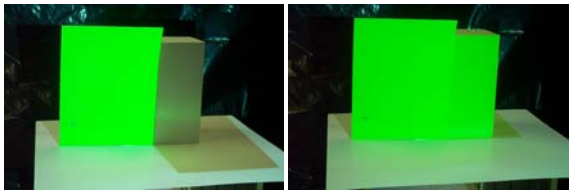


Figure 4.4: *(Left) A green paper illuminated with white light. (Right) The white diffuse surface on the right is illuminated with green light.*

4.1.2. Changing surface appearance of non-trivial projection screens

The image-based illumination of physical objects has been explored by many. But, we believe, two main challenges have kept the previous efforts to only expensive, large scale, or one-off implementations. (a) First, the geometric registration problem, which is cast as matching the projection of a single 2D image with an object. The projection of a perspective device has up to 11 degrees of freedom (6 external and 5 internal) [Fau93], therefore, any effort to manually achieve the registration is likely to be extremely tedious. We propose a new simple technique below. (b) The second problem, which appears to be unexplored, is the complete illumination of non-trivial physical objects in presence of shadows due to self occlusion. With the advent of digitally-fed projectors and real-time 3D graphics rendering, a new approach for image-based illumination is now possible. We approach these problems by creating a 3D geometric understanding of the display setup. We describe an important intensity correction step and our solution for dealing with shadows.

4.1.2.1. Authoring and Alignment

One of the important tasks in achieving compelling visualization is to create the association between the physical objects and the graphics primitives that will enhance those objects when projected. For example, how do we specify which texture image should be used for the face of a building model, or what color distribution will look better for a physical object? We need the physical object as well as its geometric 3D representation, and real or desired surface attributes. As mentioned earlier, many hardware and software solutions are now available to scan/print 3D objects and capture/create highly detailed, textured graphics models. The authoring can also be done interactively by "painting" directly on top of the physical objects. The result of the user interaction can be projected on the objects and also stored on the computer. Ideally, a more sophisticated user interface would be used to create and edit graphics primitives of different shape, color and texture.

To align a projector, first we *approximately* position the projector and then *adapt* to its geometric relationship with respect to the physical object. That relationship is computed by finding projector's intrinsic parameters and the rigid transformation between the two coordinate systems. This is a classical computer vision problem [Fau93]. As seen in Figure 4.5, we take a set of fiducials with known 3D locations on the physical object and find the corresponding projector pixels that illuminate them. This allows us to compute a 3×4 perspective projection matrix up to scale, which is decomposed to find the intrinsic and the extrinsic parameters of the projector. The rendering process uses the same internal and external parameters, so that the projected images are registered with the physical objects.

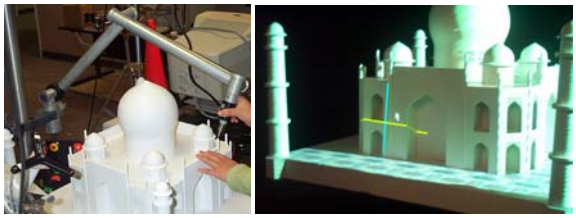


Figure 4.5: *(Left) We use a 3D touch probe scanner to create a 3D model of the real object. (Right) The projectors are calibrated with respect to the model by finding which pixels (center of cross) illuminate the known 3D fiducials.*

4.1.2.2. Intensity Correction

The intensity of the rendered image is modified on a per-pixel basis to take into account the reflectance of the neutral surface, the local orientation and distance with respect to the projector using the intensity equation (see section 4.1.1.1). Since the surface normals used to compute the $1/\cos(\theta_P)$ correction are available only at the vertices in polygonal graphics models, we exploit the rendering pipeline for approximate interpolation. We illuminate a white diffuse version of the graphics model (or a model matching appropriate $k_u(x)$ of the physical model) with a virtual white light placed at the location of the projector lamp and render it with squared distance attenuation. The resultant intensities are smooth across curved surfaces due to shading interpolation and inversely proportional to $(d(x)^2/k_u(x)\cos(\theta_P))$ factor. To use the limited dynamic range of the projectors more efficiently, we do not illuminate surfaces with $\theta_P > 60$ (since $1/\cos(\theta)$ ranges from 2 to infinity). This avoids the low sampling rate of the

projected pixels on oblique surfaces and also minimizes the misregistration artifacts due to any errors in geometric calibration. During the calculations to find the overlap regions (described below), highly oblique surfaces are considered not to be illuminated by that projector.

```
During pre-processing:
    Create 3D graphics model, G, of
    physical object
    Create 3D graphics model, B, of
    background
    Approximately position the projector
    Find perspective pose, P, of the
    projector wrt the physical object

During run-time:
    Get user location, U
    Get animation transformation, T
    Modify G's surface attributes
    Render G using the pose P, and user
    location U
    Transform B using T-1, B'
    Render B' using the pose P, and user
    location U
    Modify image intensity to compensate
    for surface orientation
```

Let is look at the steps in detail. The user's location, U, can be tracked using magnetic or optical tracking technology. To reproduce purely view-independent surface appearance (diffuse reflectance), user location is not required. For view-dependent effects such as specular highlights, approximate user location is necessary. We may assume the user is at a sweet-spot and need not track the user. The projector projection matrix, P, is obtained using an off-line calibration process similar to the technique used for finding internal and external parameters of a camera [Fau93]. We take a set of fiducials with known 3D locations on the physical object and find the corresponding projector pixels that illuminate them. This allows us to compute a 3x4 perspective projection matrix up to scale, which is decomposed to find the internal and the external parameters of the projector. The rendering process uses the same internal and external parameters, so that the projected images are registered with the physical objects. During run-time, instead of the object, G, the background, B, is transformed to create the apparent motion. At each frame, an intensity correction stage pre-multiplies the projected image with intensity weights that compensate for the local surface orientation. Otherwise, surfaces normal to the incident light will appear brighter than surfaces illuminated obliquely due to the cosine fall-off.

4.1.2.3. Complete illumination of complex 3D shapes

For complete illumination that avoids shadows due to self-occlusion, using additional projectors is an obvious choice. This leads to the more difficult problem of seamlessly merging images from multiple projectors. A naïve solution may involve letting only a single projector illuminate any given surface patch. But, there are two main issues when dealing with overlapping CRT, LCD or DLP projectors, which compel the use of feathering (or cross-fading) of intensities. The first is the lack of color equivalence between neighboring projectors [Maj00], due to manufacturing process and temperature color drift during their use. The second is our desire to minimize the sensitivity to small errors in the estimated geometric calibration parameters or mechanical variations.

Feathering is commonly used to generate seamless panoramic photomosaics by combining several views from a single location [Sze97]. Similar techniques are exploited in multi-projector wide-field-of-view displays [Pan, Ras99], and two-dimensional arrays of flat projections. In such cases, the overlap region is typically a (well-defined) contiguous region on the display surface as well as in each projector's frame buffer. In the algorithm used in [Sze97, Ras99] the intensity of a pixel is weighted proportional to the Euclidean distance to the nearest boundary (zero contribution) pixel of the (projected) image. The per-pixel weights are in the range [0, 1]. They are multiplied to the pixel intensities in the final rendered image. The pixels weights near the boundary of a source image are near zero and the pixels contribute very little, so that there is a smooth transition to the next source image. This leads to the commonly seen intensity roll-off as shown in Figure 4.6(a). Under ideal conditions and assuming color equivalence, the weight contribution of both projectors $A+B$ adds up to 1. Even when projector B's color response is different than that of A (say, attenuated—shown as B'), the resultant $A+B'$ (shown in blue) transitions smoothly in the overlap region.

This weight assignment strategy works well only when the target image illuminates a smooth continuous surface at and around the overlap. In our case, the physical model is usually made up of non-convex objects or a collection of disjoint objects resulting in shadows, fragmented overlap

regions and, more importantly, overlap regions containing surfaces with depth discontinuities, as shown in Figure 4.6(c) with a simple occluder. Now, with unequal color response, the resultant weight distribution $A+B'$ has offending sharp changes, e.g. at points f and g. This situation is analogous to image-based rendering (IBR), where warping a single depth-enhanced image creates dis-occlusion artifacts. When multiple source images are warped to the target image, the color assigned to a pixel needs to be derived (from either a single image where they overwrite each other or) as a weighted combination of corresponding pixels from source images. The feathering, which actually blurs the result, is usually necessary to overcome (minor) color difference in corresponding pixels in input images and to hide ghosting effects (due to small mis-registration errors). One of the few solutions to this is proposed by [Deb98], in which they scale the intensities by weights proportional to the angles between the target view and the source views. As mentioned in their paper, "it does not guarantee that the weights will transition smoothly across surfaces of the scene. As a result, seams can appear in the renderings where neighboring polygons are rendered with very different combinations of images.

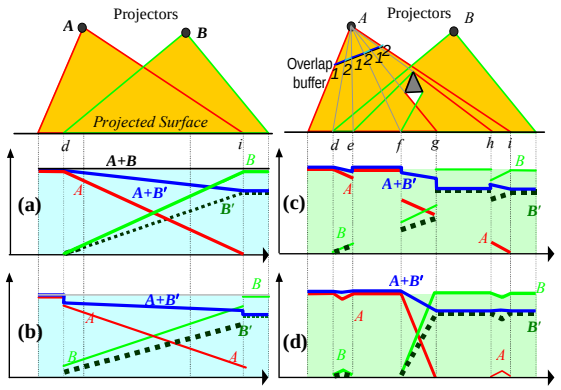


Figure 4.6: *Intensity weights using feathering methods. The plots show the contribution of projectors A, B and B′ and the resultant accumulation A+B and A+B′ along the lit planar surface. Our technique, shown in (d), creates smooth weight transitions.*

The plots in Figure 4.6(b) show a sample weighting scheme based on a similar idea and the corresponding problems. Below, we present a global solution using a new feathering algorithm that suits IBR as well as shader lamps. The algorithm is based on the following guidelines:

1. The sum of the intensity weights of the corresponding projector pixels is one so that the intensities are normalized;
2. The weights for pixels of a projector along a physical surface change smoothly in and near overlaps so that the inter-projector color differences do not create visible discontinuity in displayed images; and
3. The distribution of intensity weights for a projector within its framebuffer is smooth so that small errors in calibration or mechanical variations do not result in sharp edges.

In practice, it is easier to achieve (or maintain) precise geometric calibration than to ensure color equality among a set of projectors over a period of time [Maj00]. This makes condition 2 more important than 3. But, it is not always possible to satisfy condition 2 or 3 (e.g. if the occluder moves closer to the plane so that $f = g$ in figure 4.6) and hence they remain as guidelines rather than rules.

The three guidelines suggest solving the feathering problem, without violating the weight constraints at depth discontinuities and shadow boundaries. Traditional feathering methods use the distance to the nearest boundary pixel to find the weight [Sze97, Ras99]. Instead, we first find pixels corresponding to regions illuminated by a single projector and assign them an intensity weight of 1. Then, for each remaining pixel, the basic idea behind our technique is to find the shortest Euclidean distance to a pixel with weight 1, ignoring paths that cross depth discontinuities. The assigned weight is inversely proportional to this distance. Figure 4.6(d) shows the result of the new feathering algorithm in flatland for two projectors. Even under different color responses, the algorithm generates smooth transitions (see $A+B'$) on the planar surface in presence of shadows and fragmented overlaps. The algorithm can be used for 3 or more projectors without modification.

For a practical implementation, we use two buffers—an overlap buffer and a depth buffer. The depth buffer is updated by rendering the graphics model. The overlap buffer contains integer values to indicate the number of overlapping projectors for each pixel. The overlap regions (i.e. overlap count of two or more) are computed using the traditional shadow-buffer technique. The algorithm follows:

```
At each projector:

    Compute boundaries between regions of
    overlap count 1 and  > 1

    Compute depth discontinuities using
    edge detection in depth buffer
```

```
      For each pixel in overlap region:

         update shortest distance to
         overlap count==1 region ignoring
         paths crossing depth discontinuity

At each projector:

      For each pixel in overlap region

      Find all corresponding pixels in
      other projectors

      Assign weights inversely proportional
      to the shortest distance
```

For some pixels in the overlap region, such as region [*h,i*] for projector *A*, no nearest pixel with overlap count of 1 can be found, and so the shortest distance is set to a large value. This elegantly reduces the weight in isolated regions and also cuts down unnecessary transition zones.

4.1.3. Enhancing moving objects

We can illuminate objects so that the surface textures appear glued to the objects even as they move. In this case, we can display updated specular highlights even for a static viewer. For example, in showroom windows or on exhibition floors, one can show a rotating model of the product in changing colors or with different features enhanced. In an experimental system, a tracked "paintbrush" was used to paint on a tracked moving cuboid held by the user (Figure 4.7). The presence of the physical model allows natural haptic feedback. The need to attach a tracker and dynamic mis-registration due to tracker latency are the two main problems [Ban01].

Figure 4.7: *A tracked "paintbrush" painting on a tracked cuboid.*

4.1.3.1. Vehicle simulation, computer aided engineering and design

Let us consider the problem of creating the perception of motion without corresponding physical displacement in space. This perception of motion is known as *apparent motion*. Here we describe it in the context of simulation of motion for a car.

The motion analysis by humans is divided into three levels of processing: retinal motion sensing, 2D integration and 3D interpretation. The need to sense retinal motion, and analyze it as quickly as possible, places great demands on the visual system. A great deal of perceptual and physiological research has been conducted to discover the properties of these mechanisms. With several (retinal and other) cues, some potentially conflicting, the visual system attempts to integrate into a meaningful 'best' solution, which may be actually incorrect [Mat98]. The resulting illusions provide us an opportunity.

Using 3D computer graphics, we exploit the induced motion effect and the 3D interpretation error for some very constrained cases. For induced motion, we segment a continuous static physical scene into sub-parts illuminated with images that have consistent rate of relative displacement. We also create temporary and invalid 3D interpretation using shadows, lighting and texture movements. We describe a system to show some limited effects on a static car model and present techniques that can be used in similar setups.

The scene is made up of a toy-car model, with a simple horizontal surface and a vertical surface. The horizontal surface coarsely represents the ground surface, either a road or rough terrain. The vertical surface is the backdrop which represents everything else including bushes, trees and sky. The background can be made arbitrarily complex (but must have constant profile along any plane perpendicular to the direction of apparent motion) to improve the visual fidelity.

Figure 4.8: *(Top-left) The diorama of a car and approximate background. (Bottom) A cartoon-like environment created by image projection in which the car appears to move (Top-right) A night-time simulation using virtual headlights.*

4.1.3.2. Motion Effects

The car is simulated as driving along a road, on a rough surface, or in various other environments. To create apparent motion, we illuminate wheels with images of rotating wheels. The images of the background (made up of the backdrop and ground) move in a direction opposite to the intended car movement. In the simplest case, as seen in the video, the car appears to move forward i.e. left to right. To create this effect, the wheels rotate clockwise and the background moves right to left (Figure 4.9(i)). The crucial task for a believable movement is maintaining consistency between the angular wheel movement and corresponding translation of the ground (and the backdrop). For any duration,

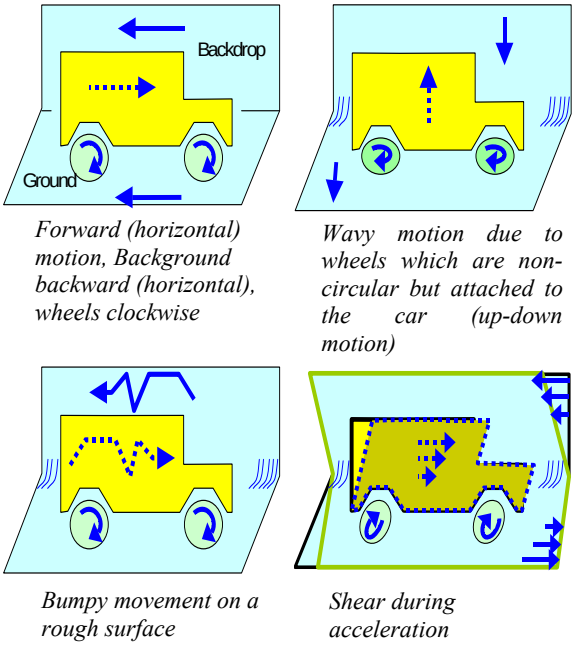
$$\int \text{wheel perimeter arclength} = \left| \text{displacement of the background} \right|$$

This ensures that the wheels are not sliding while the car is in motion. A small amount of motion blur is added to the wheels. Along the direction of the motion, the background geometry with the associated texture maps is infinitely long, and is implemented using a simple sink and a source.

We experimented with many types of cartoon or non-realistic motions. Two important factors that add to the effect are sound and removal of (physical) sharp edges in the background.

4.1.3.3. Wobbly Motion

Slow moving cartoon cars usually have a wavy or bumpy movement resulting in a small periodic or random displacement along the vertical direction. This sometimes is emphasized by non-circular wheels. In our case, the car is static. Hence, to create the desired apparent vertical shift while the car is in motion, we instead translate the background in the opposite (vertical) direction (Figure 4.9(ii)). During rendering, the 3D model of the backdrop as well as the ground is translated vertically. The amount of translation, in the case of non-circular wheels, is determined by distance between the point of contact of the wheels from wheel axis (Figure 4.9(ii)). The distance traveled is again consistent with integration of the arc length of wheel perimeter. Both types of motions can be seen in the video available on our website. (The video does not do justice and it is difficult to feel the transitions, the effect has to be experienced in person with all the three dimensional cues. Please see the subsection on user reactions.)



Forward (horizontal) motion, Background backward (horizontal), wheels clockwise

Wavy motion due to wheels which are non-circular but attached to the car (up-down motion)

Bumpy movement on a rough surface

Shear during acceleration

Figure 4.9: *Types of apparent motion for a car. Rendered motion is shown with blue arrow. The resultant apparent motion is shown with dashed blue arrow.*

4.1.3.4. Lights and Shadows

Shadows provide a very important cue in apparent motion. For example, in a two-dimensional image sequence (i.e. without any other depth cues), a ball moving diagonally upwards across the screen can be made to appear as if it is gradually moving away from the viewer by rendering a shadow that appears to stay with and under the ball. The cue overrides contradictory information regarding the unchanging size of the ball. The strength of the effect does depend on the properties of the shadow region. For example, shadows with a penumbra, and which fall below the object, work best.

We enhance the wobbly motion effect by rendering virtual shadows from directional light source (sun) or local lights (street lights). The change in shadow position creates the illusion that the change is a result of changing vertical distance between the car and the ground. We noticed that, it is not necessary to use the same light position to calculate shading and shadows ! Further, perceptually, the movement of virtual shadows (Figure 4.10) is not affected by the fixed real shadows of the physical car on the background.

For night time simulation, the only cues are headlight beams and shading due to the street lights. The spread of the parabolic projection of the headlights and change in overlap between the two beams indicates the vertical shift. The color of the two beams is intentionally chosen slightly different, so that the lateral shift as the car moves up and down is clearly seen. We also exploit spot lights from the street lights (Figure 4.11).

Figure 4.10: *Vertical displacement by shifting the background and shadows. These two pictures are taken from the same camera position, so the car in both images is at the same location. Note the change in position of the virtual shadows and the parallax for the edge between ground and backdrop.*

Figure 4.11: *Shading due to street lights as spot lights.*

4.1.3.5. Acceleration dependent modifications

Cartoon animators emphasize acceleration, such as a sudden start or a screeching halt, by geometric deformation of the object. For example, a car appears to 'work hard' to move forward while starting when the top of the car is sheared in the direction of the acceleration. Similarly a hard brake and stop is indicated by shearing the top backwards. Since we cannot shear the physical model, we enhance this effect using two tricks.

First, we implement a shear in the background that is opposite of the shear expected in the car (figure 4.9). The shear is along one dimension, along the vertical axis. Hence, for example, during a sudden start, the background shears backward and the shear at a point is proportional to the vertical distance from the center of the car. The points above the vertical center of the car translate backwards while points below translate forward. Without a loss of generality, lets assume that the car center is at the origin, the vertical direction is parallel to the z-axis, and the forward direction is parallel to the x-axis, Then the shear at a given frame is achieved using a simple transformation matrix [1, 0, -*a*, 0; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1]. Where *a* is proportional to the acceleration at that frame.

Since the acceleration is positive during starting, negative during braking, and zero during constant speed and velocity, the same shear equation can be used throughout the animation.

For the second trick, we observe that rapid acceleration also means relative slide between the ground and the wheels. Hence, for example, a sudden brake results in halt in rotation of the wheels, but the background continues to move (Please see the video available on our website).

Figure 4.12: *Shearing of the background during acceleration*

Figure 4.13: *Photo-realistic diorama (left column) vs. cartoon diorama (right column) appearance based on shading and motion.*

4.1.3.6. Tracked Illumination

We synchronize and update the rendering of the objects so that the surface textures appear glued to the objects even as they move. In this case, we rotate the car (along with the background) on a turntable. Keeping the virtual light sources fixed, we see corresponding shading changes.

4.2. Creating Consistent Occlusion

Spatial augmented reality systems share many positive properties of spatial virtual reality systems. These displays provide high resolution, improved consistency of eye accommodation and convergence, little motion sickness potential, and the possibility of an integration into common working and living environments. One of the main challenges for spatial AR systems as well as for head-mounted optical see-through displays is the generation of *correct occlusion effects* between virtual and real objects.

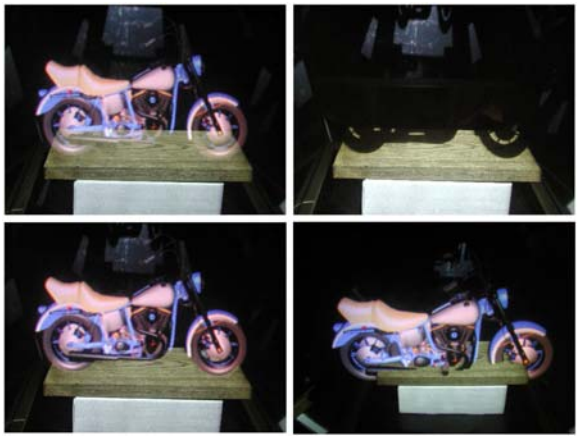


Figure 4.14: *Wrong occlusion effects with normal illumination (upper-left). Occlusion shadow generated with projector-based illumination (upper-right). Realistic occlusion of the real object by the virtual one (lower-left). Knowing depth information of real objects allows the occlusion of virtual objects by real ones (lower-right).*

The basic problem is that in most cases, the existing environment illumination is applied to lighten the physical scenery. However, light that is reflected off the real objects' surface interferes with the optically overlaid graphics – that appear as semi-transparent ghosts which are unrealistically floating in midair (cf. figure 4.14). For indoor situations, the environment illumination can be controlled and synchronized with the rendering process if the simple light bulbs are replaced by video projectors (sometimes referred to as *light projectors*). This concept is called *projector-based illumination*. It requires a physical environment that is initially not illuminated. While this condition can be man-made for many stationary display situations, some setups already provide a dark surrounding by default – like the Virtual Showcase [Bim01].

For optical see-through head-mounted displays, a special optics has been developed by Kiyokawa et al. [Kiy00] that supports *mutual occlusion* – called ELMO. ELMO uses half-silvered mirrors as optical combiners and an additional semi-transparent LCD panel in front of the conventional optics. The LCD panel is used to selectively block the incoming light on a per-pixel basis. This enables virtual objects to occlude real ones. A head-attached depth

sensor allows them to acquire depth maps of the real environment in real time. This makes the occlusion of virtual objects by real ones possible. ELMO faces a number of problems that are linked to the LCD panel: light attenuation caused by the LCD panel, and low response time and resolution of the LCD panel. However, as the first functioning system of its kind, it effectively addresses the occlusion problem of optical see-through head-mounted displays.

Head-Mounted Projective Displays, or HMPDs, (such as described by Hua et al. [Hua01]) require the observer to wear miniature projectors. The projectors beam the synthetic images directly onto the surfaces of the real objects that are within the user's field of view. Since the observer's viewing frustum can be optically matched with the projection frustum, view-dependent rendering is possible while benefiting from a view-independent projection (i.e., depth information for real objects is not required). However, the real objects' surfaces have to be coated with a retro-reflective material in terms of providing stereoscopic rendering, multi-user applications, and the usage of such displays within uncontrolled illuminated environments. The occlusion problem of optical see-through displays is not an issue for HMPDs, since the retro-reflective material avoids the problem of environment light interfering with the graphical overlays.

Video-projectors have been applied to address the occlusion problem for spatial AR configurations: Noda et al. [Nod99], for instance, has presented a stationary optical see-through display that uses a video projector to illuminate real objects selectively – not lighting those areas that are overlaid by graphics. However, view-dependent rendering is not possible in this case. The observer's viewpoint has to match with the center of projection of the video projector since the illumination pattern is rendered from this point using a normal on-axis projection. In this special case no depth information of the real environment is required for a correct rendering. In addition, stereoscopic rendering is not provided. Later, Naemura et al. [Nae02] has proposed an approach that is technically similar to Noda's. The conceptual difference, however, is that he applies a hand-held video projector as a real flashlight to interactively generate shadow effects of virtual objects on real surfaces. He does not address the occlusion problem of optical see-through displays, but focuses on enhancing such interactive mixed reality applications by providing additional visual cues through shadows. As in Noda's case no depth information of the real objects are needed.

In the following sections, general projector-based illumination techniques are described that can be applied in combination with spatial optical see-through AR displays. Off-axis and on-axis situations are treaded in exactly the same way. By using computer-controlled video-projectors as replacements for simple light bulbs, the lighting situation can be fully controlled on a per-pixel basis. This allows producing correct occlusion effects between virtual and real objects by projecting view-dependent shadows – called *occlusion shadows* [Bim02a]- directly onto real objects located behind virtual ones using projector-based illumination (cf. figure 4.14).

4.2.1. Single Users

For rendering occlusion shadows the viewpoints of each user, the intrinsic and extrinsic parameters of each light projector, as well as the virtual and the real scene must be known.

The viewpoints are continuously measured with head-tracking technology, while the projectors' parameters are determined only once during a calibration phase. Virtual objects can be interactively manipulated during runtime.

Knowing the scene and the view transformation lets us compute the perspective projection matrix (V) of the corresponding viewpoint that incorporates the model-view transformation with respect to the scene's origin.

The projectors can be calibrated to a registered geometric representation of the real scene that is registered to its physical counterpart. To do this a semi-manual calibration routine can be applied: The two-dimensional projections of known three-dimensional fiducials can be marked on the projector's image plane. Using these mappings, a numerical minimization method (such as Powell's direction set method [Pre92]) is applied to solve a *perspective N-point problem*. This results in the perspective 4x4 projection matrices P of the projector that incorporates the correct model-view transformation with respect to the scene origin.

If multiple projectors are used, the calibration process has to be repeated for each projector separately.

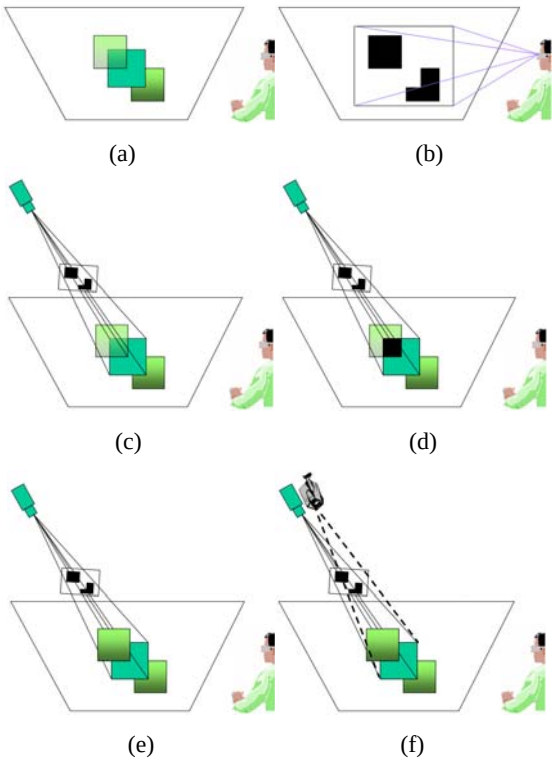


Figure 4.15: *Multi-pass rendering and perspective texture mapping for creating occlusion shadows.*

The basic algorithm below illustrates how to generate occlusion shadows for a single point of view with the aid of *mulit-pass rendering*. The depth information of both – the real and the virtual content have to be known. A *shadow mask* that contains the silhouette of the virtual content is generated in the first pass (cf. figure 4.15a-b) which is then mapped onto the known geometry of the real content in the second pass via *perspective texture mapping* (cf. 4.15c-d). Then the illumination for the real content is rendered into the frame buffer.

For now we want to assume that we project only uniformly colored light onto the real surfaces from the light projector's point of view while virtual objects are illuminated from the positions of the virtual light sources. This illumination, however, could be computed with a more an advanced BRDF model – producing a correct and matching radiance on real and virtual surfaces with respect to virtual light sources. We will describe this in section 4.6.

```
generate shadow mask (first pass):
    set projection matrix to V
    render real content into depth buffer
    render virtual content into stencil
    buffer
    render illumination for real content
    into …
    …frame buffer (previously cleared to
    black)

read-back:
    transfer frame buffer into texture
    memory T

render shadow mask (second pass):
    set projection matrix to P
    set texture matrix to V +
    normalization space correction
    clear frame buffer to black
    render real content into frame buffer
    using …
    …projective texture T
```

Rendering the real content into the depth buffer ensures a correct occlusion of virtual objects by real ones. The normalization space correction consists of a scaling by [0.5,0.5,1.0], followed by a translation of [0.5,0.5,0.5] to map from normalized screen space to normalized texture space*.

Note, that if the graphics card provides a render-to-texture option, the read-back operation from the frame buffer into texture memory can be bypassed.

*This applies for OpenGL-like definitions of the texture and normalized device coordinates.

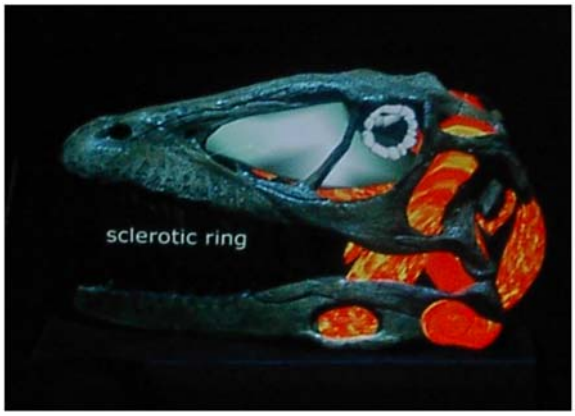


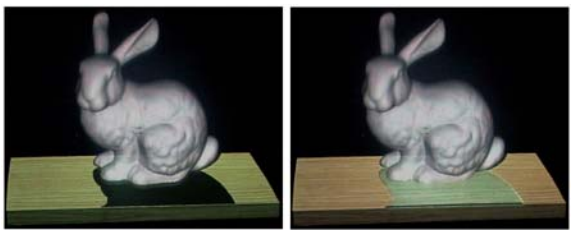
Figure 4.16: *Correct occlusion effects for a complex scene: a real dinosaur with integrated soft-tissue [Bim02b].*

While figure 4.14 illustrates the entire process on a trivial real scene, figure 4.16 shows that the same process is also efficient for complex objects.

4.2.2. Multiple Users

A clear limitation of the basic method described in section 4.2.1 is the following fact: If the same real surfaces are simultaneously visible from multiple points of view (e.g. for different observers), individual occlusion shadows that project onto these surfaces are also visible from different viewpoints at the same time.

Considering two observers, for instance, observer A might be able to see the occlusion shadow that is generated for observer B and vice versa. In addition, the shadows move if the viewers are moving, which might be confusing (cf. figure 4.17a).



(a) (b)

Figure 4.17: *(a) Occlusion shadow of second observer is clearly visible. (b) Wrongly visible occlusion shadow is covered by optically overlaying the corresponding part of the reflectance map.*

One can think of several methods to reduce or avoid these effects:

- *Method I* [Bim02a]: Occlusion shadows generated for other viewpoints are the *umbral hard-shadows* that are cast by the virtual scene with a light source positioned at the other viewpoints' locations. This fact can be made use of by attaching a point light to each viewpoint. This generates correct lighting effects on the virtual scene's surfaces – in addition to matching hard-shadows on the real scene's surfaces.
- *Method II* [Bim02a]: The *interference* between individual occlusion shadows can be *minimized* by ensuring that they are generated only on those real surfaces that are visible from the corresponding viewpoint. However, since the occlusion shadows are finally rendered from the viewpoint of the projector(s), all view-dependent computations (e.g., back-face culling and depth buffering) are done for this perspective – not for the perspectives of the actual viewpoints.
- *Method III* [Bim03]: Each projector is complemented by a video camera that is used to dynamically scan *reflectance* information from the surfaces of the real objects (cf. figure 4.15f). Knowing this reflectance information, however, leads to an effective and general solution: In addition to the virtual scene, the portions of the real scene (i.e., its registered reflectance map) that are covered by the occlusion shadows of all other observers are rendered. If this is done well, a seamless transitions between the real and the virtual portions we can create (cf. figure 4.17b). For each observer the occlusion shadows of all other observers are rendered into the stencil buffer first. This is done by rendering the real scene's geometry from each observer's perspective and adding the corresponding occlusion shadows via projective texture mapping. The stencil buffer has to be filled in such a way that the area surrounding the occlusion shadows will be blanked out in the final image. Then the real scene's reflectance map is rendered into the frame buffer (also from the perspective of the observer) and is shaded under the virtual lighting situation. After stenciling has been disabled, the virtual objects can be added to the observer's view. A possibility of obtaining the reflectance map of diffuse real scenes with projector-camera combinations is described in section 4.3.

Due to *self occlusion*, not all portions of the real content can be lit by a single light projector. A solution to this problem is to increase the number of projectors and place them in such a way that the projected light is distributed over the real content. To guarantee a *uniform*

illumination, however, surfaces should not be lit by more than one projector at the same time or with the same intensity. Otherwise, the projected light accumulates on these surfaces and they appear brighter than others. A *crossfeathering* method can be applied that balances the contribution of each projector equally.

4.3. Creating Consisting Illumination

To achieve a *consistent lighting* situation between real and virtual environments is important for convincing augmented reality (AR) applications.

A rich pallet of algorithms and techniques have been developed that match illumination for video- or image-based augmented reality. However, very little work has been done in this area for optical see-through AR.

Inspired by the pioneering work of Nakamae et al. [Nak01] and –later– Fournier et al. [For93], many researchers have approached to create consistent illumination effects while integrating synthetic objects into a real environment. Most of these approaches represent the real environment in form of images or videos. Consequently, mainly image processing, inverse rendering, inverse global illumination, image-based and photo-realistic rendering techniques are applied to solve this problem. Due to the lack of real-time processing, these approaches are only applicable in combination with desktop screens and an unresponsive* user interaction. Devices that require interactive frame-rates, such as head-tracked personal or spatial displays, cannot be supported. Representative for the large body of literature that exists in this area, several more recent achievements can be highlighted:

Boivin et al. [Boi01] present an interactive and hierarchical algorithm for reflectance recovery from a single image. They assume that the geometric model of the scenery and the lighting conditions within the image are known. Making assumptions about the scene's photometric model, a virtual image is generated with global illumination techniques (i.e., ray-tracing and radiosity). This synthetic image is then compared to the photograph and a photometric error is estimated. If this error is too large, their algorithm will use a more complex BRDF model (step by step – using diffuse, specular, isotropic, and finally anisotropic terms) in the following iterations, until the deviation between synthetic image and photograph is satisfactory. Once the reflectance of the real scenery is recovered, virtual objects can be integrated and the scene must be re-rendered. They report that the analysis and re-rendering of the sample images takes between 30 minutes to several hours – depending on the quality required and the scene's complexity.

Yu et al. [Yu99] present a robust iterative approach that uses global illumination and inverse global illumination techniques. They estimate diffuse and specular reflectance, as well as radiance and irradiance from a sparse set of photographs and the given geometry model of the real scenery. Their method is applied to the insertion of virtual objects, the modification of illumination conditions and to the re-rendering of the scenery from novel viewpoints. As for Boivin's approach, BRDF recovery and re-rendering are not supported at interactive frame-rates.

Loscos et al. [Los00] estimate only the diffuse reflectance from a set of photographs with different but controlled real world illumination conditions. They are able to insert and remove real and virtual objects and shadows, and to modify the lighting conditions. To provide an interactive manipulation of the scenery, they separate the calculation of the direct and indirect illumination. While the direct illumination is computed on a per-pixel basis, indirect illumination is generated with a hierarchical radiosity system that is optimized for dynamic updates [Dre97]. While the reflectance analysis is done during an offline preprocessing step, interactive frame-rates can be achieved during re-rendering. Depending on the performed task and the complexity of the scenery, they report re-rendering times for their examples between 1-3 seconds on a SGI R10000. Although these results are quite remarkable, the update rates are still too low to satisfy the high response requirements of stereoscopic displays that support head-tracking (and possibly multiple users).

Gibson and Murta [Gib00] present another interactive image composition method to merge synthetic objects into a single background photograph of a real environment. A geometric model of the real scenery is also assumed to be known. In contrast to the techniques described above, their approach does not consider global illumination effects to benefit from hardware accelerated multi-pass rendering. Consequently, a reflectance analysis of the real surfaces is not required, indirect illumination effects are ignored, and a modification of the lighting conditions is not possible. The illumination of the real environment is first captured in form of an omni-directional image. Then a series of high dynamic basis radiance maps are pre-computed. They are used during runtime to simulate a matching direct illumination of the synthetic objects using sphere mapping. Shadow casting between real and virtual objects is approximated with standard shadow mapping. With their method, convincing images can be rendered at frame rates

* Not real-time.

up to 10fps on an SGI Onyx 2. However, it is restricted to a static viewpoint.

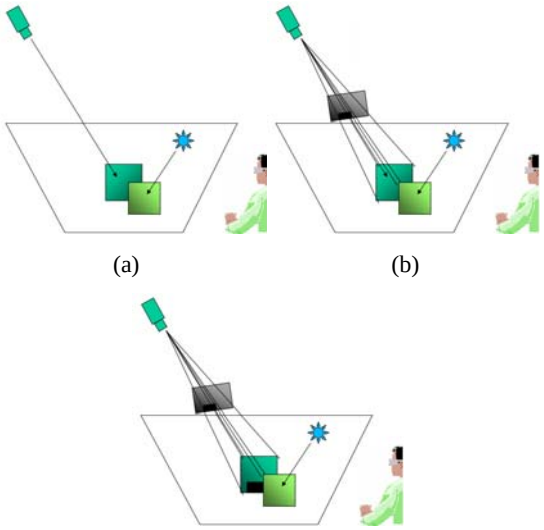


Figure 4.18: *Multi-pass rendering and perspective texture mapping for creating occlusion shadows.*

The main problem of a consistent illumination for optical see-through approaches is that the real environment is illuminated by physical light sources while the virtual environment is illuminated based on synthetic light sources (cf. figure 4.18a). This results in inconsistent shading and shadow effects unless the virtual light sources approximate the properties (such as position, direction, intensities, color, etc.) of the physical ones. This, however, is a very inflexible. In contrast to video see-through, the pixel appearance of the real environment in the video image cannot be modified to achieve a matching illumination. Consequently, the physical contribution of the real light sources has to be neutralized to illuminate the entire environment based on the virtual lighting situation (cf. figures 4.18a and 4.18c). This is only possible with controllable physical light sources, such as video projectors.

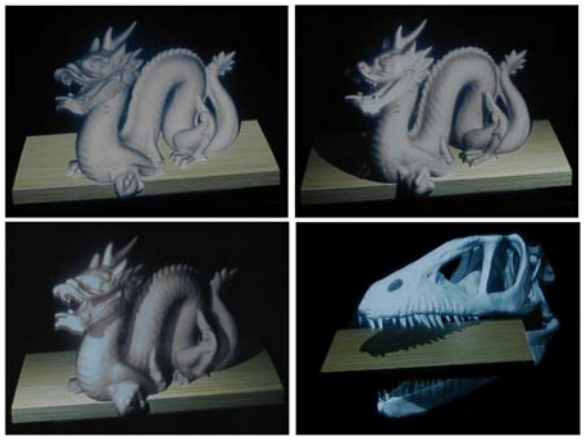


Figure 4.19: *Unrealistic illumination with correct occlusion (upper-left). Realistic illumination under varying virtual lighting conditions with matching shading and shadows (upper-right, lower-left and -right).*

This section describes methods which create a consistent illumination between real and virtual components within an optical see-through environment [Bim03]. Combinations of video projectors and cameras can be applied to capture *reflectance information* from diffuse real objects and to illuminate them under new *synthetic lighting* conditions (cf. figure 4.19). For diffuse objects, the capturing process can also benefit from *hardware acceleration* – supporting dynamic update rates. To handle *indirect lighting* effects (like *color bleeding*) an off-line *radiosity* procedure is outlined that consists of multiple rendering passes. For *direct lighting* effects (such as simple shading, shadows and reflections) hardware accelerated techniques are described which allow to achieve interactive frame rates. The reflectance information can be used in addition to solve multi-user occlusion problem discussed in section 4.2.

4.3.1. Diffuse Reflectance Analysis

As in section 4.2, we want to assume that the geometry of both object types –real and virtual– has been modeled or scanned in advance. While the material properties of virtual objects are also defined during their modeling process, the diffuse reflectance of physical objects is captured on- or off-line with a set of video projectors and cameras and a structured light analysis. This sort of analysis is standard practice for many range scanner setups. But since only diffuse real objects can be considered (a projector-based illumination will generally fail for any specular surface), a

simple diffuse reflectance analysis can benefit from hardware accelerated rendering techniques. In contrast to conventional scanning approaches, this leads to dynamic update rates. Figure 4.20 illustrates an example.

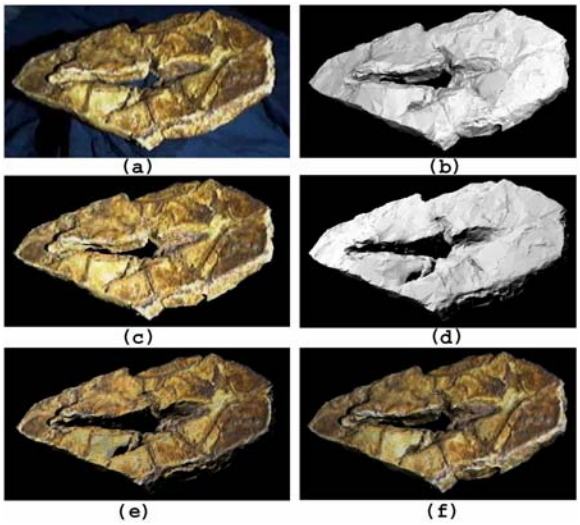


Figure 4.20: *(a) Captured radiance map of a fossilized dinosaur footprint; (b) Intensity image rendered for calibrated projector from (a); (c) Computed reflectance map; (d) Novel illumination situation; (e) Reflectance map under novel illumination from (d); (f) Reflectance map under virtual illumination from (b).*

Again, the intrinsic and extrinsic parameters of projectors and cameras within the world coordinate system have to be estimated first. Each device has to be calibrated separately. As described in section 4.2, two-dimensional projections of known three-dimensional fiducials can be interactively marked on a projector's/camera's image plane. Using these mappings, a minimization method is used to solve a perspective n-point problem for each device. This results in the perspective projection matrices P, C of a projector and a camera. Both matrices incorporate the correct model-view transformations with respect to the origin of the world coordinate system.

Once calibrated, a projector/camera combination can be used to perform the diffuse reflectance analysis as follows:

4.3.1.1. Capturing a Radiance Map

The video projector is used to send structured light samples to the diffuse physical object and illuminate it with a predefined color C_p and an estimated intensity η. Synchronized to the illumination, the video camera captures an input image. Since this image contains the diffuse reflectance of the object's surface under known lighting conditions it represents a *radiance map*. *White-balancing* and other *dynamic correction functions* have been disabled in advance. The parameters of the camera's *response function* can be adjusted manually in such a way that the recorded images approximate the real world situation as close as possible.

Some types of video projectors (such as digital light projectors, DLPs) display a single image within sequential, time-multiplexed light intervals to achieve different intensity levels per color. If such projectors are used, a single snapshot of the illuminated scene would capture only a slice of the entire display period. Consequently, this image would contain incomplete color fragments instead of a full-color image. The width of this slice depends on the exposure time of the camera. To overcome this problem, and to be independent of the camera's *exposure* capabilities, we capture a sequence of images over a predefined period of time. These images are then combined and averaged to create the final diffuse radiance map I_{rad} (cf. figure 4.20a).

4.3.1.2. Creating an Intensity Image

To extract the diffuse material reflectance out of I_{rad} the lighting conditions that have been created by the projector have to be neutralized. OpenGL's diffuse lighting component, for instance, is given by [Nei96]:

$$I_i = \frac{1}{r_i^2}\cos(\theta_i)(D_l D_m)_i$$

where I_i is the final intensity (color) of a vertex i, D_l is the diffuse color of the light, D_m is the diffuse material property, the angle θ_i is spanned by the vertex's normal and the direction vector to the light source, and the factor $1/r_j^2$ represents a square distance attenuation.

Similar as described in section 4.1, an intensity image I_{int} that contains only the diffuse illumination can be created by rendering the object's geometry (with $D_m = 1$) from the perspective of the video camera, illuminated by a

point light source (with $D_l = C_p \eta$) that is virtually located at the position of the projector (cf. figure 4.20b).

In addition, hard shadows can be added to the intensity image by applying standard shadow mapping techniques. Consequently, the background pixels of I_{int}, as well as pixels of regions that are occluded from the perspective of the light source are blanked out ($I_{\text{int}}(x,y) = 0$), while all other pixels are shaded under consideration of the above diffuse lighting equation.

4.3.1.3. Extracting and Re-Rendering Diffuse Reflectance

Given the captured radiance map I_{rad} and the rendered intensity image I_{int}, the diffuse reflectance for each surface that is visible to the camera can be computed by:

$$I_{ref}(x,y) = \frac{I_{rad}(x,y)}{I_{\text{int}}(x,y)} \quad \text{for} \quad I_{\text{int}}(x,y) > 0,$$

$$I_{ref}(x,y) = 0 \quad \text{for} \quad I_{\text{int}}(x,y) = 0$$

Note that the division of the two images can be implemented by a pixel shader, and can consequently be executed in real-time on the graphics hardware.

The reflectance image I_{ref} is stored, together with the matrix C and the real object's world-transformation O_c that is active during the capturing process within the same data-structure. This data structure is referred to as *reflectance map* (cf. figure 4.20c).

The captured reflectance map can be re-rendered together with the real object's geometric representation from any perspective with an arbitrary world-transformation O_a. Thereby, I_{ref} is applied as projective texture map with the texture matrix* set to $O_a^{-1} O_c C$. Enabling texture modulation, it can then be re-lit virtually under novel illumination conditions (cf. figures 4.20d, 4.20e and 4.20f).

The basic diffuse reflectance analysis method as described above faces the following problems:

- due to under-sampling, surfaces which span a large angle ϕ_i between their normal vectors and the direction vectors to the camera can cause texture artifacts if I_{ref} is re-mapped from a different perspective;
- a single reflectance map covers only the surface portion that is visible from the perspective of the camera;
- the radiance map can contain indirect illumination effects caused by light fractions that are diffused off other surfaces (so-called *secondary scattering*). The intensity image I_{int}, however, does not contain secondary scattering effects since a global illumination solution is not computed. Consequently, the extracted reflectance is incorrect at those areas that are indirectly illuminated by secondary scattering;
- the projector intensity η has to be estimated correctly;

To overcome the under-sampling problem, the definition can be made that only surfaces with $\phi_i \leq \phi_{\max}$ are analyzed. All other surfaces will be blanked-out in I_{ref} (i.e., $I_{ref}(x,y) = 0$). Experiments have shown that $\phi_{\max} = 60°$ is an appropriate value.

Multiple reflectance maps that cover different surface portions can be captured under varying transformations O_c or C. They are merged and *alpha blended* during re-mapping them via *multi-texturing* onto the object's geometric representation. This ensures that regions which are blanked out in one reflectance map can be covered by other reflectance maps. To generate seamless transitions between the different texture maps, *bi- or tri-linear texture filtering* can be enabled.

Illuminating the entire scene can cause an extreme secondary scattering of the light. To minimize the appearance of secondary scattering in I_{rad}, the scene can be divided into discrete pieces and their reflectance can be captured one after the other. For this, the same algorithm as described above can be applied. The difference, however, is that only one piece at a time is illuminated and rendered which then appears in I_{rad} and I_{int}. By evaluating the blanked out background information provided in I_{int}, the selected piece we can be effectively segmented in I_{int} and

* Including the corresponding mapping transformation from normalized device coordinates to normalized texture coordinates.

its reflectance can be computed. This is repeated for each front-facing piece, until I_{ref} is complete.

The projector's intensity η can be estimated as follows: First, a reflectance map is generated with an initial guess of η. This reflectance map is then re-mapped onto the object's geometric representation, which is rendered from the perspective of the camera and illuminated by a virtual point light source with η located at the projector. The rendered radiance map I_{radv} is then compared to the captured radiance map I_{rad} by determining the average square distance error Δ among all corresponding pixels. Finally, an approximation for η can be found by minimizing the error function f_Δ. For this, Brent's inverse parabolic minimization method with bracketing [Bre73] can be applied as one possibility. By estimating η, the constant black-level of the projector can also be incorporated.

4.3.2. Augmenting Radiance

In computer graphics, the *radiosity method* [Gor84] is used to approximate a *global illumination solution* by solving an *energy-flow* equation. Indirect illumination effects, such as *secondary scattering* can be simulated with *radiosity*. The general radiosity equation for *n* surface patches is given by:

$$B_i = E_i + \rho_i \sum_{j=1}^{n} B_j F_{ij}$$

where B_i is the radiance of surface i, E_i is the emitted energy per unit area of surface i, ρ_i is the reflectance of surface i, and F_{ij} represents the fraction of energy that is exchanged between surface i and surface j (the *form-factor*).

The simulation of radiosity effects within an optical see-through environment that consists of diffuse physical and virtual objects, is facing the following challenges and problems:

- light energy has to flow between all surfaces – real ones and virtual ones;
- physical objects are illuminated with physical light sources (i.e., video projectors in our case) which do not share the geometric and radiometric properties of the virtual light sources;
- no physical light energy flows from virtual objects to real ones (and vice versa). Consequently, the illuminated physical environment causes (due to the absence of the virtual objects) a different radiometric behavior than the entire environment (i.e., real and virtual objects together).

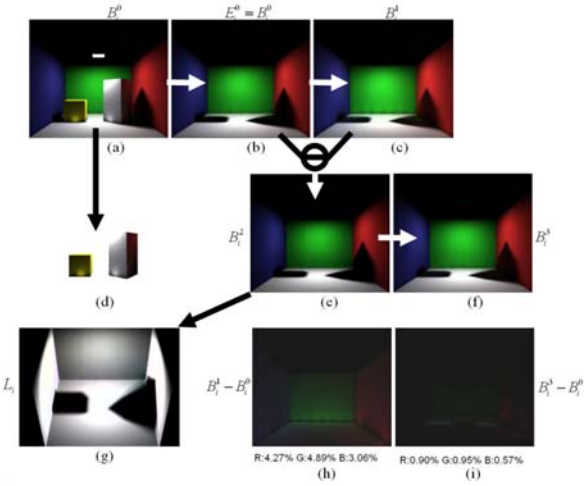


Figure 4.21: *Multi-Pass radiosity for augmenting synthetic radiance onto a real environment.*

An example is illustrated in figure 4.21a*. The entire environment consists of three walls, a floor, two boxes and a surface light source on the ceiling. We want to assume that the walls and the floor are the geometric representations of the physical environment, and the boxes as well as the light source belong to the virtual environment. While the diffuse reflectance ρ_i of the physical environment can be automatically captured (as described in section 4.3.1), it has to be defined for the virtual environment. After a radiosity simulation† of the entire environment the radiance values B_i^0 for all surfaces have been computed‡. *Color-bleeding* and *shadow-casting* are clearly visible.

For virtual objects, the computed radiance values are already correct (cf. figure 4.21d). The rendered image represents a radiance map that is generated from one

* A physical mock-up of the Cornell room has been chosen since it is used in many other examples as a reference to evaluate radiosity techniques.

† A hemi-cube-based radiosity implementation has been applied with progressive refinement, adaptive subdivision and interpolated rendering for our simulations.

‡ Note, that the upper index represents the radiosity pass.

specific perspective. Rendering the virtual objects from multiple perspectives results in multiple radiance maps that can be merged and alpha blended during re-mapping them via multi-texturing onto the virtual geometry (as described for reflectance maps in section 4.3.1). In this case, our radiance maps are equivalent to light maps that are often being applied during pre-lighting steps to speed up the online rendering process.

The pre-lit virtual objects can simply be rendered together with their light maps and can be optically overlaid over the physical environment.

The physical surfaces, however, have to emit the energy that was computed in B_i^0 (cf. figure 4.21b). To approximate this, it can be assumed first that every physical surface patch directly emits an energy E_i^0 that is equivalent to B_i^0. If this is the case, fractions of this energy will radiate to other surfaces and illuminate them in addition. This can be simulated by a second radiosity-pass (cf. figure 4.21c), which computes new reflectance values B_i^1 for all the physical surfaces, by assuming that $E_i^0 = B_i^0$, and not considering the direct influence of the virtual light source.

If we subtract the radiance values that have been computed in both passes we receive the scattered light only. That is, the light energy radiated between the physical surfaces $B_i^1 - B_i^0$ (cf. figure 4.21h).

Consequently,

$$B_i^2 = B_i^0 - \left(B_i^1 - B_i^0\right)$$

approximates the energy that has to be created physically on every real surface patch. To prove this a third radiosity pass can be applied to simulate the energy flow between the patches (cf. figure 4.21f). It can bee see that the remaining energy $B_i^1 - B_i^0$ will be nearly added, and we receive:

$$B_i^3 = B_i^2 + \left(B_i^1 - B_i^0\right) \approx B_i^0$$

By removing the virtual objects from the environment and simulating the second radiosity pass, light energy will also be radiated onto surfaces which were originally blocked or covered by the virtual objects (either completely or partially). Examples are the shadow areas that have been cast by the virtual objects. This can be observed in figure 4.21h and figure 4.21i. Consequently, negative radiance values are possible for such areas. To avoid this, the resulting values have to be clipped to a valid range.

The average deviations between B_i^0 and B_i^1, as well as between B_i^0 and B_i^3, within the three spectral samples red (R), green (G), and blue (B) are presented below figures 4.21h and 4.21i, respectively. Treating a video projector as a point light source B_i^2 can be expressed as follows:

$$B_i^2 = \rho_i L_i F_i$$

where L_i is the irradiance that has to be projected onto surface i by the projector, and F_i is the form-factor for surface i, which is given by:

$$F_i = \frac{\cos(\theta_i)}{r_i^2} h_i$$

where θ_i is the angle between a surface patch's normal and the direction vector to the projector, r_i is the distance between a surface patch and the projector, and h_i is the visibility term of the surface patch, seen from the projector's perspective.

Extending and solving the above equations for L_i, we receive (cf. figure 4.21g):

$$L_i = \frac{B_i^2}{\rho_i F_i} \eta$$

To cope with the individual brightness of a video projector, the intensity factor η can be added. How to estimate η for a specific projector was described in section 4.3.1. To be consistent with our previously used terminology, we call L_i the *irradiance map*.

The computed radiance and irradiance values are view-independent. Consequently, irradiance maps for the real objects and radiance maps for the virtual objects can be pre-computed offline.

The real objects are illuminated with projected light during runtime by rendering the generated irradiance map from the viewpoint of the projector (e.g., as illustrated in figure 4.21g). Virtual objects are rendered with the computed light maps (e.g., as illustrated in figure 4.21d) and are then optically overlaid over the real environment. Due to the view-independence of the method, the augmented scene can be observed from any perspective (i.e., head-tracking and stereoscopic rendering are possible). However, the scene has to remain static, since any modification would require to re-compute new radiance and irradiance maps throughout multiple radiosity passes. This is not yet possible at interactive rates.

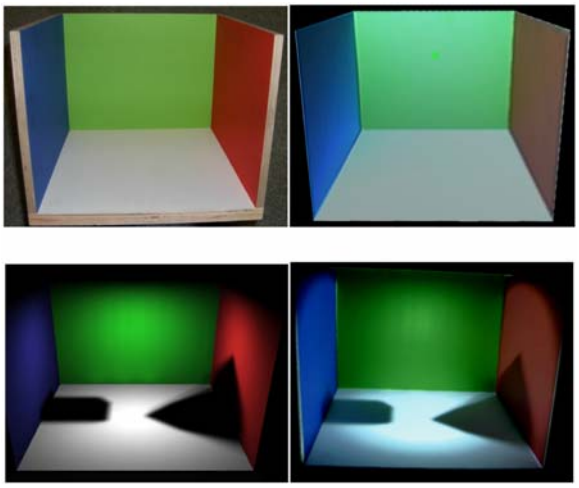


Figure 4.22: *(a) Photograph of original object under room illumination; (b) Screen-shot of captured reflectance re-lit with virtual point light source and Phong shading; (c) Screen-shot of simulated radiosity solution with captured reflectance, virtual surface light source (shown in figure 4.21), and two virtual objects (show in figure 4.21); (d) Photograph of original object illuminated with the computed irradiance.*

Figure 4.22 shows a photograph of (a) the physical object under room illumination, (b) a screen-shot of captured reflectance maps that have been re-rendered under novel lighting conditions, (c) a screen-shot of the simulated radiance situation B_i^0, and (d) a photograph of a physical object that has been illuminated with L_i. Note, that small deviations between the images can be contributed to the responds of the digital camera that was used to take the photograph, as well as to the high black-level of the projector that, for instance, makes it impossible to create completely black shadow areas.

4.3.3. Interactive Approximations: Shading and Shadows

In the following sections several *interactive rendering* methods are described that make use of *hardware acceleration*. In particular how to create matching shading, shadow and reflection effects on real and virtual objects are discussed. Indirect lighting effects such as color-bleeding, however, cannot be created with these techniques. Yet, they create acceptable results at *interactive frame rates* for multiple head-tracked users and stereoscopic viewing on conventional PCs.

The generation of *direct illumination* effects on virtual surfaces caused by virtual light sources is a standard task of today's hardware accelerated computer graphics technology. Real-time algorithms, such as Gouraud shading or Phong shading are often implemented on graphics boards.

Consistent and matching shading effects on real surfaces from virtual light sources can be achieved by using video projectors that project appropriate irradiance maps onto the real objects. In section 4.1 it is shown how to compute an irradiance map to lift the radiance properties of neutral diffuse objects with uniform white surfaces into a pre-computed radiance map of a virtual scene illuminated by virtual light sources. An irradiance map that creates virtual illumination effects on diffuse real objects with *arbitrary reflectance properties* (color and texture) can be computed as follows:

First, the real objects' captured reflectance map (I_{ref}) is rendered from the viewpoint of the projector and is shaded with all virtual light sources in the scene. This results in the radiance map I_{rad_1}. Then I_{ref} is rendered again from the viewpoint of the projector. This time, however, it is illuminated by a single point light source (with $D_l = 1 \cdot \eta$) which is located at the position of the projector. This results in the radiance map I_{rad_2}. Finally, the correct irradiance map is computed by:

$$L = \frac{I_{rad_1}}{I_{rad_2}}$$

Note that in general this method correlates to the method described in section 4.3.2. The difference is the applied illumination model. While in section 4.3.2 an indirect global illumination model (radiosity) is used, here a hardware accelerated direct models (such as Phong or Gouraud shading) is applied. It is easy to see that I_{rad_1} is the opponent to B_i^2 and that I_{rad_2} corresponds to $\rho_i F \eta$.

Note also that this method is actually completely independent of the real objects' reflectance. This can be shown by equalizing I_{rad_1} with I_{rad_2}. In this case the diffuse material property D_m (i.e., the reflectance) is canceled out. Consequently, I_{rad_1} and I_{rad_2} can be rendered with a constant (but equal) reflectance (D_m). If $D_m = 1$ is chosen then the irradiance map is simply the

quotient between the two intensity images $I_{\text{int_1}}$ and $I_{\text{int_2}}$ that result from the two different lighting conditions – the virtual one and the real one.

The irradiance map L should also contain consistent shadow information. How to achieve this is outlined below. Figure 4.19 illustrates examples with matching shading effects*.

Six types of shadows can be identified within an optical see-through environment:

- shadows on real objects created by real objects and real light sources;
- shadows on virtual objects created by virtual objects and virtual light sources;
- shadows on virtual objects created by real objects and virtual light sources;
- shadows on real objects created by real objects and virtual light sources;
- shadows on real objects created by virtual objects and virtual light sources;
- occlusion shadows;

The first type of shadow is the result of occlusions and self-occlusions of the physical environment that is illuminated by a physical light source (e.g., a video projector). Since it is focused on controlling the illumination conditions within the entire environment via virtual light sources these shadows have to be removed. This can be achieved by using multiple synchronized projectors that are able to illuminate all visible real surfaces.

The second and third shadow types can be created with standard *shadow mapping* or *shadow buffering* techniques. To cast shadows from real objects onto virtual ones, the registered geometric representations of the real objects have to be rendered together with the virtual objects when the shadow map is created (i.e., during the first shadow pass). Such geometric real world representations (sometimes called *phantoms* [Bre96]) are often rendered continuously to generate a realistic occlusion of virtual objects by real ones. Note that these hardware accelerated techniques create hard shadows while global illumination methods (such as radiosity) can create soft shadows. Texture blending, however, allows ambient light to be added to the shadow regions. This results in dark shadow regions that are blended with the underlying surface texture, instead of creating unrealistic black shadows.

Shadow types number 4 and 5 can also be created via shadow mapping. However, they are projected on the surface of the real object together with the irradiance map L, as discussed in section 4.1. Therefore, I_{rad_1} has to contain the black (non-blended) shadows of the virtual and the real objects. This is achieved by rendering all virtual objects and all phantoms during the first shadow pass to create a shadow map. During the second pass the shaded reflectance texture and the generated shadow texture are blended and mapped onto the objects' phantoms. A division of the black shadow regions by I_{rad_2} preserves these regions. Note that a blending of the projected shadows with the texture of the real objects occurs physically if the corresponding surface portions are illuminated (e.g., by a relatively small amount of projected ambient light).

Occlusion shadows have been described in section 4.2. They are special view-dependent shadows created by the projectors on the real objects' surfaces to achieve a realistic occlusion of real objects by virtual ones within optical see-through augmented environments. They are normally not visible from the perspective of the observer, since they are displayed exactly underneath the graphical overlays. Occlusion shadow-maps, however, have also to be blended to the irradiance map L before it is projected.

The entire process can be summarized in form of a three-pass rendering algorithm:

```
create an intensity image I_rad_2 of the
real object (first pass):
    render real object from perspective
    of light projector…
    …having a white diffuse material…
    …illuminated by a white virtual point
    light source…
    …located at the projector

create a shading image I_rad_1 of real and
virtual objects (second pass):
    generate shadow map for real and
    virtual objects
```

* Note that a simple wooden plate has been chosen to demonstrate and to compare the different effects. However, all techniques that are explained in these sections can be applied to arbitrary object shapes.

render real objects from perspective of light projector…

…having a white diffuse material…

…illuminated by the virtual light sources…

…with shadow mapping enabled

compute irradiance image $L = \frac{I_{rad_1}}{I_{rad_2}}$

render occlusion shadows from perspective of projector…

…and blend with L (third pass)

Note that the implementation of the image division to compute the irradiance map L can be a pixel shader, to achieve real-time performance.

4.3.4. Interactive Approximations: Reflections

Using hardware accelerated *cube mapping* techniques, the virtual representation of the real environment (i.e., the objects' geometry together with the correctly illuminated reflectance map) can be reflected by virtual objects (cf. figure 4.23).

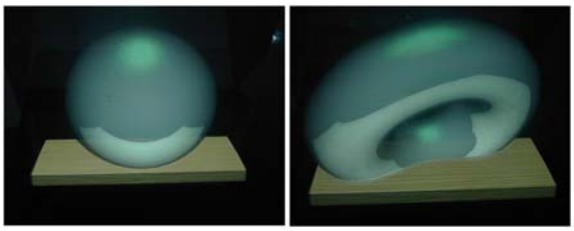


Figure 4.23: *Virtual objects reflecting and occluding the real object (wooden plate).*

Therefore, only the registered virtual representation of the real environment has to be rendered during the generation step of the cube map. Virtual objects are then simply rendered with cube mapping enabled. Note, that for conventional cube mapping, reflection effects on a virtual object are physically correct for only a single point – the center of the cube map frusta. To create convincing approximations this center has to be matched with the virtual object's center of gravity, and the cube map has to be updated every time the scene changes.

4.4. Augmenting Optical Holograms

A hologram is a *photometric emulsion* that records *interference patterns* of *coherent light*. The recording itself stores *amplitude*, *wavelength* and *phase* information of light waves. In contrast to simple photographs (which can record only amplitude and wavelength information), holograms have the ability to reconstruct complete *optical wavefronts*. This results in a three-dimensional appearance of the captured scenery, which is observable from different perspectives.

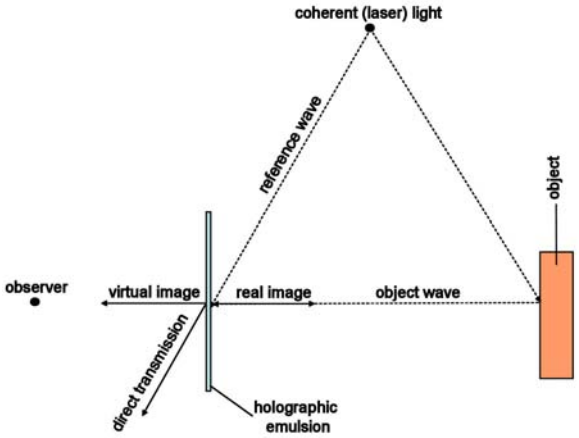


Figure 4.24: *Recording and reconstruction of optical holograms.*

Optical holograms are reconstructed by illuminating them with *monochromatic* (purity of color) light. Thereby the light has to hit the emulsion at the same angle as the reference laser beam that was used to record the hologram.

Figure 4.24 illustrates the basics of optical holographic recording and reconstruction: A laser beam is split into two identical beams. While one beam (called *reference wave*) illuminates the holographic emulsion directly, the other beam illuminates the object to be recorded. The light that is reflected from the object (called the *object wave*) shines on the emulsion and creates the fringe pattern together with the reference wave.

If the emulsion is illuminated with a copy of the reference wave, it interacts with the recorded *interference fringes* and reconstructs the object wave which is visible to the observer. The amplitude of the reconstructed object wave is proportional to the intensity of the reference wave. Its reconstruction can be controlled with a selective illumination.

There are two basic types of optical holograms: *transmission* and *reflection holograms*.

A transmission hologram is viewed with the light source and the observer on opposite sides of the holographic plate. The light is transmitted through the plate before it reaches the observer's eyes. Portions of the emulsion that are not recorded or not illuminated remain transparent.

Reflection holograms are viewed with the light source and the observer on the same side of the holographic plate. The light is reflected from the plate towards the eyes of the observer. As for transmission holograms, not recorded or not illuminated portions of the emulsion remain transparent (without opaque backing layer).

Among these two basic types of holograms, a large pallet of different variations exists. While some holograms can only be reconstructed with laser light, others can be viewed under white light. They are called *white-light holograms*.

One of the most popular white-light transmission holograms are *rainbow holograms*. With rainbow holograms, each wavelength of the light is diffracted through a different angle. This allows observing the recorded scene from different horizontal viewing positions, but also makes the scene appear in different colors when observed from different viewing positions.

In contrast to rainbow holograms, *white-light reflection holograms* can provide full parallax and display the recorded scene in a consistent –but in most cases monochrome– color for different viewing positions.

Color white-light holograms (both: transmission and reflection types) can also be produced. Usually the same content is recorded on several emulsion layers. However, each layer is exposed to laser light with a different wavelength. When reconstructed, each object wave from each layer contributes with its individual wavelength. Together, they merge into a colored image.

Today, many applications for optical holograms exist. Examples include interferometry, copy protections, data storage and holographic optical elements.

Due to their unique capability to present three-dimensional objects to many observers with almost no loss in visual quality, optical holograms are often being used in museums. They can display artefacts that are not physically present without the need to build real replicas of the originals.

In addition, medical, dental, archaeological and other holographic records can be made – both for teaching and for documentation.

Optical holograms, however, are static and lack in interactivity. Exceptions are *multiplex holograms* that are built from multiple narrow (vertical) strip-holograms that contain recordings of the same scenery at different time intervals. While moving around the hologram (or spinning a cylindrically-shaped version around its principle axis), observers can perceive the recorded scene in motion. Yet, multiplex holograms are not interactive.

Three-dimensional computer graphics in combination with stereoscopic presentation techniques represents an alternative that allows interactivity. State of the art rendering methods and graphics hardware can produce realistically looking images at interactive rates. However, they do not nearly reach the quality and realism of holographic recordings. Autostereoscopic displays allow for a glass-free observation of computer-generated scenes. Several autostereoscopic displays exist that can present multiple perspective views at a time – thus supporting multiple users simultaneously. Resolution and rendering speed, however, decreases with the number of generated views. Holographic images, in contrast, can provide all depth-cues (perspective, binocular disparity, motion parallax, convergence and accommodation) and can be viewed from a theoretically unlimited number of perspectives at the same time.

Parallax displays are display screens (e.g., CRT or LCD displays) that are overlaid with an array of light-directing or light-blocking elements. Using these elements, the emitted light is directed to both eyes differently – allowing them to see individual portions of the displayed image. The observer's visual system interprets corresponding light rays to be emitted by the same spatial point. Dividing the screen space into left-right image portions allows for a glass free separation of stereo pairs into two or more viewing zones.

Some displays control the parallax array mechanically or electronically depending on the viewpoint of the observer to direct the viewing zones more precisely towards the eyes. Others generate many dense viewing zones – each of them showing a slightly different perspective of the rendered scene at the same time. Such displays support multiple users simultaneously, but do not yet allow to reach high frame rates.

Examples of parallax displays are *parallax barrier displays* that apply an array of light-blocking elements (e.g., a light blocking film or liquid crystal barriers) in front of a screen. The light-blocking elements are used to cover

portions of the screen for one eye that are visible from the other eye.

Other examples are *lenticular sheet displays* that utilize refraction of a lens array (e.g., small cylindrical, prism-like or spherical lenses) to direct the light into the different viewing-zones. Images that are generated with lenticular sheet displays appear brighter than the ones displayed on barrier displays. While prisms and cylinders provide a horizontal parallax only, spherical lenses support a full parallax.

Several research groups are working on computer generated holograms. There are two main types: *digital holography* and *electro-holography*.

In digital holography [Yam90], holographic printers are used to sequentially expose small fractions of the photometric emulsion with a computer generated image. Conventional holograms which display a computer-generated content are the result. This technique can also be used for the construction of large-scale, tiled holograms [Klu02]. Although digital holograms can be multiplexed to display scenes in motion, they remain non-interactive.

Electro-holography aims at the computer-based generation and display of holograms in real time [Kol89, Luc97]. Holographic fringes can be computed by rendering multiple perspective images that are combined into a stereogram [Luc95], or by simulating the optical interference and calculating the interference pattern [Luc93]. Once computed, the fringes are dynamically visualized with a holographic display. Since a massive amount of data has to be processed, transmitted and stored to create holograms, today's computer technology still sets the limits of electro-holography. To overcome some of the performance issues, advanced reduction and compression methods have been developed. This results in electro-holograms that are interactive, but small, low-resolution and pure in color. Recent advances on consumer graphics hardware may reveal potential acceleration possibilities [Pet03].

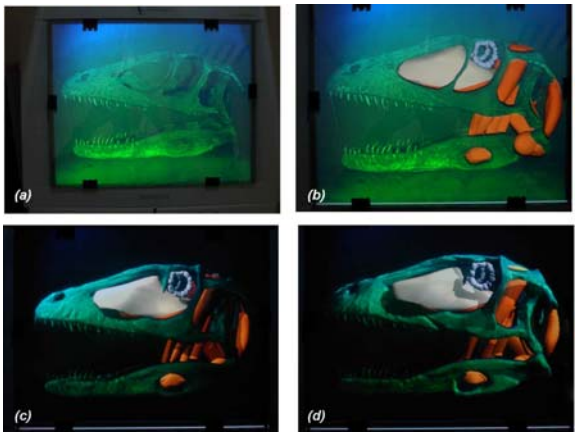


Figure 4.25: *A rainbow hologram of a dinosaur skull combined with three dimensional graphical elements and synthetic occlusion and shading effects. (a) The hologram only. (b) Optically integrated graphical elements (muscles and other soft-tissue). (c and d) Consistent illumination effects between holographic and graphical content.*

Combining optical holograms with 2D or 3D graphical elements can be an acceptable trade-off between quality and interactivity (cf. figure 4.25). While the holographic content provides a high quality but remains static, additional graphical information can be generated, inserted, modified and animated at interactive rates.

Technically, optical combiners such as mirror beam combiners or semi-transparent screens (see chapter 3) can be used to visually overlay the output rendered on a screen over a holographic plate. In this case, however, the reconstructed light of the hologram will interfere with the overlaid light of the rendered graphics and an effective combination is impossible. However, the holographic emulsion can be used as optical combiner itself, since it is transparent if not illuminated in the correct way. Consequently, it provides see-through capabilities.

This section describes how holograms can be optically combined with interactive, stereoscopic or autostereoscopic computer graphics [Bim04a]. While section 4.4.1 explains how correct occlusion effects between holographic and graphical content can be achieved, section 4.4.2 outlines how a consistent illumination is created for such chases.

4.4.1. Partially Reconstructing Wavefronts

A key solution to this problem is to reconstruct the object wave only partially – not at those places where graphical elements have been inserted.

This requires a point light source capable of selectively emitting light in different directions – creating an "incomplete" reference wave. Conventional video projectors are such light sources. They are also well suited for viewing white-light reflection or transmission holograms, since today's high-intensity discharge (HDI) lamps can produce a very bright light.

If autostereoscopic displays, such as parallax displays are used to render 3D graphics registered to the hologram, then both holographic and graphical content appear three-dimensional within the same space. This is also the case if stereoscopic displays with special glasses that separate the stereo images are applied.

Reflection holograms without opaque backing layer and transmission holograms both remain transparent if not illuminated. Thus they can serve as optical combiners themselves – leading to very compact displays. The illumination and rendering techniques that are described in these sections are the same for both hologram types.

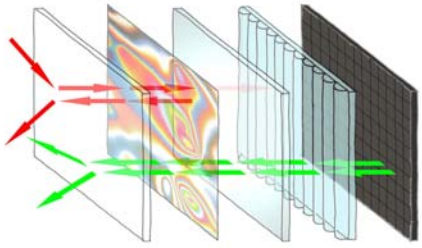


Figure 4.26: *An explosion model of the stacked structure of optical layers. From left to right: glass protection, holographic emulsion, mirror beam combiner (only for transmission holograms), lenticular lens sheet, LCD array. The light-rays (red arrows) are reflected and reconstruct the object wave on their way back through the emulsion. The stereoscopic images (green arrows) pass through all layers until they are merged with the hologram.*

Figure 4.26 illustrates an example of how a transmission hologram can be combined effectively with a flat-panel lenticular lens sheet display (a variation of a parallax display that utilizes refraction of a lens array to direct the light into the different viewing-zones).

Placing a transmission hologram in front of a mirror beam combiner allows to illuminate it from the front and to augment it with graphics from the back. For reflection holograms, this beam combiner is not necessary.

A thin glass plate protects the emulsion from being damaged and keeps it flat to prevent optical distortion. The lenticular lens sheet directs the light emitted from the LCD array through all layers towards the eyes of the observer. The projected light is transmitted through the first two layers, and is partially reflected back (either by the beam combiner in combination with a transmission hologram or by a reflection hologram) – reconstructing the recorded content. The remaining portion of light that is transmitted through all layers is mostly absorbed by the screen.

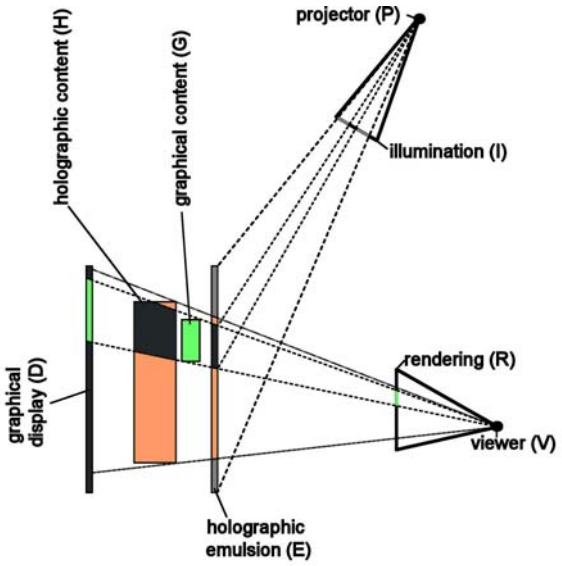


Figure 4.27: *A conceptual sketch of the display constellation. The colored areas on the graphical display and on the holographic emulsion illustrate which portion of the visible image is hologram (red) and which is graphics (green).*

Figure 4.27 illustrates how the selective illumination on the holographic plate is computed to reconstruct the portion of the hologram that is not occluded by graphics.

The following outlines how rendering and illumination can be realized with conventional graphics hardware. It is assumed that depth information of the holographic content (H), as well as a scene description of the graphical content (G) are available. Both contents are geometrically aligned during an offline registered step. If optical markers are recorded in the hologram together with the actual content,

cameras can be used to perform this registration automatically. In addition, the extrinsic and intrinsic parameters of the video projector (P) with respect to the holographic emulsion (E) have to be known. They are also determined during an offline calibration step. If it is mechanically possible to mount the holographic emulsion close to the graphical display (i.e., the distance between E and D is small), then E and D can be approximated to be identical.

First, an intermediate texture image (T) is created from V over E by rendering H into the graphics card's depth buffer and filling the card's frame buffer entirely with predefined light color values. In addition, G is rendered into depth and stencil buffers. The stencilled areas in the frame buffer are cleared in black and the result is copied into the memory block allocated for T.

Note that if a render-to-texture option is provided by the graphics card, the read-back operation from frame buffer into the texture memory is not necessary. The final illumination image (I) is rendered from P by drawing E into the frame buffer and texturing E's geometry with T. The illumination image (I) is beamed onto the holographic emulsion (E) with the projector (P).

Second, a rendering image (R) is generated from V over D (off-axis) by rendering H into the depth buffer, and G into depth and frame buffers. The rendering image (R) is displayed on the graphical display (D).

This can be summarizes with the following algorithm:

```
create intermediate texture T from V over
E (first pass):
     render H into depth buffer
     fill frame buffer with light color
     render G into depth and stencil
     buffer
     fill stenciled areas in frame buffer
     with black

render E from P textured with T and
display on projector (second pass)

create rendering image R from V over D
(optical inlay):
     render H into depth buffer
     render G into depth buffer and frame
     buffer
     display G on projector
```

Figure 4.25a shows a photograph of the entire reconstructed hologram while it is illuminated with a projected uniform light.

By applying the presented techniques, illumination and stereoscopic images are generated in such a way that graphical and holographic content can be merged within the same space (cf. figure 4.25b).

4.4.2. Modifying Amplitude Information

The amplitude of the reconstructed object wave is proportional to the intensity of the reference wave. Beside using an incomplete reference wave for reconstructing a fraction of the hologram, intensity variations of the projected light allow to locally modify the recorded object-wave's amplitude.

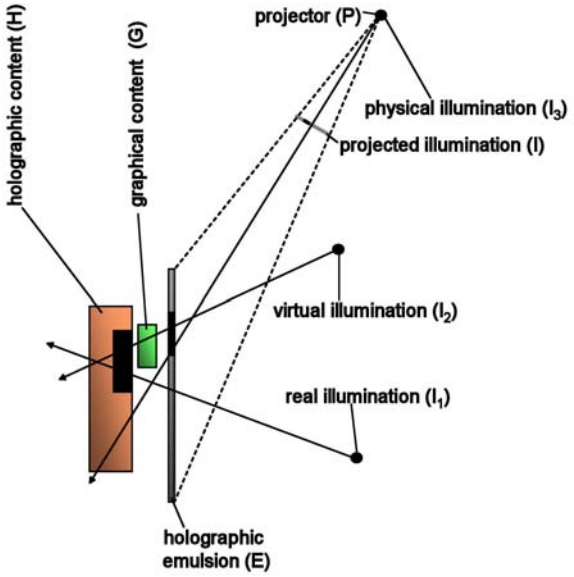


Figure 4.28: *Light-interaction between hologram and graphics: To simulate virtual shading and shadow effects on the holographic content, the recorded and the physical illumination effects have to be neutralized.*

Practically this means that for creating the illumination image (I), shading and shadowing techniques are used to render the holographic content, instead of rendering it with a uniform intensity.

To do this, the shading effects caused by the real light sources that were used for illumination while the hologram was recorded, as well as the physical lighting effects caused by the video projector on the holographic plate have to be neutralized. Then the influence of a synthetic illumination has to be simulated. This can also be done with conventional graphics hardware (cf. figure 4.28). Three intensity images have to be rendered:

For the first image (I_1), H is rendered from V over E with a white diffuse material factor and graphical light sources that generate approximately the same shading and shadow effects on H as the real light sources that were used during the holographic recording process. This results in the intermediate texture T_1. I_1 is generated by rendering E from P and texturing it with T_1. It simulates the intensity of the recorded object wave. The same process is repeated to create the second image (I_2) – but this time graphical light sources are used to shade H under the new, virtual lighting situation. The ratio I_2/I_1 represents the required intensity of the reference wave at holographic plate E.

For the third image (I_3), E is rendered from P with a white diffuse material factor and a virtual point light source located at the projector's position. This intensity image represents the geometric relationship between the video projector as a physical point light source and the holographic plate:

$$F = \frac{\cos(\theta)}{r^2}$$

It contains form factor components, such as the square distance attenuation (r^2) and the angular correlation ($\cos(\theta)$) of the projected light onto the holographic plate and allows neutralizing the physical effects of the projector itself.

The final illumination image (I) can be computed in real time with $I=I_2/I_1/I_3$ via pixel shaders. The projection of I onto E will neutralize the physical and the recorded illumination effects as good as possible, and will create new shadings and shadows based on the virtual illumination. Note that as described previously the graphical content has to be stencilled out in I before displaying it.

During all illumination and rendering steps, hardware-accelerated shadow mapping techniques are used to simulate real and virtual shadow effects on H and on G. Finally, synthetic shadows can be cast correctly from all elements (holographic and graphical) onto all other elements.

The following algorithm summarized this approach:

```
create intensity image I1 (first pass):
     render H from V over E (white diffuse
     factor)…
     …and graphical light sources that
     simulate real…
     …shading on H → T1
     render E from P textured with T1

create intensity image I2 (second pass):
     render H from V over E (white diffuse
     factor)…
     …and graphical light sources that
     simulate virtual…
     …shading on H → T2
     render E from P textured with T1

create intensity image I3 (third pass):
     render E from P (white diffuse
     factor)…
     …and graphical point light source
     attached to P

create and display illumination image I
from P (fourth pass):
     I=I2/I1/I3  (via pixel shader)
```

The capabilities of this technique are clearly limited. It produces acceptable results if the recorded scenery has been illuminated well while making the hologram. Recorded shadows and extreme shading differences cannot be neutralized. Furthermore, recorded color, reflections and higher-order optical effects cannot be cancelled out either.

Projecting an intensity image that contains new shading and shadow effects instead of a uniform illumination allows neutralizing most of the diffuse shading that is recorded in the hologram and produced by the projector. The holographic and graphical content can then be consistently illuminated (creating matching shading and shadow effects) under a novel lighting.

Figures 4.25c and 4.25d illustrate the synthetic shading effects that are caused by a virtual light source. In addition,

virtual shadows are cast correctly between hologram and graphical elements. Although figure 4.25 illustrates the results with a monochrome transmission hologram, the same effects are achieved with reflection holograms and color holograms.

Since all the discussed rendering techniques (like shadow mapping and shading) are supported by hardware-accelerated consumer graphics cards, interactive frame rates are easily achieved.

Using the concept described in this section a pallet of different display variations can be developed. With only minor changes of the presented techniques, for example, arbitrarily curved shapes (such as cylindrical shapes being used for multiplex holograms) can be supported instead of simple planar plates. Even without graphical augmentations, the projector-based illumination alone holds several potentials. In combination with optical or digital holograms it can be used to create visual effects. Certain portions of a hologram, for instance, can be made temporarily invisible while others can be highlighted. Emerging large-scale autostereoscopic displays and existing stereoscopic projection screens allow to up-scale the proposed concept. Not only the display, but also the holograms can be composed from multiple smaller tiles to reach large dimensions and high resolutions.

4.5. Superimposing Pictorial Artwork with Projected Imagery

A seamless and space efficient way for integrating visual information directly into *pictorial artwork* is to use the artwork itself as *information display* [Bim04b]. It can serve as diffuse projection screen and conventional video projectors can be applied to display computer graphics together with the painted content (cf. figure 4.29).

The main difference of this approach to all the methods that are described above is, that an arbitrarily textured surface has to be augmented with colored information. To perceive the projected imagery in the correct colors and intensities, however, requires that the influence of the underlying physical *color pigments* is neutralized. In most situations, this is not possible if untreated images are simply projected directly onto arbitrary colored surfaces. The problem is that the projected light interacts with the color pigments on the canvas and is partially absorbed if the pigment's color isn't fully white.

Figure 4.29: *Results of color correction process with a single projector on real drawing: (a) Real drawing (64x48cm) under environment light. (b) Output image emitted onto drawing. (c) Partially augmented drawing. (d) Output image on a white piece of paper. (e-h) Close-ups. While the upper body part coincides in drawing and painting, Michelangelo modified the lower body part. The arrows indicate the displaced knee and belly sections. They point at the same spot on the drawing.*

A solution to this problem is provided by a new film material which has two properties: first, it is completely transparent and second, it diffuses a fraction of the light that is projected onto it. The film consists of an even deposition of fine particles on both sides of a polyester base with no visible artifacts. It was used for the creation of special effects in Hollywood movies, such as Minority Report (20th Century Fox, 2002) and Paycheck (Paramount Pictures, 2003), and sells for $350 per square foot. Initial measurements have revealed that in average *20%* (*+/- 1%*)

of the light that strikes the film is diffused while the remaining fraction is transmitted towards the canvas (with or without direct contact). This *0.1mm* thin *transparent film* can be seamlessly overlaid over the canvas by integrating it into the frame that holds the artwork. Off-the-shelf 1100 ANSI lumen XGA digital light projectors have been used to display images on film and canvas.

4.5.1. Technical Approach and Mathematical Model

If a light beam with incident radiance *L* is projected onto the transparent film material that is located on top of the original artwork, a portion *d* of *L* is directly diffused from the film while the remaining portion *t* of *L* is transmitted through the film. The transmitted light *tL* interacts with the underlying pigment's diffuse reflectance *M* on the canvas, and a color blended light fraction *tLM* is diffused. The portion *tLMt* is then transmitted through the film, while the remaining part *tLMd* is reflected back towards the canvas where it is color blended and diffused from the same pigment again. This ping-pong effect between film material and canvas is repeated infinitely while for every pass a continuously decreasing amount of light is transmitted through the film that contributes to the resulting radiance *R*. Mathematically, this can be expressed as an infinite geometric series that converges towards a finite value. The same is true for the environment light with incident radiance *E* that is emitted from uncontrollable light sources. Since these light sources also illuminate the canvas and the film material, the environment light's contribution to *R* has to be considered as well.

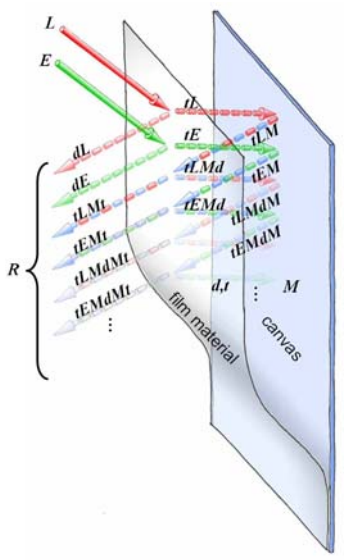


Figure 4.30: *Interaction of projected light and environment light with the canvas and the film material (sequence diagram).*

Figure 4.30 describes this process in form of a sequence diagram. Note that in contrast to this conceptual illustration, there is no physical gap between film material and canvas, and that the light interaction occurs at the same spot.

If all parameters (*L*, *E*, *M*, *t*, and *d*) are known we can compute the resulting radiance *R* that is visible to an observer in front of the canvas:

$$R = (L+E)d + (L+E)t^2 M \sum_{i=0}^{\infty} (Md)^i =$$

$$(L+E)\left(d + \frac{t^2 M}{1-Md}\right)$$

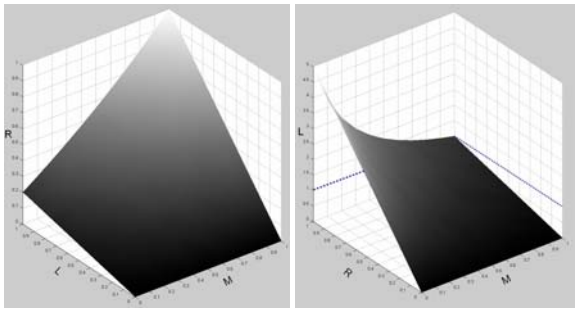
Since *R* forms the image we expect to see is known, we need to solve the above equation for *L*:

$$L = \frac{R}{\left(d + \frac{t^2 M}{1-Md}\right)} - E$$

This allows computing the incident radiance *L* that needs to be projected onto the film and the canvas to create the known result *R*. The radiant intensity *I* of the projector to create *L* is related to a discretized pixel value and is given by:

$$I = L\frac{r^2}{\cos\alpha}s$$

where $r^2/\cos\alpha$ are the form factor components: square distance attenuation and angular correlation of the projected light onto the canvas. The additional factor s allows scaling the intensity to avoid clipping and to consider the simultaneous contributions of multiple projectors.



(a) (b)

Figure 4.31: *Visualization of equation R (a) and equation L (b) over R, L, and M (without E, t=80% and d=20%).*

This approach has clear limitations, which are illustrated in figures 4.31a and 4.31b. Not all radiances R can be produced under every condition. If M is dark, most of L and E are absorbed. In an extreme case, the corresponding pigment is black (M=0). In this case the right term of the equation that computes R is canceled out. The remaining left term –which depends on the diffusion factor d of the film material– sets the boundaries of the final result that can be produced. The intersection of the surface with the RL-plane in figure 4.31a illustrates these limitations. Consequently, in the worst case of this example only 20% of R can be generated. This situation is also reflected in figure 4.31b as the intersection of the surface with the LR-plane. Here we want to assume that $sr^2/\cos\alpha=1$, which results in $L=I$. For a single video beamer, the projected radiance L and consequently the radiant intensity I cannot exceed the normalized intensity value of 1 (dotted line). But for creating most of the resulting radiance values, L and I must be larger. This situation worsens for $r^2/\cos\alpha>1$ and for $E\rightarrow 1$ or $M\rightarrow 0$.

However, the contributions of multiple (n) projectors allow displacing this boundary with:

$$L = \sum_{i=1}^{n} L_i \quad , \quad I_i = L_i \frac{r_i^2}{\cos\alpha_i} s_i$$

If all projectors provide linear transfer functions (e.g., after gamma correction) and identical brightness, $s_i=1/n$ balances the load among them equally. However, s_i might be decreased further to avoid clipping and to adapt for differently aged bulbs.

For both illustrations in figures 4.31a and 4.31b, the environment light E is not considered and is set to zero. Additional environment light would simply shift the surface in figure 4.31a up on the R-axis, and the surface in figure 4.31b down on the L-axis. Note that the above discussed mathematical model has to be applied to all color channels (e.g., red, green, and blue for projected graphical images) separately.

4.5.2. Real-Time Color Correction

The equations described in section 4.5.1 can be implemented as a pixel shader to support a color correction in real time. Figure 4.32 illustrates the rendering process based on an example of Michelangelo's masterpiece *Creation of Adam*. Although we will use this example to explain the process, it is universal and can be applied with arbitrary background images.

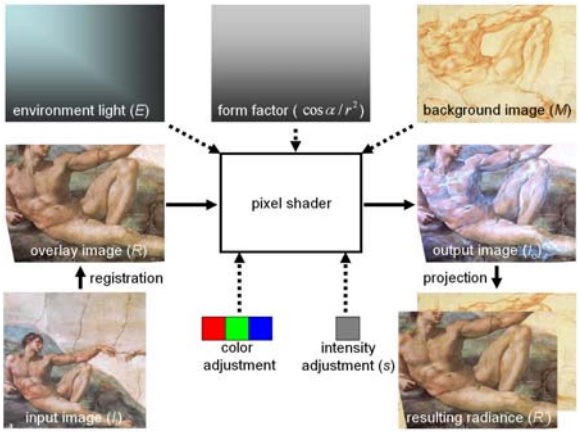


Figure 4.32: *Real-time color-correction process with pixel shader.*

In this example, a copy of an early sanguine sketch of the Adam scene (now being displayed in the London British Museum) serves as real background image M. Our goal is to overlay it entirely with a registered photograph R of the actual ceiling fresco painted the Sistine Chapel.

The first step of the rendering process is to create an input image I_i, which needs to be overlaid. This image can be either dynamically rendered (e.g., as part of a real-time animation or an interactive experience), it can be played back (e.g., frames of a prerecorded movie) or it can be static (e.g., a photograph of the corresponding ceiling portion – as it is the case in our example above).

The input image has to be registered to the physical drawing. Registration is achieved by texture mapping I_i onto a pre-distorted image geometry that is precisely aligned with the physical drawing. The amount of distortion depends on the geometric relation between the video projector and the canvas. It reaches from simple *keystone deformations* to more complex *curvilinear warping* effects (e.g., if *lens distortion* of the projector has to be neutralized). Several automatic approaches have been described that apply video cameras and image analysis to align multiple images of tiled projection screens [Che00, Yan01]. A structured light registration benefits from controllable features, such as projected grid edges or Gaussian matchpoints that can be easily detected in the camera views with a sub-pixel precision. In the case described above, however, a digital representation of the artistic content has to be registered against its physical representation on the canvas, rather than registering one projected structured light image against another one. To detect non-structured artistic features, such as fine lines, in the artwork and register them automatically against the corresponding features in the digital content represents a no-trivial task of computer vision – especially if projected pixels and physical pigments have to be aligned very precisely on the canvas. One can be critical about the feasibility and precision of an automatic method for this problem. I can be solved with a manual registration process that benefits from the resolution of the human visual system. Since the following steps have to be performed only once, they represent an acceptable solution.

An off-line registration process allows to interactively identify 2D correspondences between artistic features in the background image *M* within the image space –that is an image of the drawing displayed on a control screen– and the physical drawing on the wall. This is done using a 2D input device – such as a conventional mouse whose pointer is visible on the control screen and as projection on the canvas. A dual output graphics card and an additional video signal splitter are used to drive the control screen and one or two projectors. The result is a set of 2D vertex fiducials with their corresponding texture coordinates within the image space. The fiducials are Delauny triangulated and the texture coordinates are used to map the correct image portions of *I* onto the image geometry. This results in the overlay image *R*. It has to be stressed, that a precise correspondence between *R* and *M* is important to achieve qualitatively good results. The measurement of 50 to 70 fiducials proved to be sufficient for medium sized canvases. In contrast to *uniform grid methods* normally applied for projector alignment, this general geometric registration allows correlating arbitrary features in the physical drawing with the corresponding pixels of *M* in the image space. Thus, it provides an accurate matching which can be regionally improved further if linear interpolation within single grid triangles fails to be precise enough. The registration results do not change if projector and background image are fixed. Before *R* is rendered the color-correction pixel-shader has to be enabled. Five parameters are passed to it to ensure that equations discussed in section 4.5.1 can be computed.

The first parameter is the *environment light E* in form of intensity texture that has the same size as *R*. It contains intensity values that represent the uncontrollable lighting situation on the canvas. The intensity values can be determined by measuring the irradiance of the environment light with a lightmeter for a discrete number of sample spots on the canvas' surface – resulting in the lux values *E'*. To be processed by the shader, these values have to be normalized to an intensity space that ranges from 0 to 1. To do this, the same spots are measured again, but this time the highest intensity possible (i.e., a white image) is projected onto the lightmeter which is measured in addition to the environment light. These measurements are equivalent to the total irradiance $T'=L'+E'$, and also carry the unit lux. Since we know that $L'=T'-E'$ is equivalent to the scaled intensity value $\cos\alpha/r^2$, we can convert the measured radiance of the environment light from lux into the normalized intensity space with $E=E'/(T'-E')\ \cos\alpha/r^2$. To approximate the intensity values for the entire image area in *E*, all the measured spots are mapped into the image space, are Delauny triangulated and the values for the remaining pixels are linearly interpolated by the graphics pipeline. The values for the remaining pixels are linearly interpolated by the graphics card while rendering the Delauny mesh. Note that *E* is constant if the environment light does not change. For the reasons that are described below, we can assume that $\cos\alpha/r^2$ is constant and equals 1.

The second parameter is the *form factor* that represents the geometric relation between the video projector as a point light source and the canvas. Since it does not change for a fixed relation between projector and canvas, it can be precomputed and passed to the pixel shader in form of an intensity texture with the same dimensions as *E* and *R*. Similar as for the techniques described in the previous sections this texture can be produced by the graphics pipeline: A geometric model of the canvas is rendered with a white diffuse reflectance from the viewpoint of the

projector. Attaching a virtual point light source (also with a white diffuse light component) to the position of the projector and enabling square distance attenuation produces intensities that are proportional to $\cos\alpha/r^2$. The required reciprocal can be computed by the pixel shader. Practically (i.e., for normal-sized canvases and non-extreme projector configurations), the form factor can be assumed to be constant over all pixels. It is then contained by the intensity adjustment parameter *s*.

The third parameter is the *background image M*. It also has the same dimensions as *E* and *R*. This image can be generated by, for example, scanning the color values or taking a photograph of the original drawing under uniform illumination.

The fourth and fifth parameters contain *color and intensity adjustment values* that allow fine-tuning the video projector's individual color response and prevent from intensity clipping. They also allow adopting for color drifts that can be introduced during capturing the background image and allow considering the contributions of multiple projectors. Note that gamma correction has to be applied in advance. This regards projectors with non-linear transfer functions as well as projectors with linear transfer functions that apply a de-gamma mapping on the video input. Gamma correction is usually supported by the graphics hardware and the video driver, but can also be carried out by the pixel shader. These values are adjusted manually, but the support of automated methods for color-matching multiple projectors [Maj00] is imaginable.

The output image I_o is the final result of this rendering process and will be displayed by the video projector. If projected geometrically correct onto the drawing, the result *R'* will be visible. Both images, *R* and *R'* are mostly identical, except for slight artifacts that are due to the limitations discussed above. Figure 4.29 shows the result of our example projected onto the real drawing with a single video projector.

Apparently, the underlying drawing can be made partially or completely invisible to display the graphical overlay in the correct colors on top of it. Close-ups are illustrated in figures 4.29e-h, in which diverging body parts (such as belly and knee) are overdrawn and displaced by the projection.

Figure 4.33: *Results of color correction process with two projectors: (top).The limited intensity capabilities of a single projector result in visible artifacts. (bottom) The contribution of a second projector reduces these effects.*

Some intensities and colors that are required to neutralize the underlying color pigments cannot be achieved by a single video projector. The worst case is to turn a black pigment on the canvas into a white color spot. Figures 4.31a and 4.31b illustrate that in such a case the required intensity can easily exceed the boundary of 1 in our normalized intensity space. The pixel shader clips these values to 1 which results in visible artifacts.

The simultaneous contributions of multiple projectors can reduce or even eliminate these effects. Figure 4.33 shows the extreme case of an input image that has no geometric correspondences to the underlying background image. In addition, it approaches to create bright colors (the sky) on top of dark color pigments on the canvas. In figure 4.33a, a single projector is used. Intensity values that are too large are clipped and result in visible artifacts. Balancing the intensity load between two projectors reduces these artifacts clearly (cf. figure 4.33b).

Due to hardware acceleration of today's graphics cards, the color correction process can be easily performed in real time. Note that none of the photographs in this article have been retouched. Slight variations of color and brightness are due to different camera responds.

As discussed above, the color correction method has limitations that are mainly defined by the capabilities of the applied hardware. For example, the restricted *resolution*,

brightness, *contrast*, *minimum focus distance* and *black-level* of video projectors are issues that will certainly be improved by future generations. The XGA resolution and the brightness of 1100 ANSI lumen of the low-cost projectors that were used to create the results for this section was appropriate for small and medium sized paintings in a normally lit environment. An up-scaling is possible by using more projectors, but downscaling would either result in a loss of effective resolution or in focus problems.

Black, for instance, is a color that cannot be projected. Instead the environment light together with the *black level* of the projectors illuminates areas that need to appear black. However, the human vision system adjusts well to *local contrast effects* – which makes these areas appear much darker than they actually are. Even with little environment light, the high black-level of video projectors causes this illusion to fail in extreme situations, such as the one shown in figure 4.34. The development of video projectors indicates that a decrease of the black-level and an increase of the contrast ratio can be expected in future.

(a) (b)

(c) (d)

Figure 4.34: *Rembrandt's self-portrait: (a) copy of original painting as it looks today (illuminated under environment light), (b)–(d) various cleaning stages to remove the overpainted layers form 1935(d), 1950(c) and 1980(b) are projected onto (a). Only black and white photographs of these stages are available. The high black-level of the video projectors prevents from creating a totally black color on the canvas. Extreme regions, such as overlaid hair and hat cannot appear completely black for this reason.*

Light can damage the artwork. Especially ultra violet (UV) and infrared (IR) radiation produced by the lamps of video projectors is critical. Commercially available UV/IR blocking filters can be mounted in front of the projectors' lenses to remove most of these unwanted rays while transmitting visible wavelengths. For the remaining visible light portion a rule of thumb advises to illuminate valuable and delicate pieces permanently with no more than 100lx-150lx. The potential damage caused by light is cumulative (e.g. 1 hour with 1000lx equals 1000 hour with 1lx) and depends on the material and color of the painting and the wavelength (i.e. the color) of the light. A temporary illumination of a higher light intensity is not critical. During a 2-3 minute presentation, an increased lighting is only temporary and such highlight situations usually only appear (if at all) for a short period of time (e.g., a few seconds) at varying locations on the canvas. Nevertheless, using the intensity adjustment described above, the maximum light level can be constrained to be below an upper threshold. However, this might cause visible artifacts depending on the presented content, the paining, and the environment light (as described in figure 4.31). Thus it is important to reach a good balance between total illumination (projected light and environment light) over time and convincing presentation effects.

The method described above currently considers only the intensity of the environment light *E*. This is adequate for regular white light sources but will result in artifacts if visible color shading is created on the canvas by the environment illumination. Without a modification to either the mathematical model or the rendering process the environment light's color can be compensated by determining it with aid of *colorimeters*, encoding this information in *E*, and passing it to the pixel shader.

The presented concept and techniques are applicable in combination with diffuse pictorial artwork, such as watercolor paintings, pen or ink drawings, sanguine sketches, or matt oil paintings. Extreme light and view-dependent effects, such as *non-Lambertian specular reflections*, *self-shadows*, *sub-surface scattering* and *interreflections* that are created by brush strokes, paint material or canvas textures cannot be handled with this method.

References

[Ban01] D Bandyopadhyay, R Raskar, A State, H Fuchs, Dynamic Spatially Augmented 3D Painting. UNC Chapel Hill Tech Report TR01-006, 2001.

[Bim01] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55, 2001.

[Bim02b] Bimber, O., Gatesy, S.M., Witmer, L.M., Raskar, R. and Encarnação, L.M. Merging Fossil Specimens with Computer-Generated Information. IEEE Computer, September, pp. 45-50, 2002.

[Bim02a] Bimber, O. and Fröhlich, B. Occlusion Shadows: Using Projected Light to Generate Realistic Occlusion Effects for View-Dependent Optical See-Through Displays. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 186-195, 2002.

[Bim03] Bimber, O. Grundhöfer, A., Wetzstein, G., and Knödel, S. Consistent Illumination within Optical See-Through Augmented Environments, In proceedings of IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'03), , pp. 198-207, 2003.

[Bim04a] Bimber, O. Combining Optical Holograms with Interactive Computer Graphics. In IEEE Computer, January issue, pp. 85-91, 2004.

[Bim04b] Bimber, O., Coriand, F., Kleppe, A., Bruns, E., Zollmann, S., and Langlotz, T. Superimposing Pictorial Artwork with Projected Imagery. Submitted to IEEE MultiMedia, 2004.

[Boi01] Boivin, S. and Gagalowicz, A. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In proceedings of ACM Siggraph, pp. 107-116, 2001.

[Bre73] Brent, R.P. Algorithms for minimization without derivatives. Prentice-Hall, Engelwood Cliffs, NJ, 1973.

[Bre96] Breen, D.E., Whitaker, R. T., Rose, E. and Tuceryan, M. Interactive Occlusion and Automatic Object Placement for Augmented Reality. Computer and Graphics Forum (proceedings of EUROGRAPHICS'96), vol. 15, no. 3, pp. C11-C22, 1996.

[Che00] Chen, Y., D. Clark, Finkelstein, A., Housel, T. and Li., K., Automatic Alignment of high resolution Multi-Projector Displays using an un-calibrated Camera, In proceedings of IEEE Visualization, pp. 125-130, 2000.

[Che93] S. E. Chen, and L. Williams. View Interpolation from Image Synthesis. SIGGRAPH '93, pp. 279-288, July 1993.

[Deb96] P. Debevec, C. J. Taylor, and J. Malik. Modeling and Rendering Architecture from Photographs. SIGGRAPH '96, August 1996.

[Deb98] P. Debevec, Y. Yu, and G. Borshukov. Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping. Proc. of 9th Eurographics Workshop on Rendering, June 1998.

[Dre97] Dreesen, F., von Bally, G. Color holography in a single layer for documentation and analysis of cultural heritage. In Optics within Life Sciences (OWLS IV), Springer Verlag Berlin, Heidelberg, New York, pp. 79-82, 1997.

[Dre00] Dreesen, F., Deleré, H., von Bally, G. High Resolution Color-Holography for Archeological and Medical Applications. In Optics within Life Sciences (OWLS V), Springer Verlag Berlin, Heidelberg, New York, pp. 349-352, 2000.

[Dre97] Drettakis, G. and Sillion, F. Interactive update of global illumination using a line-space hierarchy. In proceedings of ACM Siggraph'97, pp- 57-64, 1997.

[Fau93] O. Faugeras. Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, Cambridge, Massachusetts, 1993.

[For93] Fournier, A. Gunawan, A.S., and Romanzin, C. Common illumination between real and computer generated scenes. In proceedings of Graphics Interface'93, pp. 254-262, 1993.

[Gib00] Gibson, S. and Murta, A. Interactive rendering with real-world illumination. In proceedings of 11th Eurographics Workshop on Rendering, pp. 365-376, 2000.

[Gor84] Goral, C.M., Torrance, K.E., Greenberg, D.P., and Battaile, B. Modeling the interaction of light between diffuse surfaces. Computer In proceedings of ACM Siggraph'84, vol. 18, no. 3, pp. 212-222, 1984.

[Gor96] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. SIGGRAPH '96, August 1996.

[Hua01] Hua, H., Gao, C., Brown, L., Ahuja, N., and Rolland, J.P. Using a head-mounted projective display in interactive augmented environments. In proceedings of IEEE and ACM International Symposium on Augmented Reality (ISAR'01), pp. 217-223, 2001.

[Kaj86] J. T. Kajiya. The Rendering Equation. Computer Graphics 20(4) (1986), pp. 143-151.

[Kiy00] Kiyokawa, K., Kurata, Y. and Ohno, H. An Optical See-through Display for Mutual Occlusion of Real and Virtual Environments. In proceedings of IEEE & ACM ISAR 2000, pp. 60-67, 2000.

[Klu02] Klug, M.A., Scalable digital holographic displays, IS&T PICS 2001: Image Processing, Image Quality, Image Capture Systems Conference, Montreal, Quebec, Canada, pp. 26-32, 2001.

[Kol89] Kollin, J.S., Benton, S.A., and Jepsen, M.L. Real-Time Dispaly of 3-D Computer Holograms by Scanning the Image of an Acousto-Optic Mudulator. In Proceedings of SPIE, vol. 1136, Holographic Optics II: Principle and Applications, 1989.

[Lev96] M. Levoy, and P. Hanrahan. Light Field Rendering. SIGGRAPH ’96, August 1996.

[Los00] Loscos, C., Drettakis, G., and Robert, L. Interactive virtual relighting of real scenes. IEEE Transactions on Visualization and Computer Graphics, vol, 6, no. 3, pp. 289-305, 2000.

[Low01] Low, K., Welch, G., Lastra, A. and Fuchs, H. Life-Sized Projector-Based Dioramas, In proceedings of Symposium on Virtual Reality Software and Technology, pp-93-101, 2001.

[Luc93] Lucente, M. Interactive computation of holograms using a look-up table. Journal of Electronic Imaging, vol. 2, no. 1, pp. 28-34, 1993.

[Luc95] Lucente, M. and Tinsley, A. Rendering interactive images. Computer Graphics (proceedings of SIGGRAPH'95), pp. 387-394, 1995.

[Luc97] Lucente, M. Interactive three-dimensional holographic displays: seeing the future in depth. SIGGRAPH Computer Graphics, special issue on Current, New, and Emerging Display Systems, 1997.

[Mat98] Mather G, Verstraten F, and Anstis S The Motion Aftereffect: a Modern Perspective. Cambridge, Mass: MIT Press, 1998. (http://www.biols.susx.ac.uk/home/George_Mather/Motion/)

[Maj00] Majumder, A., He, Z., Towles, H., and Welch, G., Achieving Color Uniformity Across Multi-Projector Displays, In proceedings of IEEE Visualization, pp. 117–124, 2000.

[McM96] L. McMillan, and G. Bishop. Plenoptic Modeling. SIGGRAPH ’95, August 1995.

[Nae02] Naemura, T., Nitta, T., Mimura, A., Harashima, H. Virtual Shadows – Enhanced Interaction in Mixed Reality Environments. In proceedings of IEEE Virtual Reality (IEEE VR’02), pp. 293-294, 2002.

[Nak01] Nakamae, E., Harada, K., Ishizaki, T., and Nishita, T.A montage method: The overlaying of computer generated images onto background photographs. In proceedings of ACM Siggraph’86, pp. 207-214, 1986.

[Nei96] Neider, J., Davis, T., and Woo, M. OpenGL Programming Guide. Release 1, Addison Wesley, ISBN 0-201-63274-8, pp.157-194, 1996.

[Nod99] Noda, S., Ban, Y., Sato, K., and Chihara, K. An Optical See-Through Mixed Reality Display with Realtime Rangefinder and an Active Pattern Light Source. Transactions of the Virtual Reality Society of Japan, vol. 4, no. 4, pp. 665-670, 1999.

[Pan] Panoram Technology. http://www.panoramtech.com

[Pet03] Petz, C. and Magnor, M. Fast Hologram Synthesis for 3D Geometry Models using Graphics Hardware. In proceedings of SPIE’03, Practical Holography XVII and Holographic Materials IX, vol. 5005, pp 266-275, 2003.

[Pin01] Pinhanez, C. The everywhere displays projector: A device to create ubiquitous graphical interfaces, In proceedings of Ubiquitous Computing, pp. 315-331, 2001.

[Pre92] Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P. Numerical Recipes in C - The Art of Scientific Computing (2nd edition), Cambridge University Press, ISBN 0-521-43108-5, pp. 412-420,1992.

[Ras98] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays. SIGGRAPH ’98, July 1998

[Ras99] R. Raskar, M. Brown, R. Yang, W. Chen, G. Welch, H. Towles, B. Seales, H. Fuchs. Multi-Projector Displays Using Camera-Based Registration. IEEE Visualization 99, October 1999.

[Ras99b] R. Raskar, G. Welch, W. Chen. Tabletop Spatially Augmented Reality: Bringing Physical Models to Life using Projected Imagery. Second Int Workshop on Augmented Reality (IWAR'99), October 1999, San Francisco, CA

[Ras01] Raskar, R., Welch, G., Low, K.L. and Bandyopadhyay, D. Shader Lamps: Animating real objects with image-based illumination, In proceedings of Eurographics Rendering Workshop, pp. 89-102, 2001.

[Ras02] Raskar, R., Ziegler, R. and Willwacher, T. Cartoon Dioramas in Motion, In proceedings of Int. Symp on Non-photorealistic Animation and Rendering, pp. 7-ff, 2002.

[Ras03] Raskar, R., van Baar, J., Beardsly, P., Willwacher, T., Rao, S. and Forlines, C. iLamps: Geometrically Aware and Self-Configuring Projectors, In proceedings of ACM Siggraph, pp. 809-818, 2003.

[Ras04] R. Raskar, P Bearsdley, J VanBaar, Y Wang, P Dietz, J Lee, D Leigh, T Willwacher. RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors. SIGGRAPH '04, 2004.

[Sze97] R. Szeliski and H. Shum. Creating Full View Panoramic Mosaics and Environment Maps. SIGGRAPH '97, August 1997.

[Und99] Underkoffler, J., Ullmer, B. and Ishii, H. Emancipated pixels: real-world graphics in the luminous room, In proceedings of ACM Siggraph, pp. 385-392, 1999.

[Yan01] Yang, R., Gotz, D., Hensley, J., Towels., H., and Borwn, M., PixelFlex: A Reconfigurable Multi-Projector Display System, In proceedings of IEEE Visualization, 2001.

[Yam90] Yamaguchi, M, Ohyama, N., and Honda, T. Holographic 3-D Printer. In Proceedings of SPIE, V. 1212, p. 84, 1990.

[Yos03] Yoshida, T., Horii, C. and Sato, K. A Virtual Color reconstruction System for Real Heritage with Light Projection, In proceedings of Virtual Systems and Multimedia, pp. 158-164, 2003.

[Yu99] Yu, Y., Debevec, P., Malik, J., and Hawkins, T. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In proceedings of ACM Siggraph' 99, pp. 215-224, 1999.

5. Summary and enabling technologies

In this tutorial we discussed the Spatial Augmented Reality (SAR) concept. For specific applications, SAR allows to overcome some of the limitations linked to conventional AR displays. In addition it holds the potential of opening doors to new application domains.

First, we gave an overview over different AR display techniques that may enable readers to identify parallels between virtual reality and augmented reality, and stimulate them to think about alternative display approaches for AR.

One of our focuses was on spatial optical see-through technology: We described different display configurations using transparent screens and mirror beam-splitters, as well as interactive rendering techniques that support such configurations and neutralize optical distortion.

The other focus was on projector-based augmentation and illumination. Rendering techniques for non-trivial (geometry and texture) projection screens were described. Finally, the projector-based illumination concept was explained, and examples were outlined how it can be used to create consistent illumination and occlusion effects.

We want to annotate that upcoming and new technology will not only open new possibilities for SAR, but also for other display concepts, such as hand-held and head-attached displays.

Projectors of the near future, for instance, will be compact, portable, and with the built-in awareness which will enable them to automatically create satisfactory displays on many of the surfaces in the everyday environment. Alongside the advantages, there are limitations, but we anticipate projectors being complementary to other modes of display for everyday personal use in the future, and to have new application areas for which they are especially suited. LEDs are replacing lamps and reflective instead of transmissive displays (DLPs, LCOS) are becoming popular. Both lead to improved efficiency requiring less power and less cooling. DLP and LCOS projectors can display images at extremely high frame rates, currently 180Hz and 540 Hz respectively, but lack video bandwidth. Several efforts are already in the making and are very promising. For example Symbol Technologies [Sym02] has demonstrated a small laser projector (two tiny steering mirrors for vertical and horizontal deflection) and has even built a handheld 3D scanner based on such a projector. Siemens has built a 'mini-beamer' attachment for mobile-phones [Sie02]. Cam3D has built a 'Wedge' display where a projector can be converted into a 'flat panel' display by projecting images at the bottom of a wedge shaped glass [Tra02]. A

future mobile projector may double up as 'flat panel' when there is no appropriate surface to illuminate, or ambient light is problematic. Super bright, sharp infinite focus laser projectors are also becoming widespread [Jen02] which may allow shape-adaptive projection without focus and ambient lighting problems. In addition suitable input devices are also appearing; e.g., Canesta [Cas02] has built a projected laser pattern on which one can type. The finger movement is detected by IR sensing. Finally novel lamp designs, especially those based on LEDs or lasers are creating smaller, lighter, efficient and long-life solutions. Two important problems in Augmented Reality, object identification and determining the pose of the displayed image with respect to the physical object, can be solved by using photosensing RFID tags. This is being explored as a Radio Frequency Identification and Geometry (RFIG) discovery method [Ras04]. Figure 5.2 shows a current application in warehouse scenario where two employees can locate products that are about to expire using RF channel to send the query but optical channel to locate the corresponding wireless tags. The handheld projector finally displayed the appropriate information overlaid on those objects.

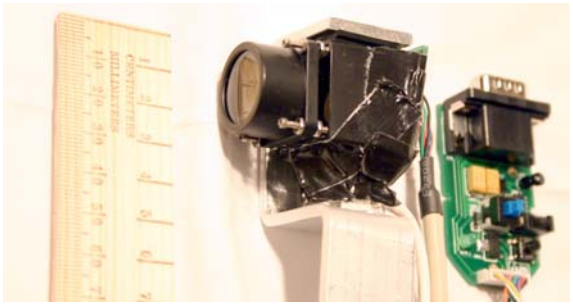


Figure 5.1: *A micro-projector prototype we have built -- to indicate the viability of designing a projector for truly handheld use. It is a 1-inch cube using an LED and LCOS imager to create a color animated projection at a distance of about 12-inches.*

Figure 5.2: *Warehouse scenario using RFIG (Ras04). A user directs a handheld projector at tagged inventory, with communication mediated by two channels -- RF and photo-sensing on the tags. The user sees a projection of the retrieved tag information collocated with the physical objects, and performs a desktop-like interaction with the projection. A second user performs similar operations, without conflict in the interaction because the projector beams do not overlap.*

Organic Light Emitting Diodes (OLEDs) [How01], for instance, may replace the crystalline LEDs that are currently being used to build the miniature displays for HMDs. OLEDs promise to produce cheap and very high-resolution full-color matrix displays that can give head-attached displays a technological push. In contrast to normal LEDs, OLEDs are made from plastic compounds instead from semi-conducting elements, such as silicon or gallium, etc. Like LEDs, OLEDs glow when voltage is applied. Two main classes exist today: small molecule OLEDs and polymer OLEDs. While small molecule OLEDs are built up by depositing molecules of the compound onto the display itself under very low pressure, polymer OLEDs have the active molecules suspended in a liquid-like pigment in paint. It can be printed onto displays using ink jets, screen printing or any of the various contact techniques used for ordinary inks. Small molecule OLED displays are limited in size, but they may be suitable for head-mounted displays. Polymer OLEDs can be used to build large scale displays – such as 500 inch displays or larger. Resolution approaching 300 dpi is also possible, approaching the quality of ink on paper. They may become more interesting for spatial AR approaches.

The general advantages of OLEDs are:

- Because OLED displays are fluorescent and don't require backlights they will need far less power than LCD screens (currently LCDs require three times as much power than OLEDs);
- OLED layers can be made much thinner than LCD layers (about a thousand times thinner than a human hair);
- In contrast to LCDs, OLEDs don't use polarized light filters. The displayed images on OLEDs can be viewed from a much wider angle;
- OLEDs have a much wider working temperature range than LCDs;
- Can be "printed" onto large-scale and flexible display surfaces of any shape and (almost) any material.

Today's reality, however, is that the OLED compounds degrade over time (especially when they get in contact with oxygen or water) – limiting the maximum lifetime of a display. Different colors degrade at different rates – making the color balance change. These problems may be solved in future.

A variation of OLEDs are *Light Emitting Polymers* (LEPs) [Bur90] that provide the opportunity for the fabrication of large, flexible, full-color, fast emissive displays with a high resolution, a wide viewing angle and a high durability. In LEP technology, a thin film of light-emitting polymer is applied onto a glass or plastic substrate coated with a transparent electrode. A metal electrode is evaporated on top of the polymer. The polymer emits light when the electric field between the two electrodes is activated. The response time of LEPs is ultra-fast (sub-microsecond) and is unaffected by temperature. Consequently they may support high enough frame rates for active stereoscopic rendering. Light emission occurs at low voltage, and (unlike LCD or plasma displays) it can be fabricated on a single sheet of glass. Also, because it can be made of flexible plastic substrates, it is not only extremely difficult to break, but can also be molded into different shapes and contours.

The advantages of LEPs can be summarized as follows:

- Since a low voltage is required LEPs need little power;
- The response time of LEPs is very high – potentially allowing active stereoscopic rendering;
- Can be fabricated on transparent glass or plastic surfaces of any shape;
- LEPs provide a high contrast (currently between 3-5 times higher than LCDs).

Transparent LEPs may present other future alternatives for spatial AR configurations.

Electronic paper (or E-Ink) [Xer03] also has potential as an AR display technology of the future. Here an electric charge moves magnetic colored capsules within the "paper" either toward or away from the surface in order to form an image. The capsules retain their positions until another charge is applied. The ink simply resides on the display while an image is not changing; therefore, it consumes no power. Philips and other companies in the field are working on color as well as bendable applications. The main advantage of electronic paper is that it does not require power at all, as long as the displayed image does not change. The current generation of electronic paper, however, is a black-on-white technology where the ink is pumped onto a white background, giving maximum readability.

Several approaches exist that allow a projection of 2D images onto a screen composed of fog [Fog04] or condensed air [IO204]. Such *fog/air displays* [Pau04] suffer from distortion caused by turbulences but allow through-the-screen interactions since they lack a physical screen boundary.

Solid-state displays [Dow96] generate visible photons (i.e. light) within a transparent host material by exciting optically active ions with special energy sources. Ions at a known three-dimensional position within the host materials can be excited by crossing energy beams, such as infrared lasers, ultraviolet sources of radiation, or electron beams.

Examples of suitable host materials are various gases, crystals and electro-active polymers. The host material must provide several properties:

- In its initial state it must be transparent;
- It must emit visible light in its excited state;
- Its inner structure must be homogenous;
- It must have a refraction index similar to air to avoid distortion.

If it was possible to use air as the host material, then this approach would represent the Holy Grail – not only for spatial augmented reality displays, but for 3D display technology in general. Unfortunately this is not yet feasible. In addition, conceptual problems, such as the "ghost voxel problem" (when energy beams that are used to create voxels intersect with other beams and create unwanted voxels) have to be solved.

Parallax displays are display screens (e.g., CRT or LCD displays) that are overlaid with an array of light-directing elements. Depending on the observer's location, the emitted light that is presented by the display is directed in a way that it appears to originate from different parts of the display while changing the viewpoint. If the light is directed to both eyes individually, the observer's visual system interprets the different light information to be emitted by the same spatial point. Examples of parallax displays are parallax barrier displays (cf. figure 5.3-top) that apply a controllable array of light-blocking elements (e.g., a light blocking film or liquid crystal barriers [Per00]) in front of a CRT screen. Depending on the observer's viewpoint, these light-blocking elements are used to direct the displayed stereo-images to the corresponding eyes. Other examples are lenticular sheet displays (cf. figure 5.3-bottom) that apply an array of optical elements (e.g., small cylindrical or spherical lenses) to direct the light for a limited number of defined viewing-zones.

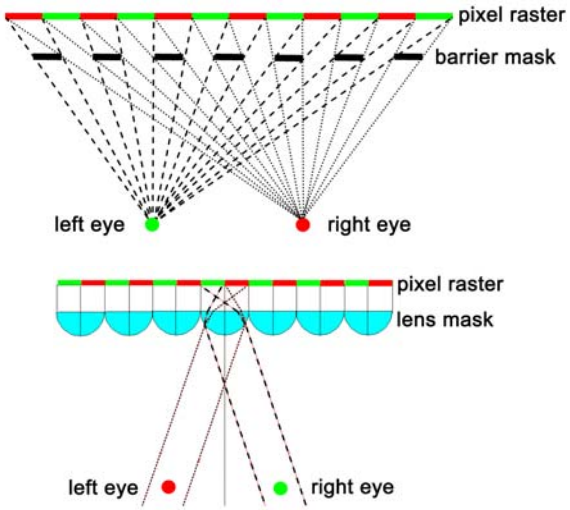


Figure 5.3: *Basic parallax display concepts: (top) parallax barrier display and (bottom) lenticular sheet display.*

Parallax displays can be published and mass-produced in a wide range of sizes, and can be used to display photo-realistic images. Many different types are commercially available and used from desktop screens to cell-phone displays. We believe that the application of parallax displays will be the next logical and feasible step for SAR solutions towards autostereoscopy.

Figure 5.4: *A first autostereoscopic, multi-user capable SAR display, (information available through www.uni-weimar.de/medien/AR).*

Figure 5.4 shows a first running autostereoscopic SAR display. It uses a special barrier mask to generate multiple viewing zones simultaneously and is able to support many surrounding observes at the same time. This allows detaching the display technology from the users completely.

In the future, a stronger combination of computer graphics and holography (as explained in chapter 4) can also be expected. Any three-dimensionally programmed computer image can, in principle, be transformed into a hologram basically by way of Fourier transformations. These, however, presuppose large computational, network and storage capacities. For instance, a pre-rendered or recorded movie-length holographic video could require a petabyte (1 million gigabytes) of storage. Holographic storage itself, which uses lasers to record data in the three dimensions of a clear crystalline medium may be a future solution to the problem that non-static holograms require a massive amount of data. While computer generated static images can already be transformed into large digital holograms using holographic printers [Klu01], interactive or real-time electronic holography [Luc97] with an acceptable quality (size, resolution and colors) would still require the invention of more advanced light modulators, faster computers with a higher bandwidth, and better compression techniques. However, the combination of high quality optical holograms and interactive computer graphics is already feasible with off-the-shelf hardware today [Bim04].

In the short run, especially high-resolution bright and flexible projection devices, high-performance and cost-efficient rendering hardware, reliable, precise and wireless tracking technology, and advanced interaction techniques and devices will pave the way for forthcoming alternative AR configurations. In the long run, new display concepts, such as autostereoscopy and holography will replace goggle-bound stereoscopic displays (at least for non-

mobile applications). However, the underlying technology must be robust, flexible and the technology that directly interfaces to users should adapt to humans, rather than forcing users to adapt to the technology. Therefore, human-centered and seamless technologies, devices and techniques will play a major role for augmented reality of the future. That this is feasible can be demonstrated based on the example of one early SAR display – the Virtual Showcase [Bim01]. Since its invention in 2000, several prototypes have been exhibited to an approximated total number of more than 25,000 – 30,000 visitors/users (user-study and picture gallery are available through www.uni-weimar.de/AR). Some events were technically supervised short-term exhibitions (1-7 days) at conferences and trade shows, others have been non-supervised long-term exhibitions (3-4 month) in museums. During these years, we have made the experience that SAR technology can be robust, attractive and efficient enough to be applied successfully outside research laboratories.

References

[Bim01] Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55, 2001.

[Bim04] Bimber, O. Combining Optical Holograms with Interactive Computer Graphics. In IEEE Computer, January issue, pp. 85-91, 2004.

[Bur90] Burroughes, J.H. Light Emitting Polymers. Nature, pp. 347, 1990.

[Cas02] Canesta. Miniature Laser Projector Keyboard. Retrieved from the World Wide Web: http://www.canesta.com. 2002.

[Dow96] Downing, E.A., Hesselink, L., Ralston, J. and Macfarlane, R. A three-color, solid-state three-dimensional display. Science, vol. 273, pp. 1185-1189, 1996.

[Fog04] Fogscreen Inc. Projecting Images in thin Air. Retrieved from the World Wide Web: http://www.fogscreen.com/ , 2004.

[How01] Howard, W. E. Organic Light Emitting Diodes (OLED). OLED Technology Primer, Retrieved from the World Wide Web: http://www.emagin.com/oledpri.htm, 2001.

[IO204] IO2 Technology, The HelioDisplay. Retrieved from the World Wide Web: http://www.io2technology.com/ , 2004.

[Jen02] Jenoptik. Laser Projector. Retrieved from the World Wide Web: http://www.jenoptik.com, 2002.

[Klu01] Klug, M.A., Scalable digital holographic displays, IS&T PICS 2001: Image Processing, Image Quality, Image Capture Systems Conference, Montreal, Quebec, Canada, pp. 26-32, 2001.

[Luc97] Lucente, M. Interactive three-dimensional holographic displays: seeing the future in depth. SIGGRAPH Computer Graphics, special issue on "Current, New, and Emerging Display Systems", 1997.

[Pau04] Paulson, L.D. Displaying Data in Thin Air. IEEE Computer, March issue, p. 19, 2004.

[Per00] Perlin, K., Paxia, S., and Kollin, J.S. An autostereoscopic display. Computer Graphics (proceedings of SIGGRAPH'00), pp. 319-326, 2000.

[Ras04] R. Raskar, P Bearsdley, J VanBaar, Y Wang, P Dietz, J Lee, D Leigh, T Willwacher. RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors. SIGGRAPH '04, 2004.

[Sie02] Siemens. Siemens Mini Beamer. Retrieved from the World Wide Web: http://w4.siemens.de/en2/html/press//newsdesk archive/2002/-foe02121b.html, Cited December 2002.

[Sym02] Symbol. Laser Projection Display. Retrieved from the World Wide Web: http://www.symbol.com/products/oem/lpd.html, December, 2002.

[Tra02] Travis, A., Payne, F., Zhong, J., and More, J. Flat panel display using projection within a wedge-shaped waveguide. Retrieved from the World Wide Web: http://ds.dial.pipex.com/cam3d/technology/technology01.html, 2002.

[Xer03] Xerox PARC, Electronic Reusable Paper. Retrieved from the World Wide Web: http://www2.parc.com/dhl/projects/gyricon/ , 2003.

Acknowledgements

We would like to acknowledge and thank all contributors of this tutorial: M. Agrawala, R. T. Azuma, J. van Baar, D. Bandyopadhyay, P. Beardsley, E. Foxlin, G. Goebbels, H. Hua, M. Inami, N. Kawakami, Y. Kitamura, K. Kiyokawa, K. Low, J. Newman, T. Ogi, C. Pinhanez, A. Preston, B. Schwald, G. Stetten, B. Kaelin, G. Wetzstein, T. Willwacher and R. Ziegler.

The Virtual Showcase project is supported by the European Union (IST-2001-28610).

The HoloGraphics project is supported by the Deutsche Forschungsgemeinschaft (DFG BI 835/1-1).

Spatial Augmented Reality

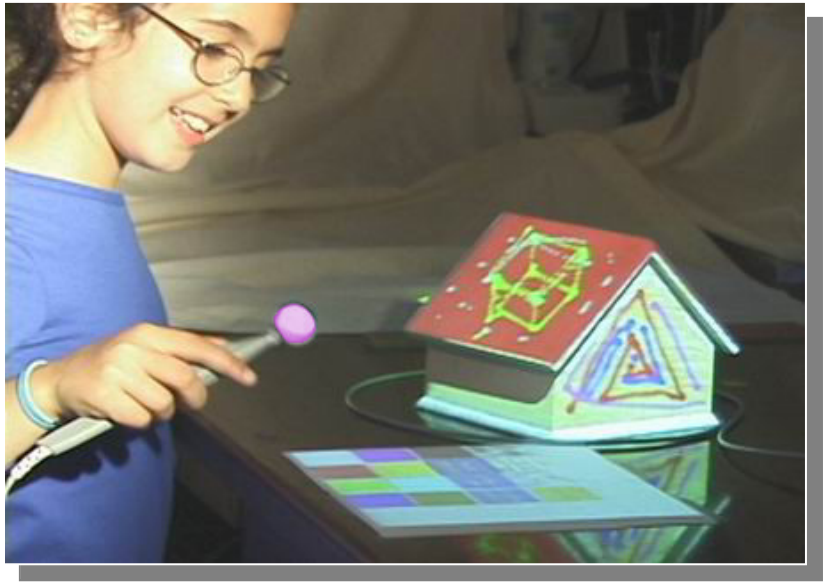
- Oliver Bimber
  - University of Weimar
- Ramesh Raskar
  - Mitsubishi Electric Research Labs

Welcome

- AR Display Approaches
  - Traditional, Goggle-bound, Alternative
- Spatial Augmentation
  - Projectors, Non-planar, Mobile objects, Change appearance, Interaction
- Exploiting Optical Elements
  - Mirrors, Beam splitters, Holograms
- Applications
  - Prototypes in Art, Research and Industry

A Non-traditional Approach

- Painting with Light
  - [Bandyopadhyay, Raskar, Fuchs 2001]



Goals

- Understand advantages of Spatial AR
- Discuss issues in traditional AR approaches
- Explore alternative AR methods
  - Graphics, Vision, Optics techniques
  - Learn math of rendering and calibration
- See new applications in art and industry

- What we will not cover
  - Fundamentals of AR and VR

Speaker

Oliver Bimber is currently a Junior Professor for Augmented Reality at the Bauhaus University Weimar, Germany. He serves on the editorial board of the IEEE Computer magazine (graphics and multimedia editor). Bimber invented several AR displays, such as the Virtual Showcase, Interactive Holograms and Augmented Paintings. He published in several journals, such as IEEE Computer, IEEE CG&A, Presence, Computers and Graphics, and at many graphics, AR and VR conferences. Bimber is member of IEEE, Eurographics, ACM and ACM Siggraph. He has exhibited in the Emerging Technology Laboratory of Siggraph'02 and was a speaker in the Siggraph'02 course "Building Interactive Workspaces". Together with Ramesh Raskar, Bimber organized this course at Eurographics 2003 and 2004

Speaker

Ramesh Raskar is a Senior Research Scientist at MERL-Cambridge Research. His research interests include projector-based graphics, projective geometry and non-photorealistic rendering. During his doctoral research at U. of North Carolina at Chapel Hill, he developed a framework for projector based 3D graphics, which can simplify the constraints on conventional immersive displays, and enable new projector-assisted applications. He has published several articles on immersive projector-based displays, spatially augmented reality and has introduced Shader Lamps, a new approach for projector-based augmentation. His technical papers have appeared in SIGGRAPH, EuroGraphics, IEEE VR, IEEE Visualization, CVPR and many other graphics and vision conferences. He was a course organizer and speaker for Siggraph 2002-2006. He is a member of the ACM and IEEE.

http://www.merl.com/people/raskar/raskar.html

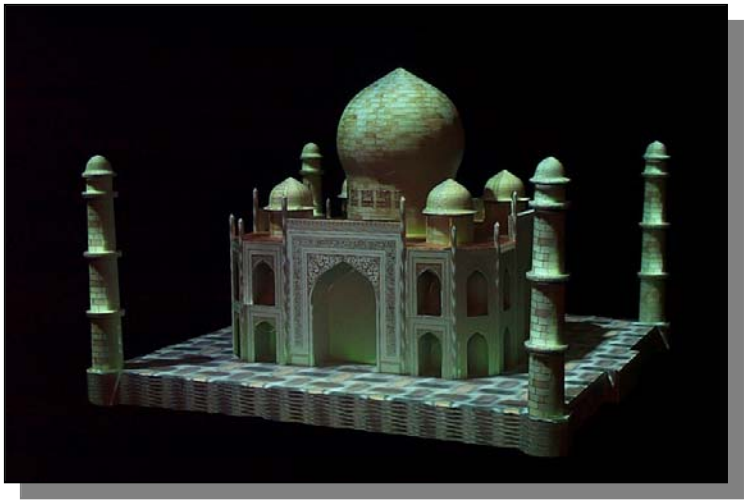
Opportunities

- Think beyond goggle-bound AR
- Learn techniques using projectors, flat displays and optics
- Explore more realistic augmented environments
- Learn how to build your own spatial AR displays (only covered in tutorial notes).
- Learn how to extend your own software framework to support spatial AR displays
- Get an impression on applicability

ShaderLamps

Virtual Reflectance

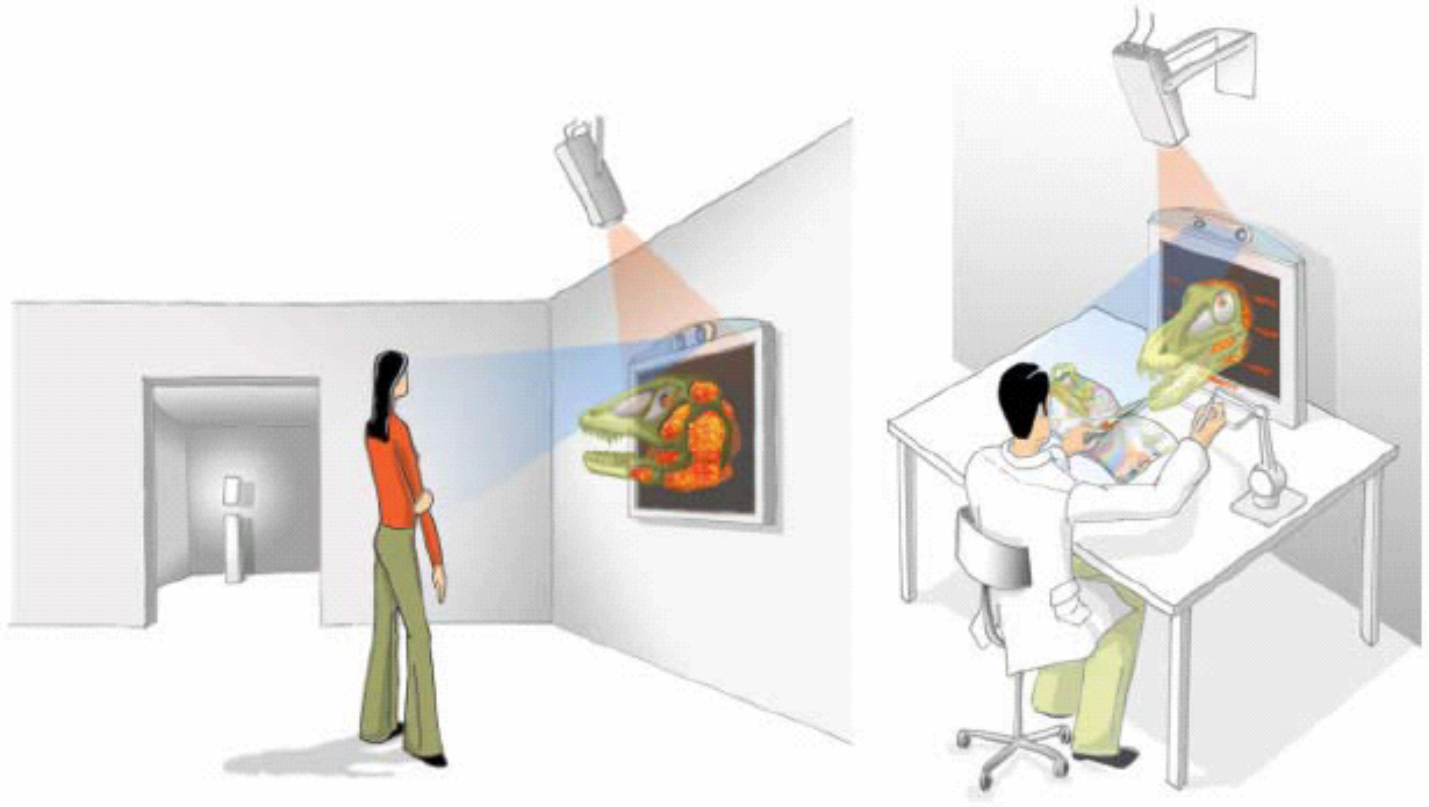
Virtual Illumination



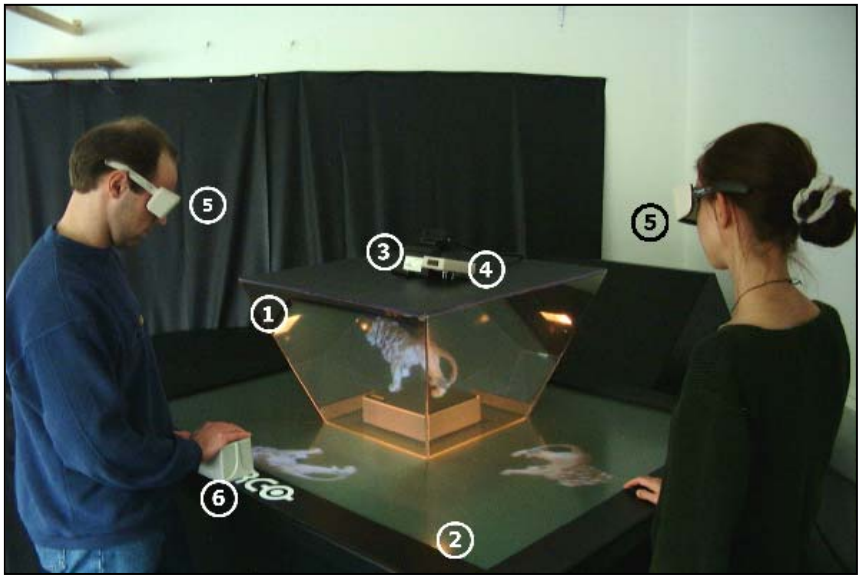
Virtual Motion

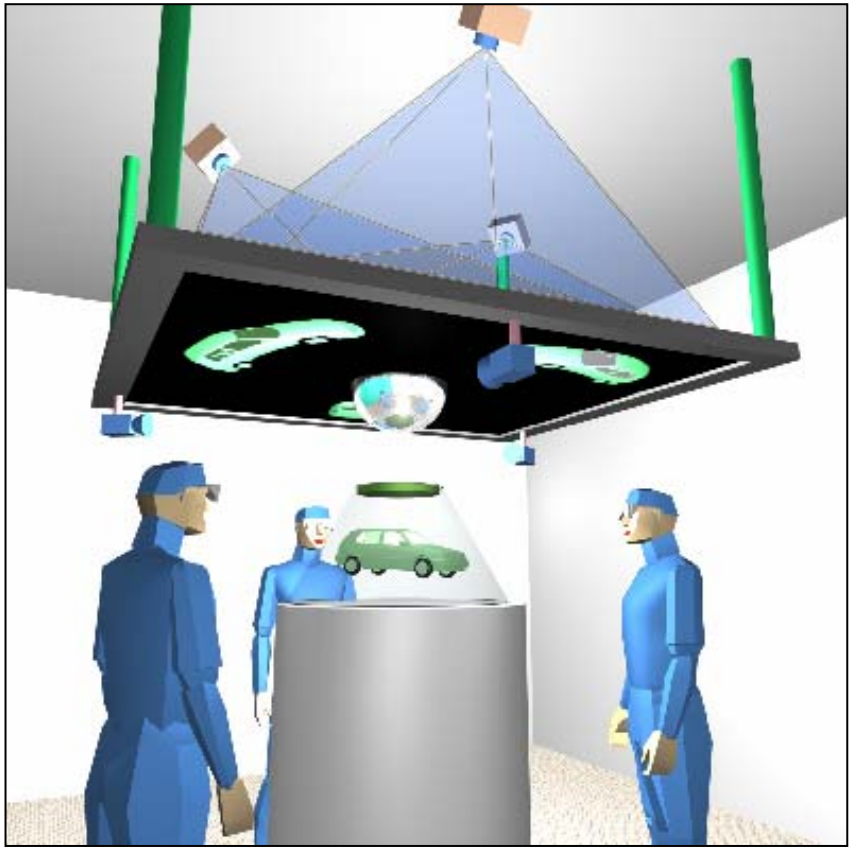
Interaction

HoloGraphics



Virtual Showcase



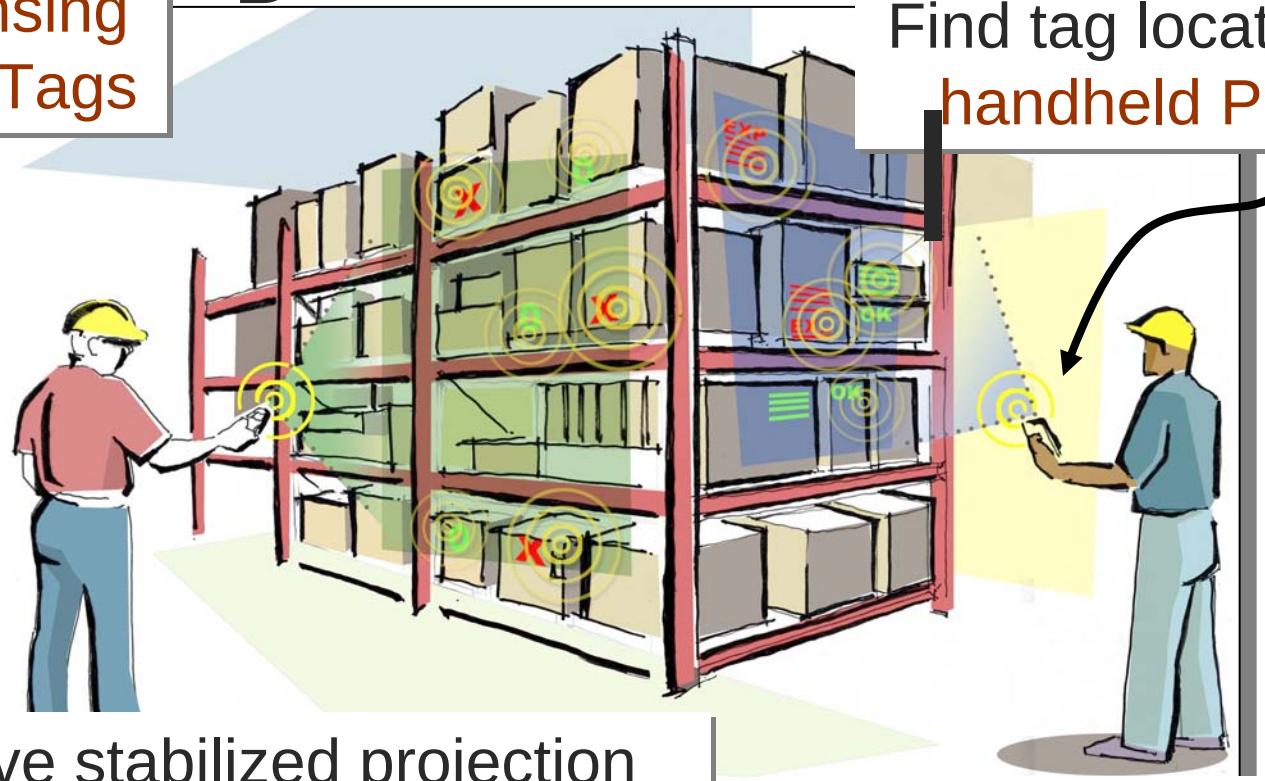


Handheld AR

F (Radio Frequency Id & Geometry) F G

Photosensing Wireless Tags

Find tag location using handheld Projector



Interactive stabilized projection

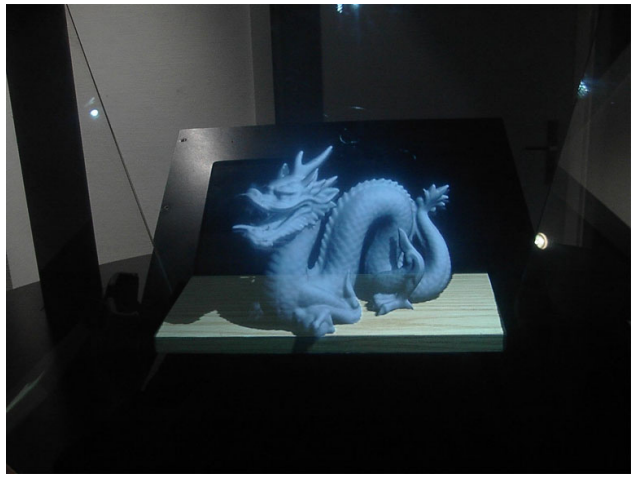
Many geometric ops

Survey

- How many of you are **users of VR system** ?
- How many of you are **users of AR system** ?
- How many of you are involved in AR **design/development** ?
- How many do **active programming** ?
- What **type of AR** do you use:
  HMD ? Handheld ? Projector ?
- **Field of work**:
  Academics? Industry ? Research ? Art ?

Augmented Reality

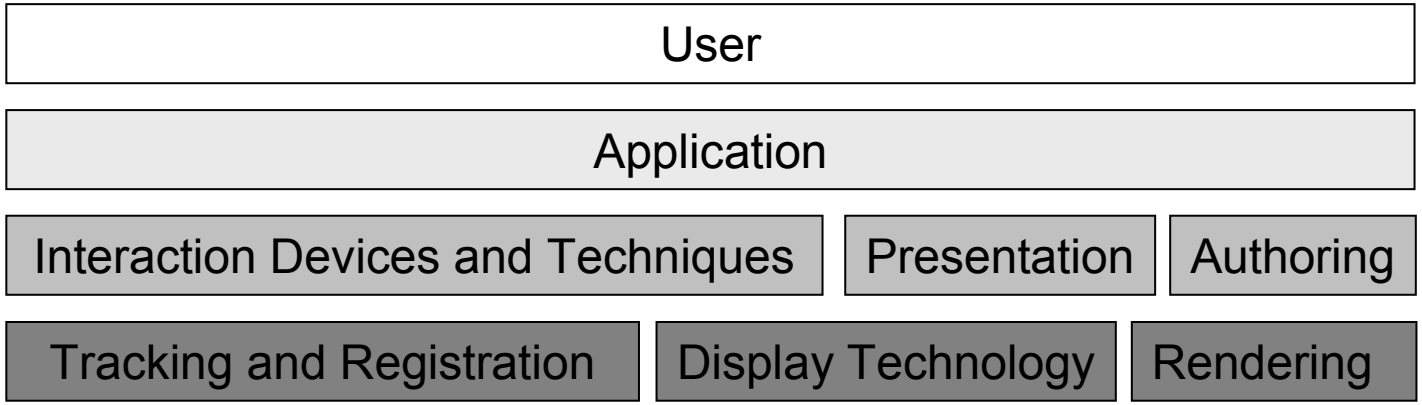
Bimber, et al. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55,2001.



optical see-through

video see-through

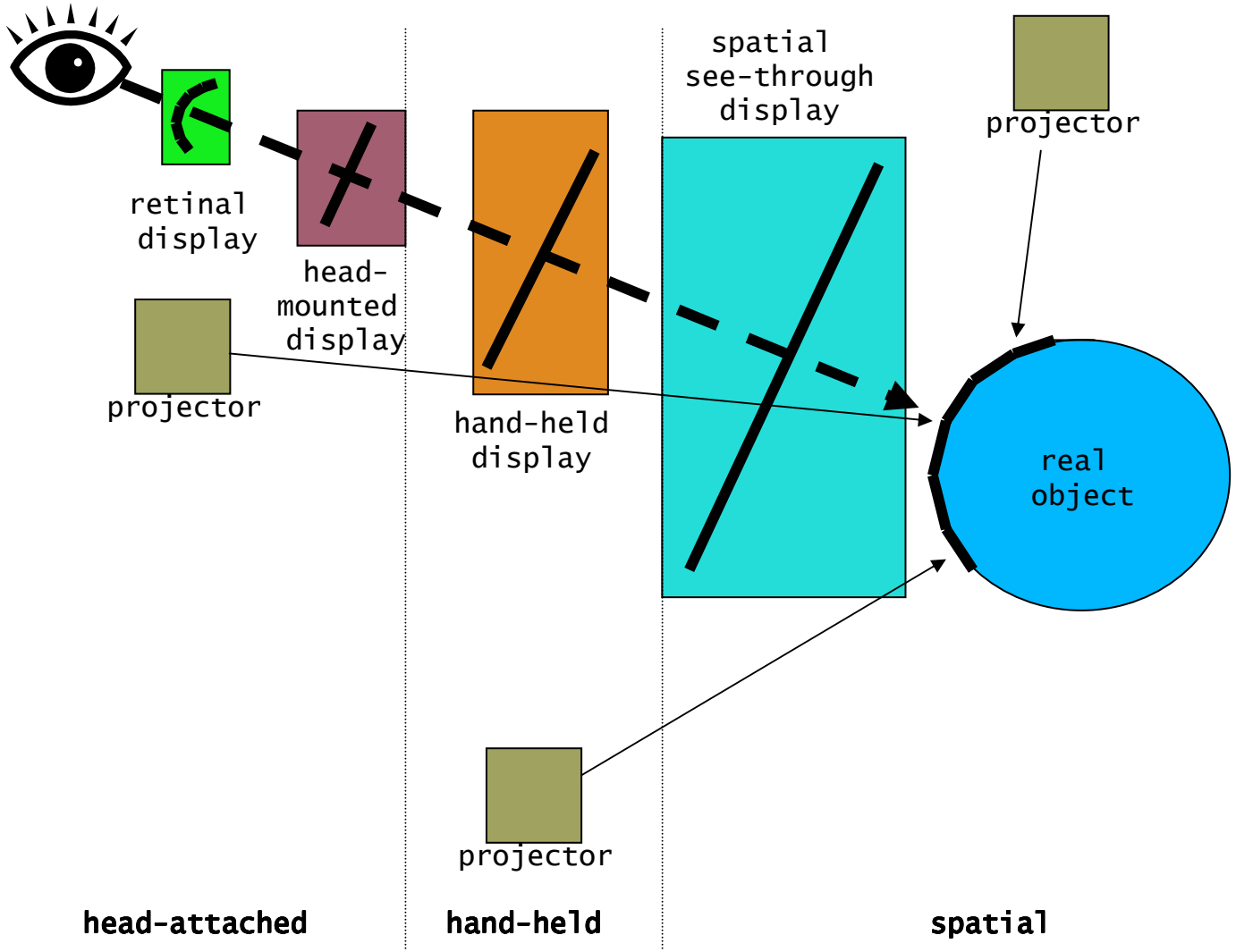
Gibson, et. al, Interactive Rendering with Real-World Illumination, In proceedings of 11th Eurographics Workshop on Rendering, Brno, Czech Republic, June 2000.



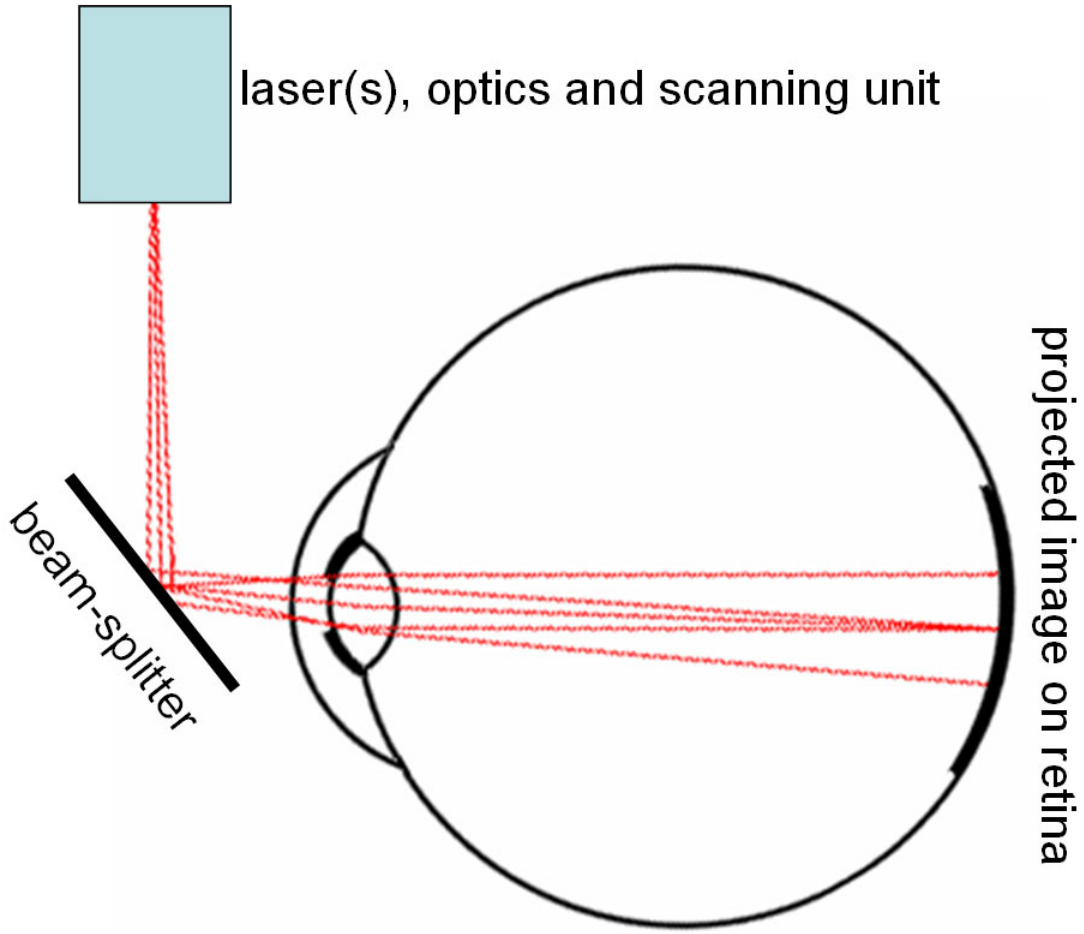
Ultimate AR Displays?

20th Century Fox / Lucasfilm Ltd.

Today's AR Displays


retinal display
head-mounted display
projector
hand-held display
spatial see-through display
projector
real object
projector
head-attached
hand-held
spatial

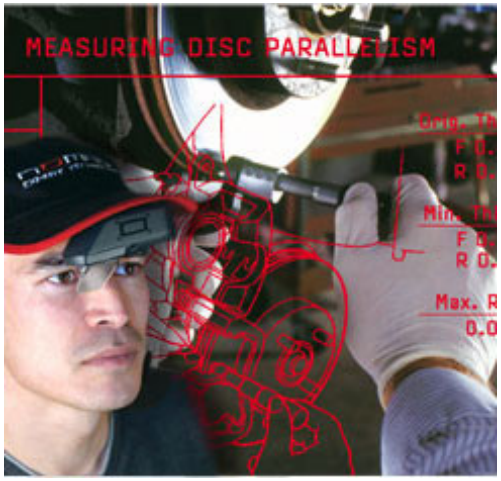
Retinal Displays



Retinal Displays

Pros:

- High brightness and contrast;
- Low power consumption (make them well suited for mobile outdoor applications);
- Support mobile applications;
- Support multi-user applications;
- Future generations also hold the potential to provide dynamic re-focus, full-color stereoscopic images, and an extremely high resolution and large field-of-view.

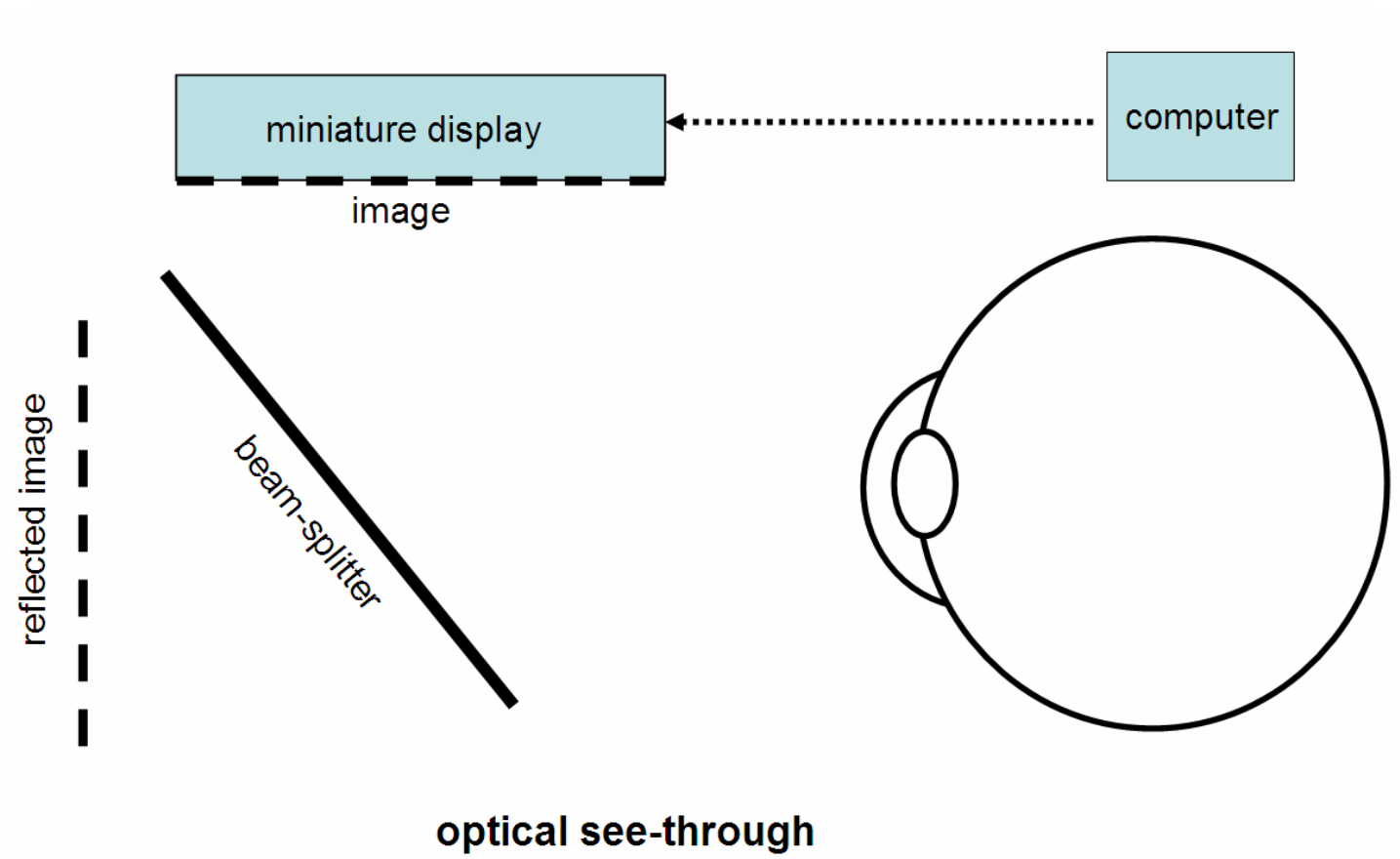


Microvision, Inc.

Cons:

- Only monochrome (red) images;
- The sense of ocular accommodation is not supported;
- Stereoscopic versions do not exist.

Head-Mounted Displays



Head-Mounted Displays

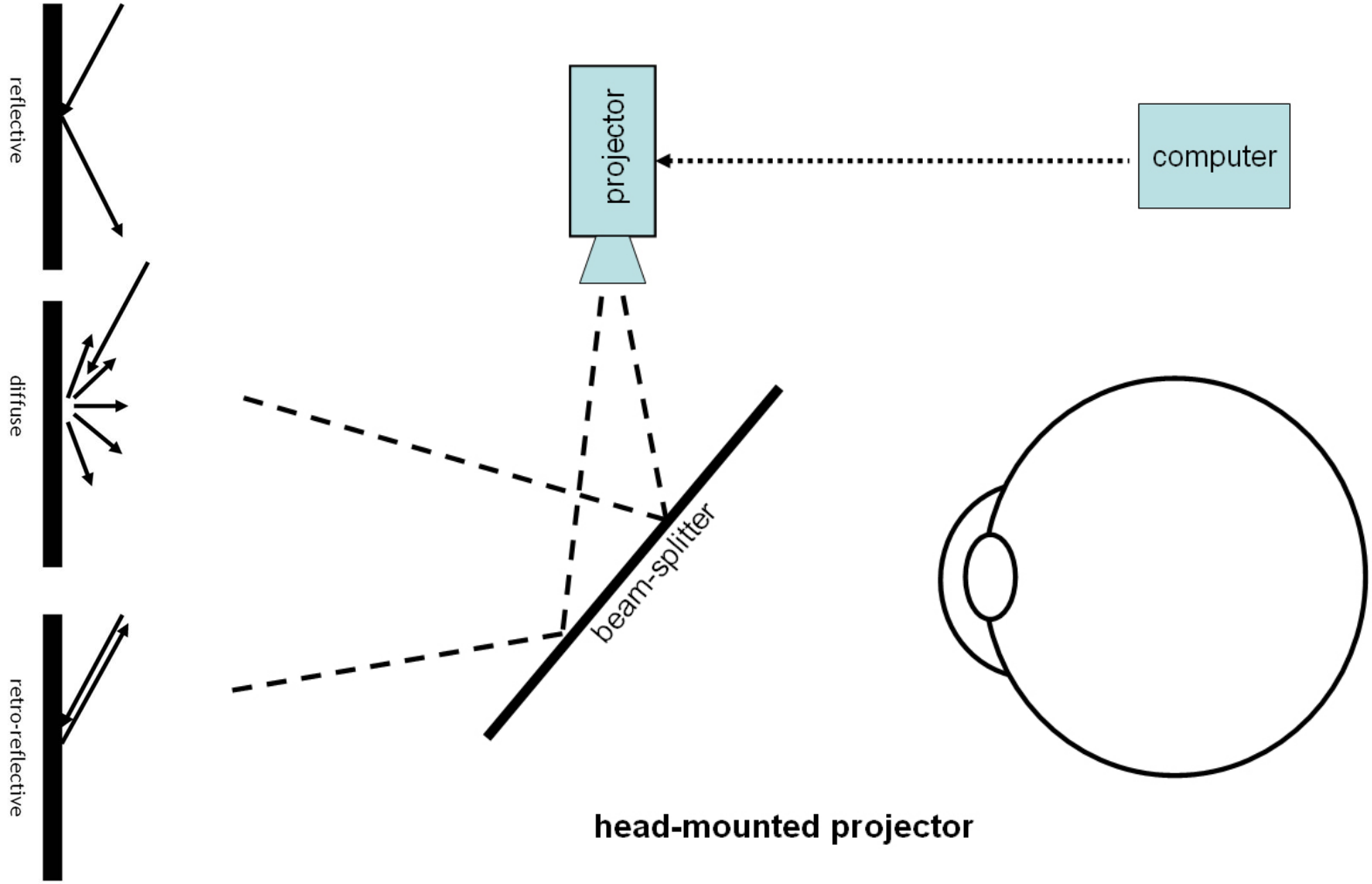
Pros:

- Currently the dominant display technology for AR;
- Support mobile applications;
- Support multi-user applications;
- Some specialized versions support mutual occlusion (like Kiyokawa's ELMO)

Cons:

- Lack in resolution;
- Limited field of view;
- Heavy optics and vs. ergonomic devices;
- Constant image depth;
- Optical see-through: difficult (user and session dependent) calibration and precise head-tracking;
- Video see-through: image processing for optical tracking increases the end-to-end system delay;
- Conventional optical see-through devices are incapable of providing consistent occlusion.

Head-Mounted Projectors

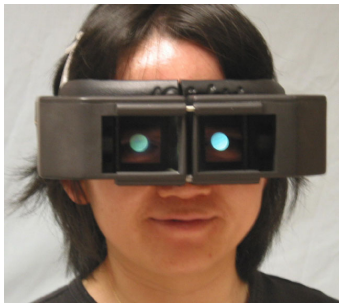


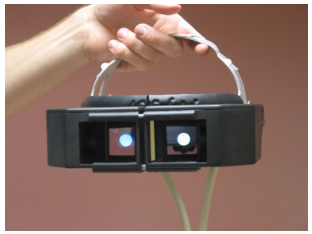
head-mounted projector

Head-Mounted Projectors

Pros:

- Support mobile applications;
- Support multi-user applications;
- Have the potential of combining the advantages of projection displays (i.e., resolution, FOV) with the advantages of traditional HMDs.





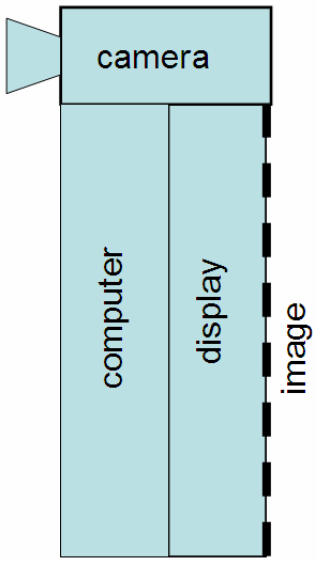
Hua, et al. Using a head-mounted projective display in interactive augmented environments. In Proceedings of IEEE and ACM International Symposium on Augmented Reality 2001, pp. 217-223, 2001.

Inami, et al. and Tachi, S., Visuo-Haptic Display Using Head-Mounted Projector, Proceedings of IEEE Virtual Reality 2000, pp.233-240, 2000.

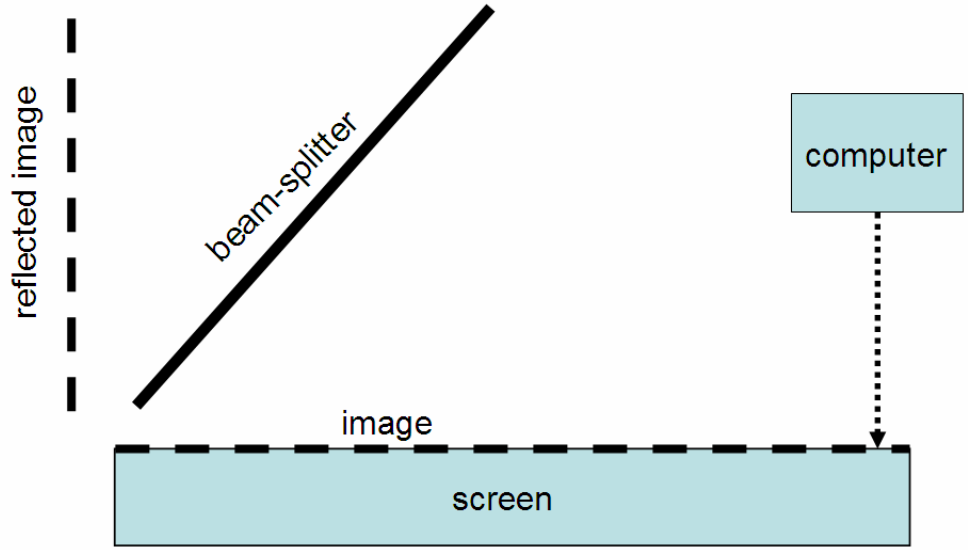
Cons:

- Limited resolution and brightness;
- Head-mounted projective displays might require special display surfaces (i.e., retro-reflective surfaces) to provide bright images;
- For projective head-mounted displays, the brightness of the images depends on the environmental light conditions;
- Projective head-mounted displays can only be used indoors, since they require the presence of a ceiling.

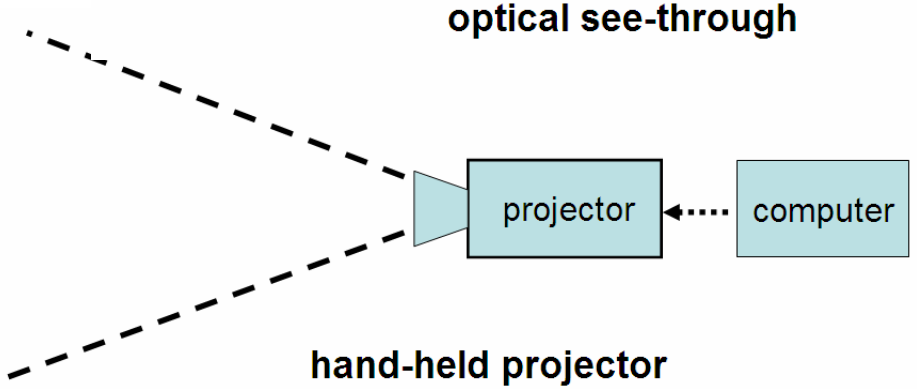
Hand-Held Displays



video see-through



optical see-through



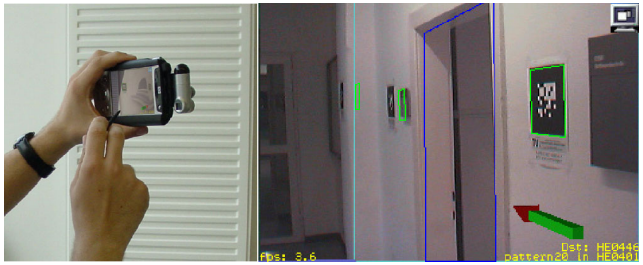
hand-held projector

Hand-Held Displays

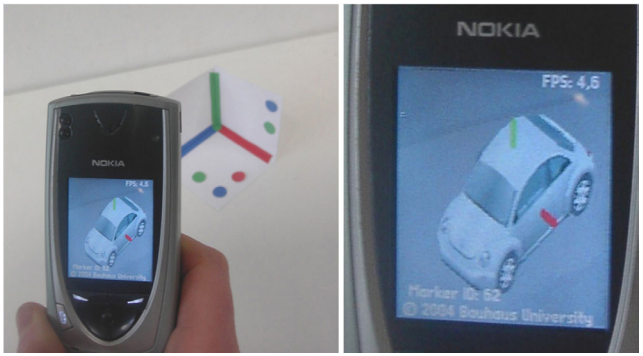
Pros:

- Realistic alternative to HMDs for mobile applications;
- Consumer devices such as PDAs and cellphones have the potential to address a mass-market;
- Support mobile applications;
- Support multi-user applications.

www.uni-weimar.de/medien/AR



Wagner, D, et al. First steps towards handheld augmented reality. In proceedings of International Conference on Wearable Computers, pp. 127-136, 2003.



Moehring, et al.
Video See-Through and Optical Tracking with Consumer Cell Phones
In proceedings of International Symposium on Augmented and Mixed Reality (ISMAR'04), 2004.

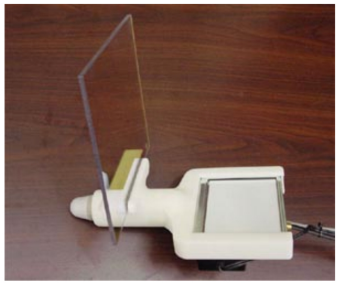
Cons:

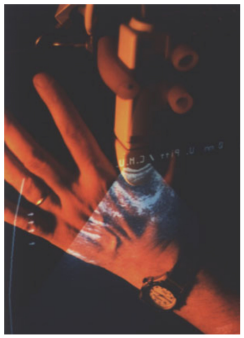
- Image analysis and rendering components is processor and memory intensive;
- Limited screen size of most hand-held devices restricts the covered field-of-view;
- Optics and image sensor chips of integrated cameras in consumer hand-held devices is targeted to other applications and consequently provide a limited quality for image processing tasks;
- Compared to head-attached devices, hand-held devices do not provide a completely hands-free working.

Hand-Held Displays

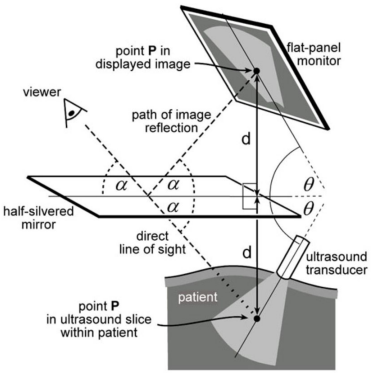
Pros:

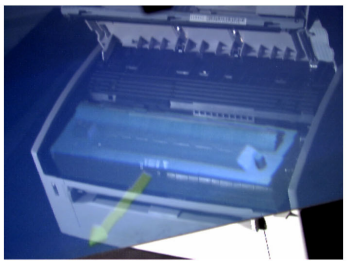
- Realistic alternative to HMDs for mobile applications;
- Consumer devices such as PDAs and cellphones have the potential to address a mass-market;
- Support mobile applications;
- Support multi-user applications.





Stetten, et al. Real Time Tomographic Reflection: Phantoms for Calibration and Biopsy, In proceedings of IEEE/ACM International Symposium on Augmented Reality (ISMAR'01), pp. 11-19.
US and foreign patents are pending





Bimber, et al. Augmented Reality with Back-Projection Systems using Transflective Surfaces. Computer Graphics Forum (proceedings of EUROGRAPHICS 2000 - EG'2000), vol. 19, no. 3, pp.161-168, 2000.

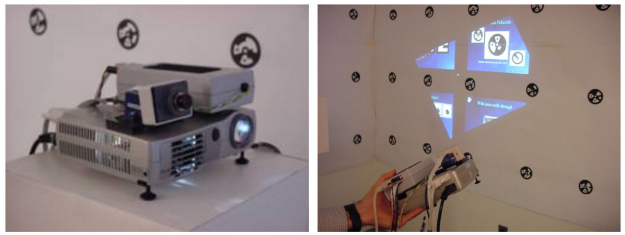
Cons:

- Image analysis and rendering components is processor and memory intensive;
- Limited screen size of most hand-held devices restricts the covered field-of-view;
- Optics and image sensor chips of integrated cameras in consumer hand-held devices is targeted to other applications and consequently provide a limited quality for image processing tasks;
- Compared to head-attached devices, hand-held devices do not provide a completely hands-free working.

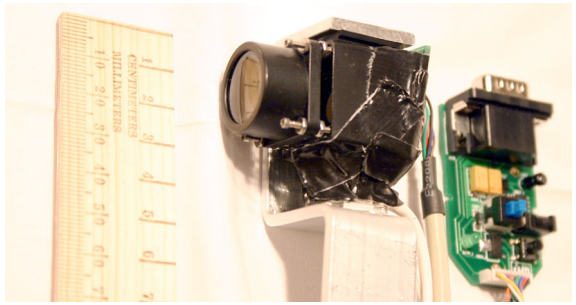
Hand-Held Displays

Pros:

- Realistic alternative to HMDs for mobile applications;
- Consumer devices such as PDAs and cellphones have the potential to address a mass-market;
- Support mobile applications;
- Support multi-user applications.



Raskar, et al. iLamps: Geometrically Aware and Self-Configuring Projectors. In Computer Graphics (proceedings of SIGGRAPH'03), 2003.



Raskar, et al. RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors. SIGGRAPH '04, 2004.

Cons:

- Image analysis and rendering components is processor and memory intensive;
- Limited screen size of most hand-held devices restricts the covered field-of-view;
- Optics and image sensor chips of integrated cameras in consumer hand-held devices is targeted to other applications and consequently provide a limited quality for image processing tasks;
- Compared to head-attached devices, hand-held devices do not provide a completely hands-free working.

Screen-Based Video See-Through Displays

Pros:

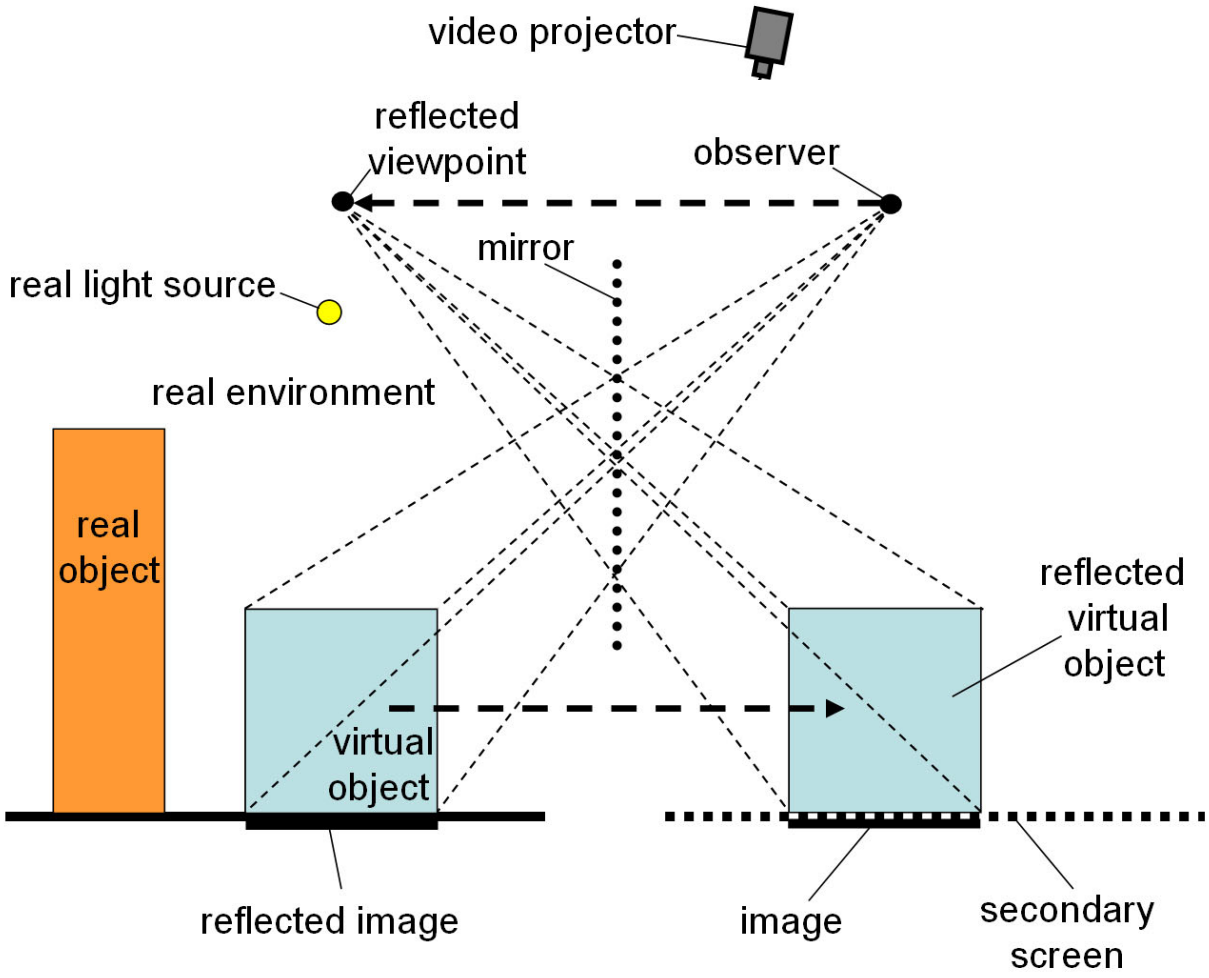
- Common technique if mobile application does not have to be supported;
- Cost-efficient and easy to realize with off-the-shelf components.

www.uni-weimar.de/medien/AR

Cons:

- Small field of view - screen-size is scalable if projection displays are applied;
- Limited resolution (especially of the real environment);
- Does not support mobile applications;
- Does not support multiple users;
- Does not support stereo;
- Mostly provides a remote viewing, rather than supporting a see-through metaphor;
- Direct interaction with the real environment and the graphical augmentation is usually not supported.

Spatial Optical See-Through Displays



Spatial Optical See-Through Displays

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors;
- An easier and more stable calibration;
- Better controllable environment.

Bimber, et al. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55,2001.

Bimber, et al. The Extended Virtual Table: An Optical Extension for Table-Like Projection Systems. Presence: Teleoperators and Virtual Environments, vol.10, no. 6, 2001, pp. 613-631.

Cons:

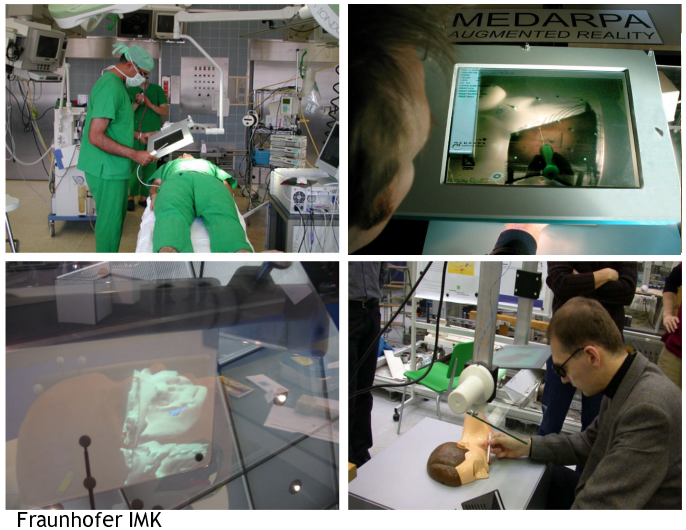
- Do not support mobile applications;
- In most cases, the applied optics prevents a direct manipulative interaction;
- Number of users is restricted;
- Mutual occlusion between real and virtual environment is normally not supported;
- Window violation (clipping of displayed image at screen / optics) is possible.

Spatial Optical See-Through Displays

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors;
- An easier and more stable calibration;
- Better controllable environment.

Schwald, et al. A Flexible Tracking Concept Applied to Medical Scenarios Using an AR Window. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 261-262, 2002.



Fraunhofer IMK
(www.arsys-tricorder.de)

Cons:

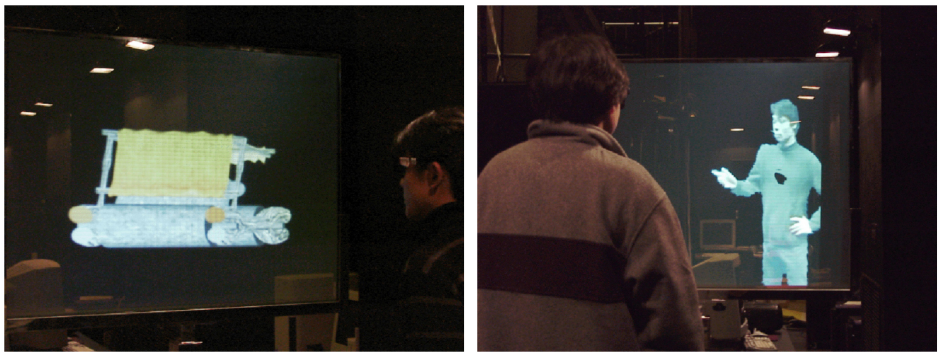
- Do not support mobile applications;
- In most cases, the applied optics prevents a direct manipulative interaction;
- Number of users is restricted;
- Mutual occlusion between real and virtual environment is normally not supported;
- Window violation (clipping of displayed image at screen / optics) is possible.

Spatial Optical See-Through Displays

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors;
- An easier and more stable calibration;
- Better controllable environment.

Laser Magic Productions, www.laser-magic.com



Ogi, et al. Invisible Interface for Immersive Virtual World. In proceedings of the Immersive Projection Technology Workshop (IPT'01), pp. 237-246, Stuttgart, Germany, 2001.

Cons:

- Do not support mobile applications;
- In most cases, the applied optics prevents a direct manipulative interaction;
- Number of users is restricted;
- Mutual occlusion between real and virtual environment is normally not supported;
- Window violation (clipping of displayed image at screen / optics) is possible.

Spatial Optical See-Through Displays

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors;
- An easier and more stable calibration;
- Better controllable environment.

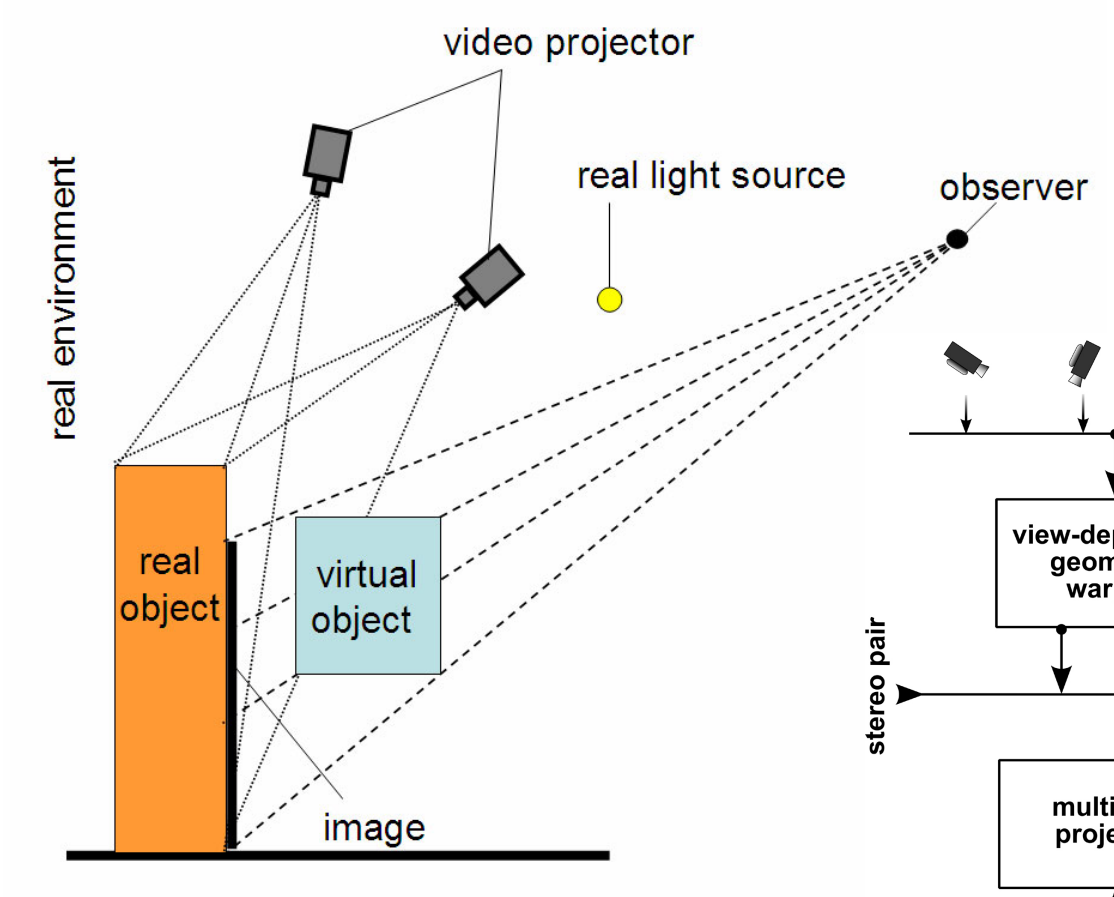
Fogscreen Inc., http://www.fogscreen.com/

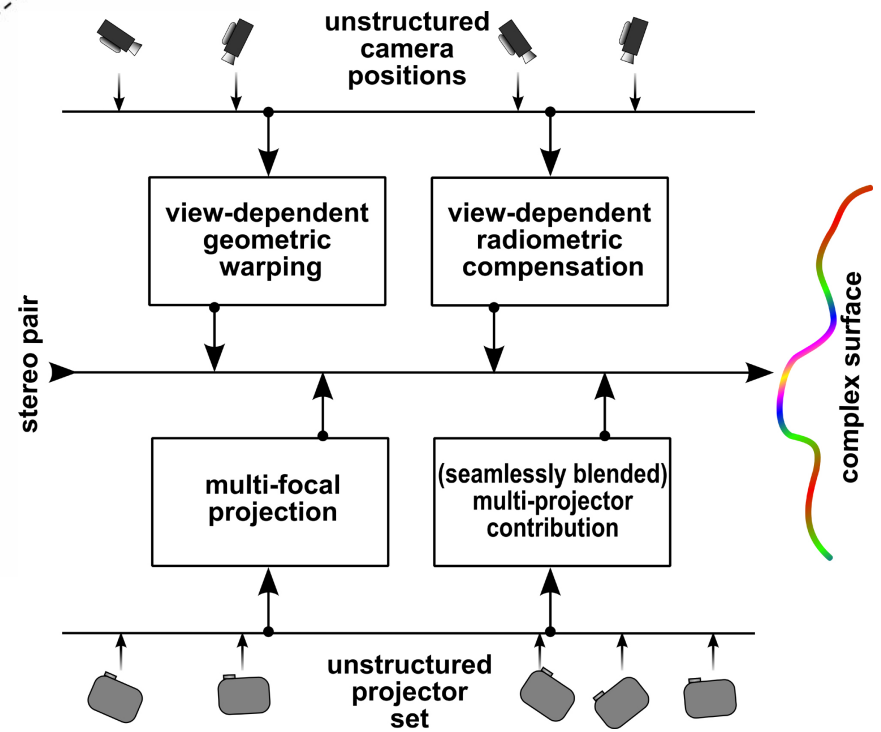
IO2 Technology, http://www.io2technology.com/

Cons:

- Do not support mobile applications;
- In most cases, the applied optics prevents a direct manipulative interaction;
- Number of users is restricted;
- Mutual occlusion between real and virtual environment is normally not supported;
- Window violation (clipping of displayed image at screen / optics) is possible.

Projection-Based Augmentation

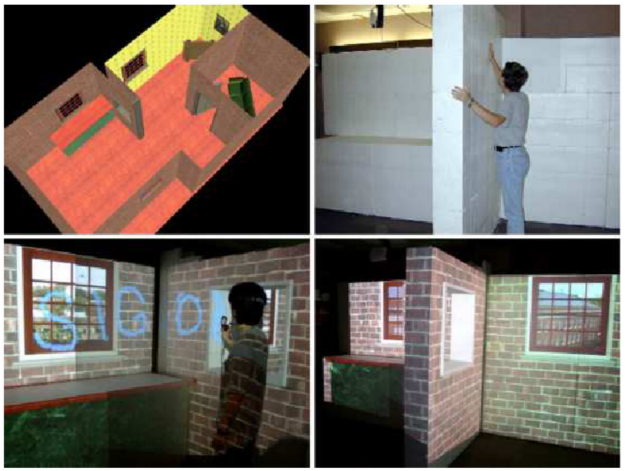




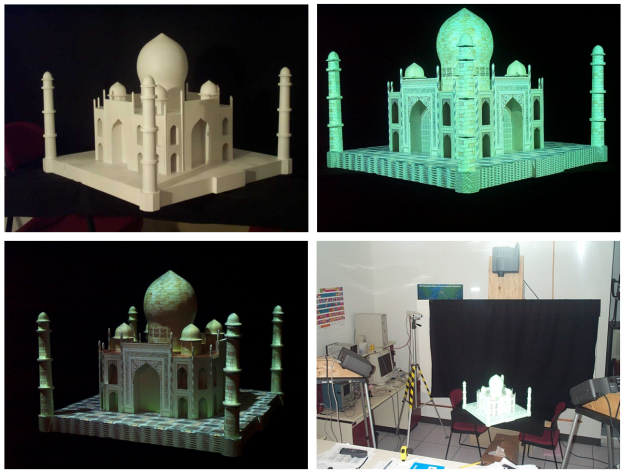
Projection-Based Augmentation

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors.



Low, et al. Life-Sized Projector-Based Dioramas. Symposium on Virtual Reality Software and Technology, 2001.



Raskar, et al. Shader Lamps: Animating real objects with image-based illumination. In Proceedings of Eurographics Rendering Workshop (EGRW'01), 2001.

Cons:

- Do not support mobile applications;
- Shadow-casting because of front-projection;
- Restrictions of the display area (size, shape, and color);
- Restricted to a single user in case of non-zero parallax;
- Conventional projectors can only focus on a single focal plane;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

Projection-Based Augmentation

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors.

Pinhanez. The everywhere displays projector: A device to create ubiquitous graphical interfaces. In Proc. of Ubiquitous Computing 2001 (Ubicomp'01), Atlanta, Georgia, 2000.

Cons:

- Do not support mobile applications;
- Shadow-casting because of front-projection;
- Restrictions of the display area (size, shape, and color);
- Restricted to a single user in case of non-zero parallax;
- Conventional projectors can only focus on a single focal plane;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

Projection-Based Augmentation

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors.

Bimber, e al. Superimposing Pictorial Artwork with Projected Imagery, IEEE MultiMedia, pp. 16-26, 2005

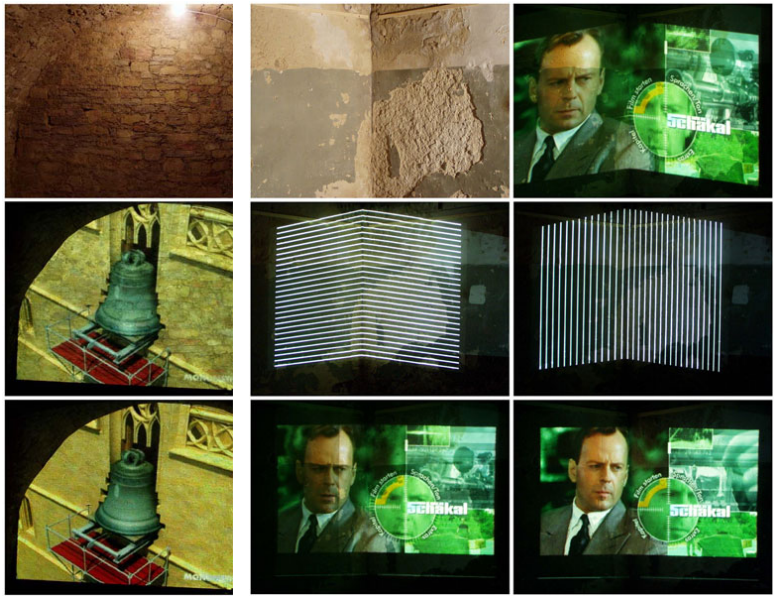
Cons:

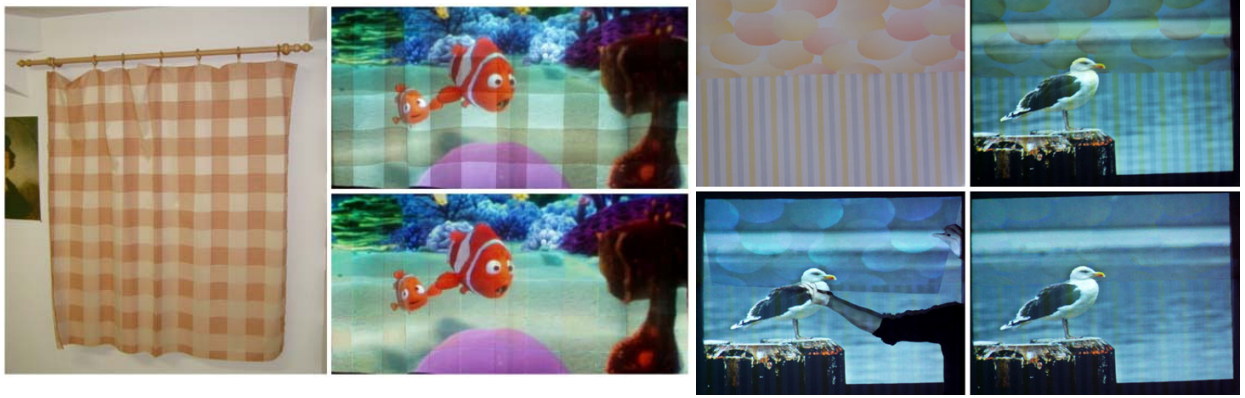
- Do not support mobile applications;
- Shadow-casting because of front-projection;
- Restrictions of the display area (size, shape, and color);
- Restricted to a single user in case of non-zero parallax;
- Conventional projectors can only focus on a single focal plane;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

Projection-Based Augmentation

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors.





Cons:

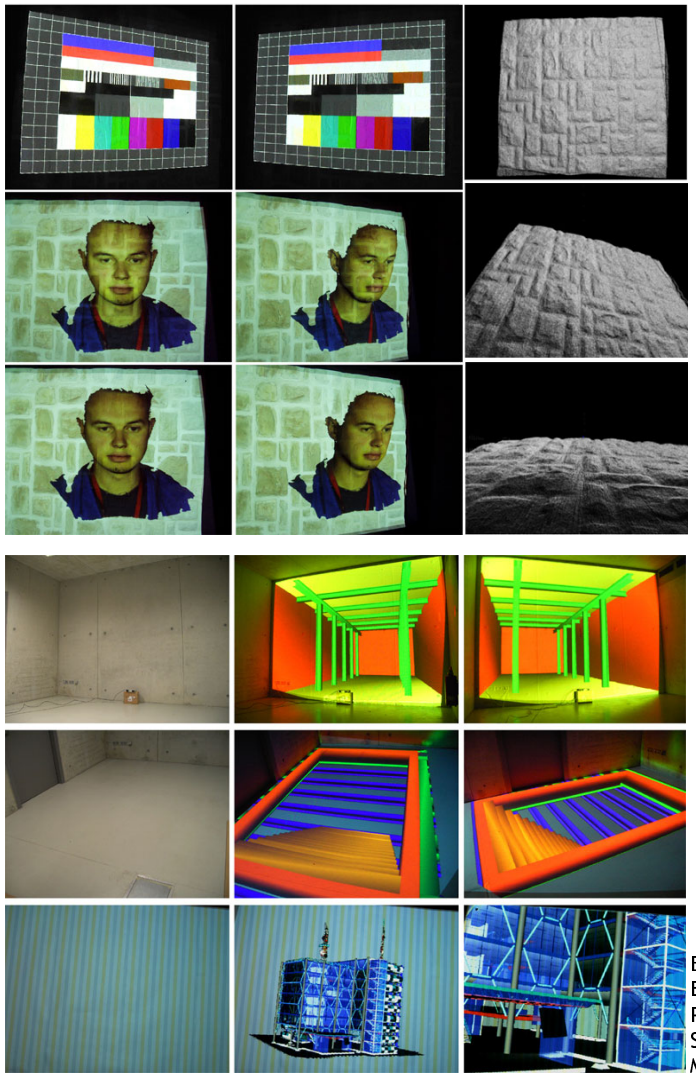
- Do not support mobile applications;
- Shadow-casting because of front-projection;
- Restrictions of the display area (size, shape, and color);
- Restricted to a single user in case of non-zero parallax;
- Conventional projectors can only focus on a single focal plane;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

Bimber. et al.
Embedded Entertainment with Smart Projectors
IEEE Computer, pp. 56-63, 2005

Projection-Based Augmentation

Pros:

- Easier eye accommodation and vergence;
- Higher and scalable resolution;
- Larger and scalable field of view;
- Improved ergonomic factors.



Cons:

- Do not support mobile applications;
- Shadow-casting because of front-projection;
- Restrictions of the display area (size, shape, and color);
- Restricted to a single user in case of non-zero parallax;
- Conventional projectors can only focus on a single focal plane;
- The complexity of consistent geometric alignment and color calibration increases with the number of applied projectors.

Bimber. et al.
Enabling View-Dependent Stereoscopic Projection in Real Environments
Submitted to the International Symposium on Mixed and Augmented Reality (ISMAR'05), 2005

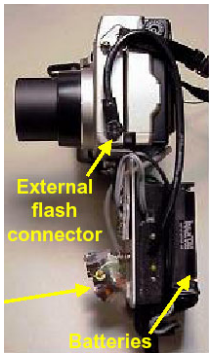
Enabling and Future Technologies

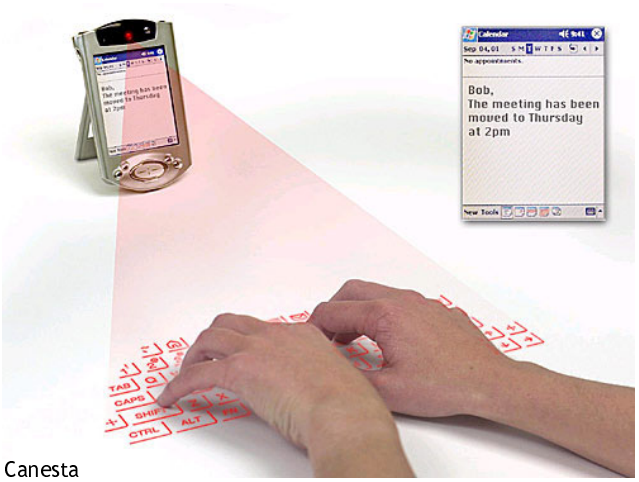
Future of Projectors

- Compact and portable;
- LEDs will replace lamps;
- Reflective (DLP, LCOS) will replace transmissive;
- Less power and less cooling;
- Extremely high frame-rates (up to 180Hz / 540Hz);
- Laser-projectors offer infinite focus and are unproblematic with ambient environment light;
- Multi-focal projection

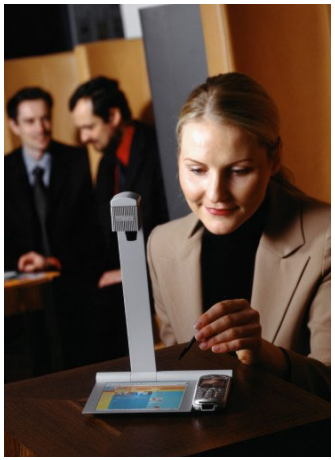
Symbol ET.

Strat, et al. M.M. Casual 3D Photography, Siggraph'03, Sketches and Applications





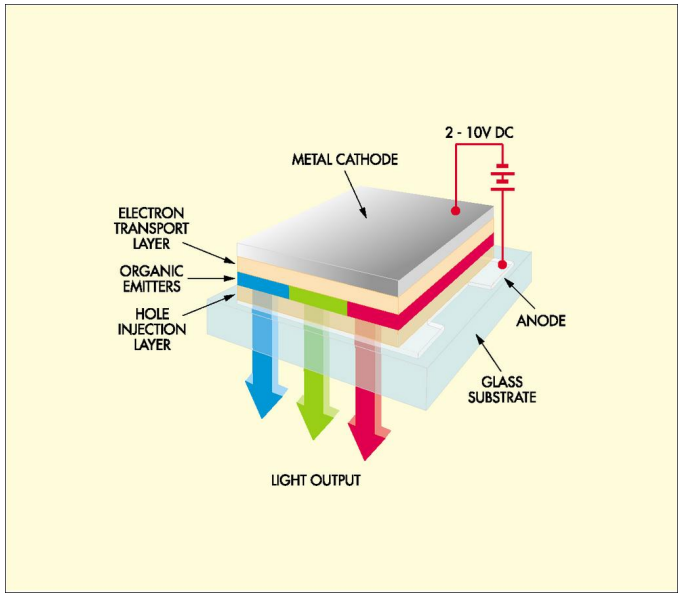
Canesta



Siemens

OLEDs

- Consist of organic film
  - carbon-base
  - polymer-based
- Emit light under voltage (emissive)
- Horizontal cathodes and vertical anodes
- Electricity flows in one direction only → LED (PN or NP)
- Don't require backlights (they are fluorescent)
- Large scale displays (500 inch displays or larger)
- High resolution (300 dpi or higher)
- Different types
  - PHOLEDs (phosphorescent OLEDs)
  - TOLEDs (transparent OLEDs)
  - FOLEDs (flexible OLEDs)
- Active and passive matrix
- Inkjet technology can be used to print OLEDs
- Low power consumption (currently LCDs require three times as much power than OLEDs)
- Thin layers (about a thousand times thinner than a human hair)
- Can be viewed from a much wider angle (don't polarize light)
- Degrade over time (esp. different colors)

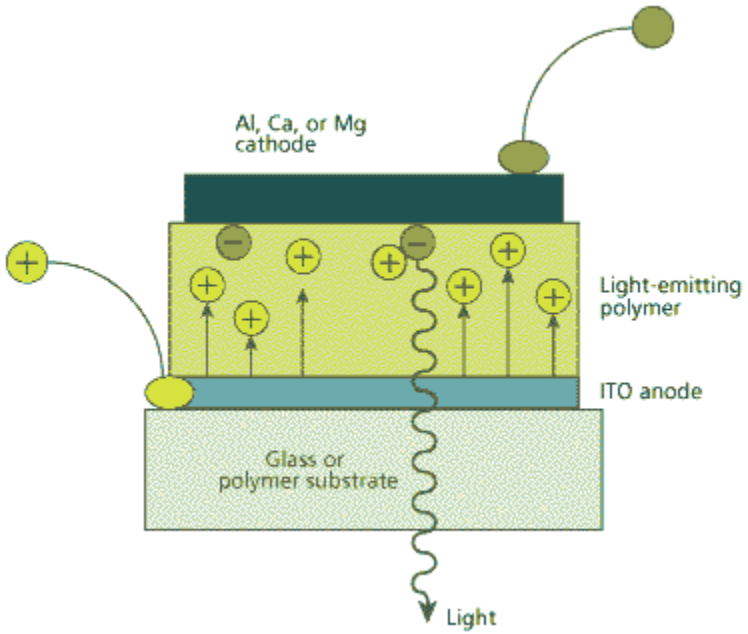


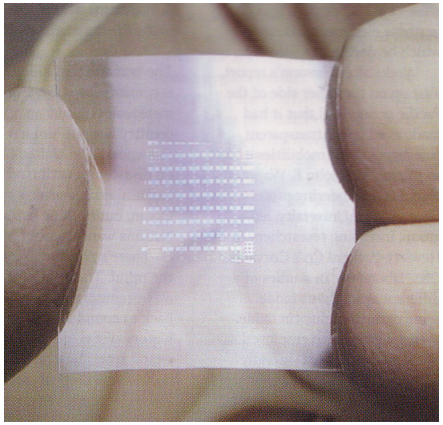
Silicon Chip

Visionox Technology

LEPs

- Thin film of light-emitting polymer (LEPs) is applied onto a glass or plastic substrate coated with a transparent electrode (glass or plastic)
- Ultra-fast responds time (sub-microsecond → may support active stereo!)
- Flexible (different shapes possible)
- Low power consumption
- PLEDs provide a high contrast (currently between 3-5 times higher than LCDs)
- Full color spectrum possible



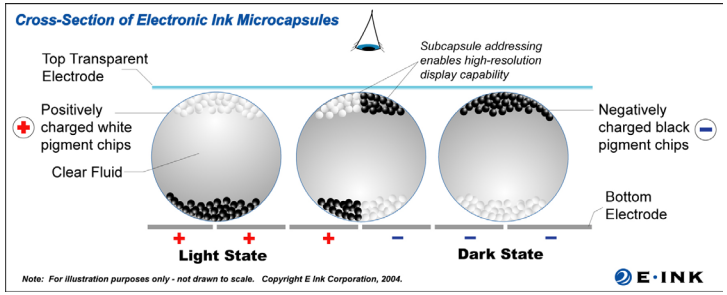


Tokyo Institute of Technology:
transparent transistors

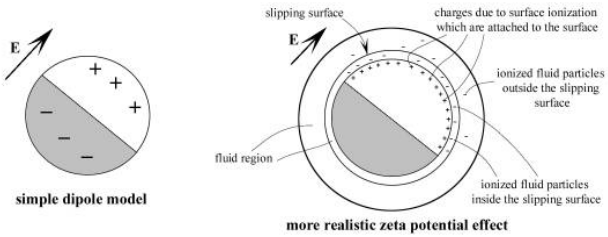
Electronic Paper / E-Ink

- + (black) and – (white) charged elements between two electrodes
- Elements can be rotated or moved
- Depending on polarization of electricity, black or white elements move or rotate to visible surface
- Bi-stable: no electricity is needed to keep state (only to change state)
- Active or passive matrix
- Very low power required (interesting for mobile devices)
- May allow to build flexible displays (active / passive matrix must be flexible too)
- Grayscales: degree of rotation or ratio between black and white elements
- Color possible too

Toppan Printing Co., Ltd., E Ink Corporation

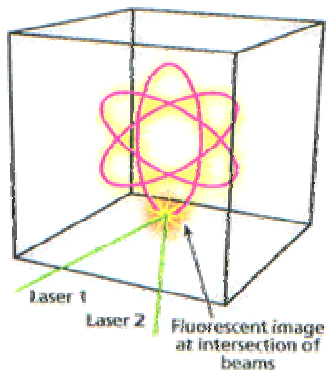


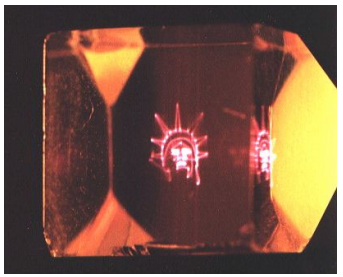
Xerox, PARC



Solid-State Displays

- Generate visible photons within a transparent host material by exciting optically active ions with special energy sources;
- Crossing energy beams (infrared lasers, ultraviolet sources of radiation, or electron beams);
- Host materials are various gases, crystals and electro-active polymers;
- Host material must have following properties:
  - In its initial state it must be transparent;
  - It must emit visible light in its excited state;
  - Its inner structure must be homogenous;
  - It must have a refraction index similar to air to avoid distortion.
- Conceptual problems, such as the "ghost voxel problem" (when energy beams that are used to create voxels intersect with other beams and create unwanted voxels) have to be solved;
- Not (yet) possible in air.





Downing, et al. A Three-Color, Solid-State, Three-Dimensional Display, Science, Vol. 273, pp. 1185-1189, 1996.

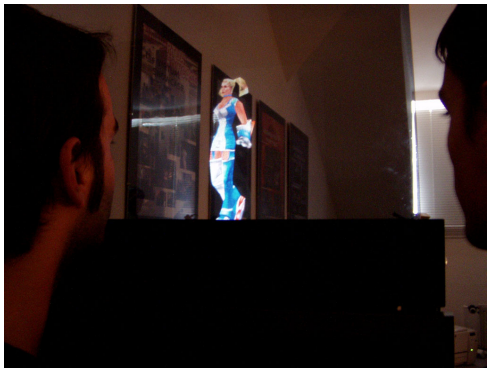
Volume Displays:

Langhans, et al. Solid FELIX: A static 3D-Laser Display, In Proceedings of Symposium on Electronic Imagining: Science and Technology (IS&T/SPIE), vol. 5006, 2003.

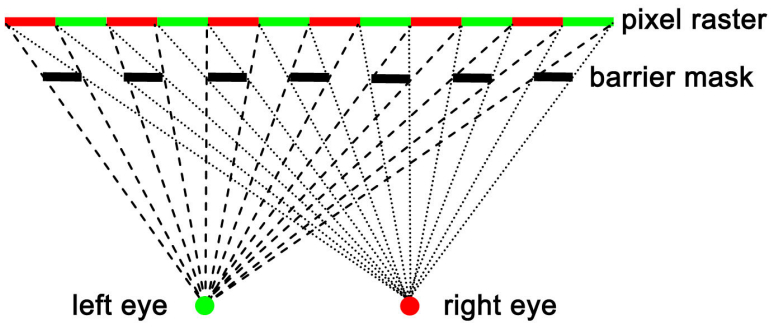
Jansson, et. al. A Three-Dimensional Computer Display, Computer Graphics in CAD/CAM Systems , Annual Conference, Cambridge, April 1979.

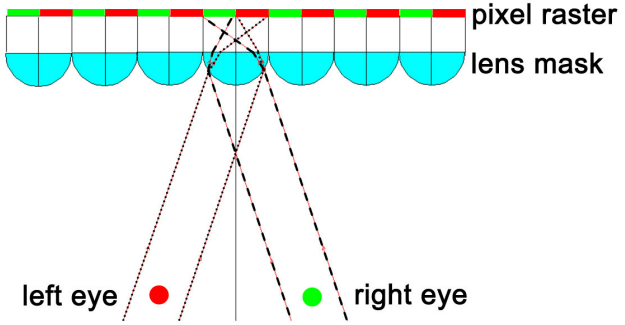
Parallax Displays

- Display screens (e.g., CRT or LCD displays) are overlaid with an array of light-directing elements for stereo-separation;
- Parallax barrier displays apply a controllable array of light-blocking elements (e.g., a light blocking film or liquid crystal barriers) in front of a CRT screen;
- Lenticular sheet displays that apply an array of optical elements (e.g., small cylindrical or spherical lenses) to direct the light for a limited number of defined viewing-zones;
- Commercially available;
- Multi-user capable.



www.uni-weimar.de/medien/AR



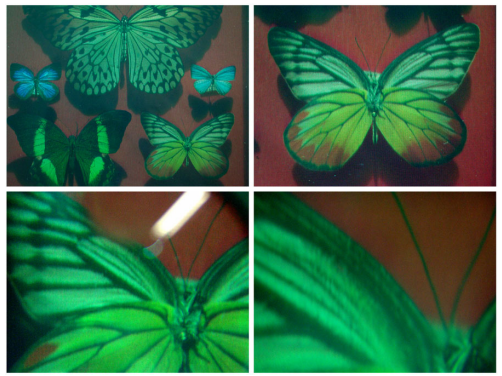


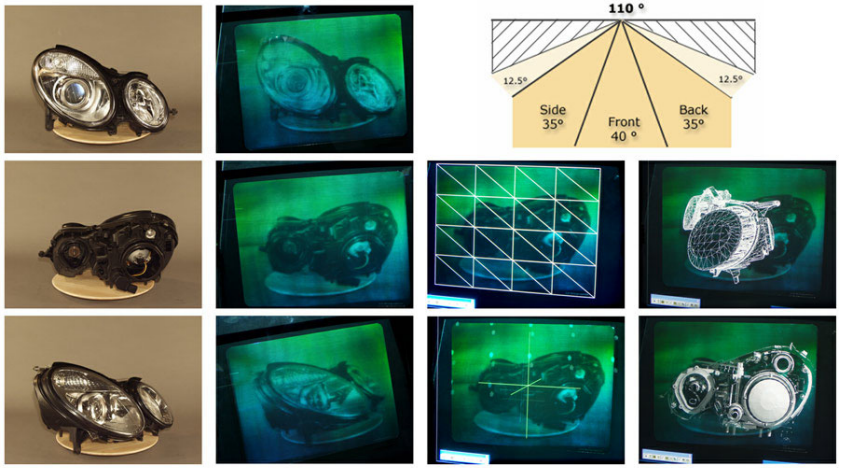
Holography

- Different types: optical, digital, electro-holograms;
- Ultra-high resolution (Terra-Pixels);
- Ultra-high information content (Terra-Bytes);
- Reconstruct entire wavefronts;
- Support all depth queues and an unlimited number of users;
- Optical Holograms
  - Analog (no computational power required for display);
  - Are static;
  - Can be combined with interactive computer graphics.

Example for electro-hologram

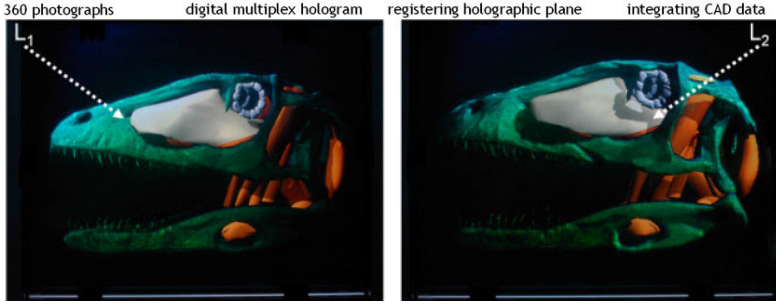
Courtesy: QinetiQ



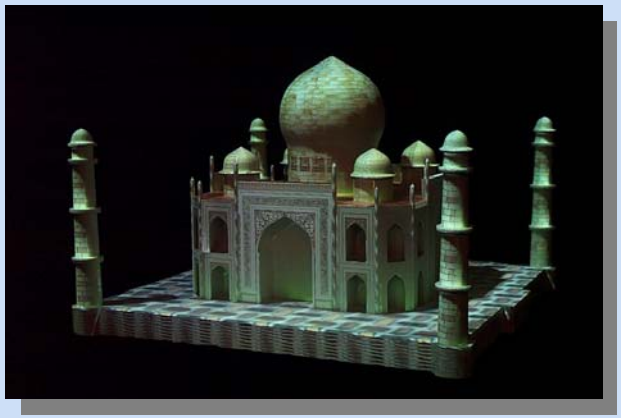


360 photographs | digital multiplex hologram | registering holographic plane | integrating CAD data

Bimber Combining Optical Holograms with Interactive Computer Graphics IEEE Computer (Cover Featuer), pp. 85-91, January 2004



Spatial Augmentation



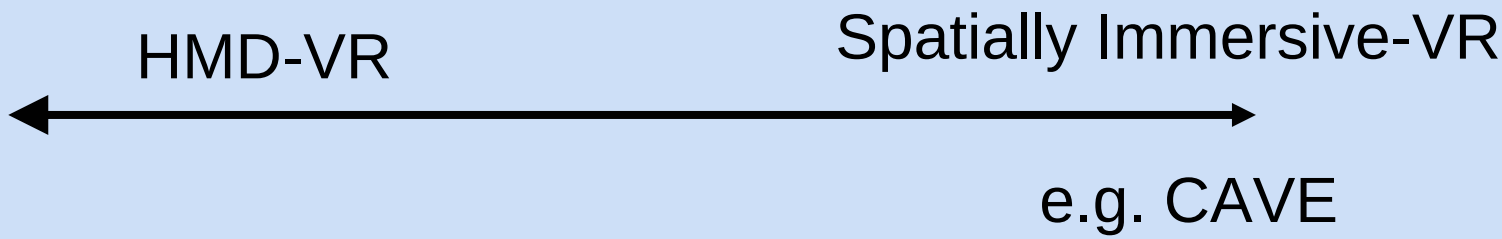
http://www.cs.unc.edu/~raskar/Projector/

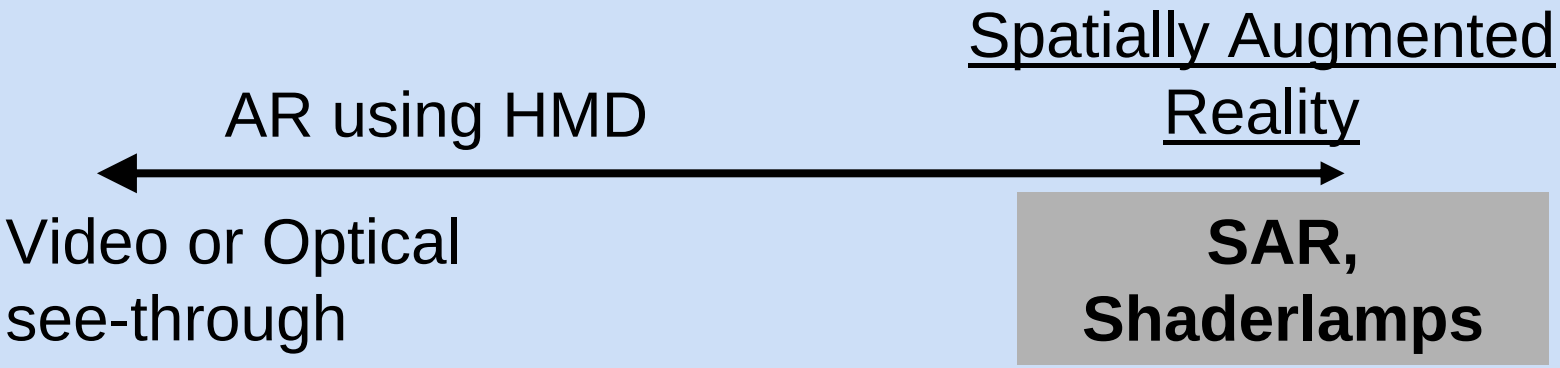
Ramesh Raskar
Mitsubishi Electric Research Labs
Cambridge, MA

Projector-based AR
Outline

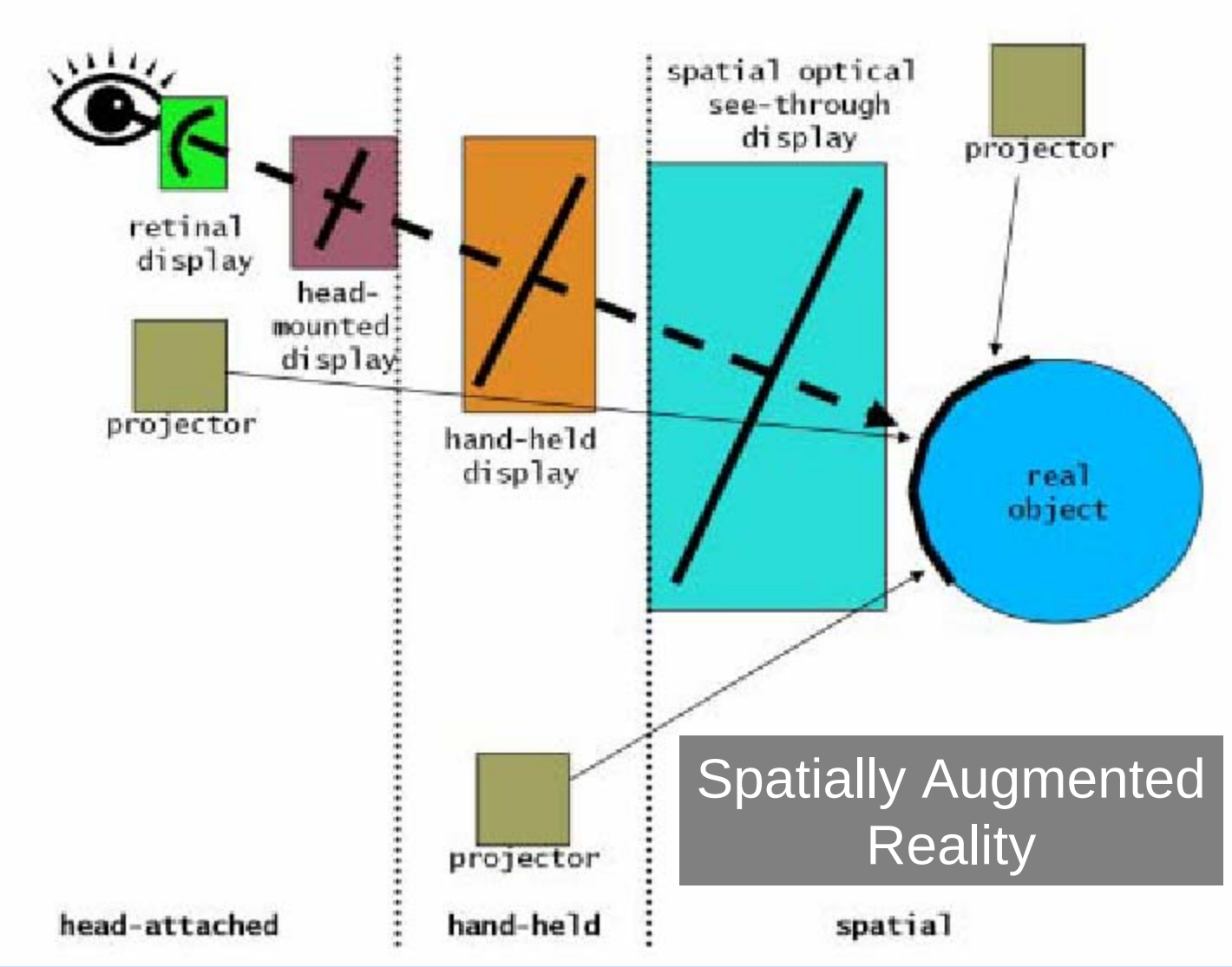
- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

Spatially Augmented Reality (SAR)





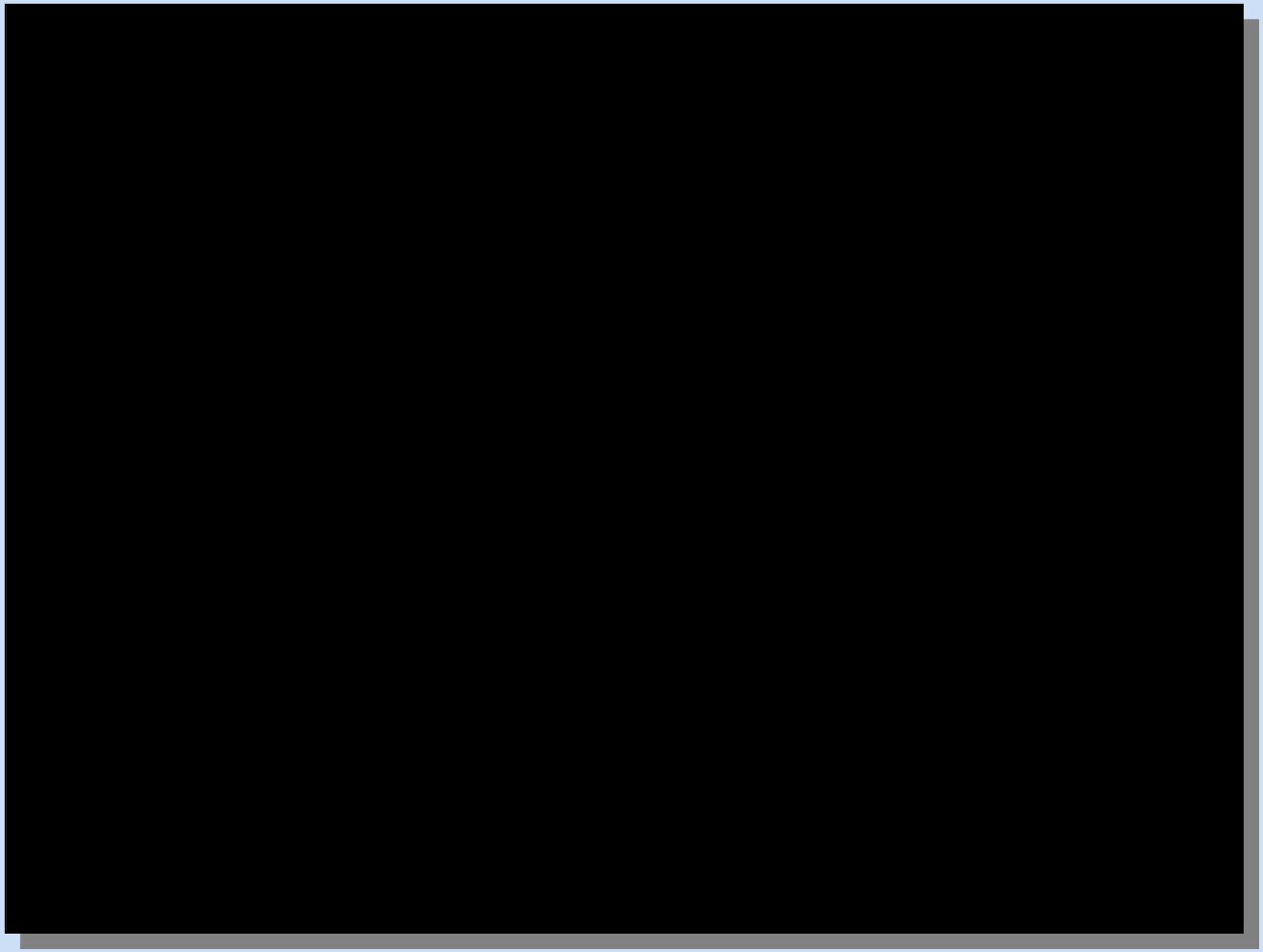
Classification of AR



Spatially Augmented Reality

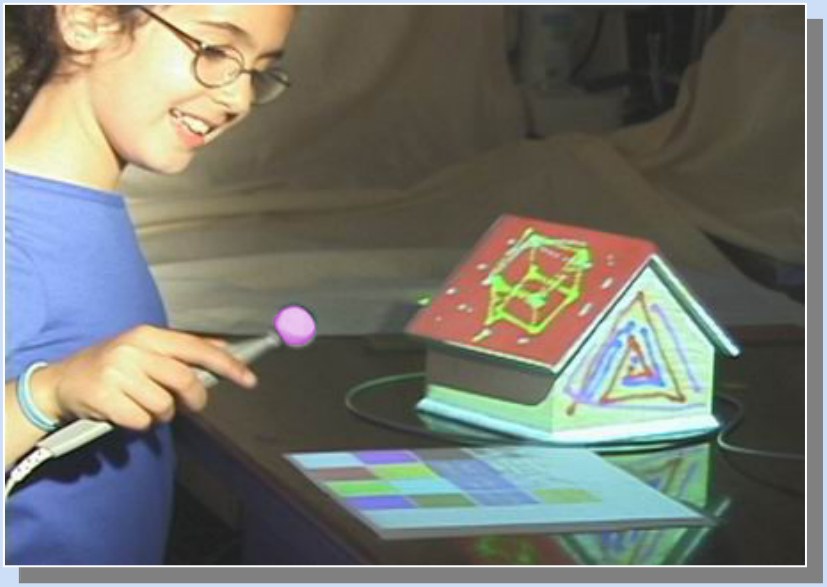
Raskar, vanBaar, Beardsley, Willwacher, Rao, Forlines
‘iLamps: Geometrically Aware and Self-Configurable Projectors’,
SIGGRAPH 2003

Prototype Handheld Projector

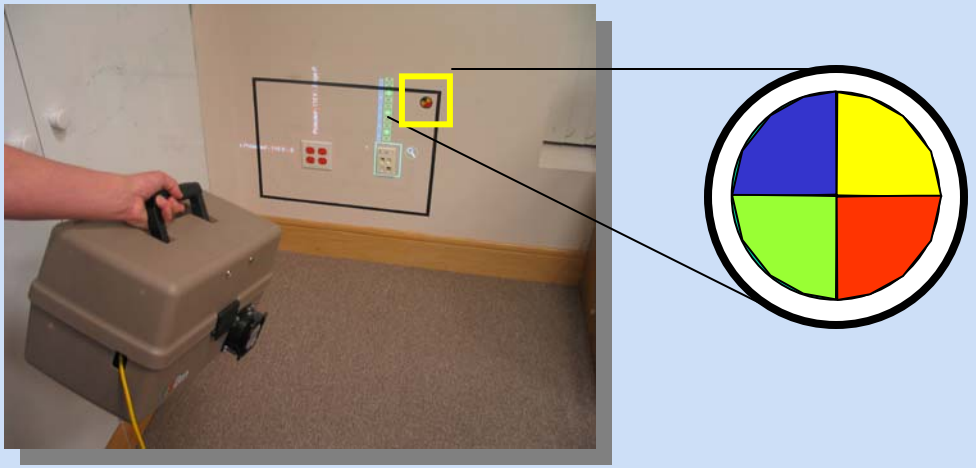


Non-trivial Projection Screens

- Painting with Light
  - [Bandyopadhyay, Raskar, Fuchs 2001]



AR Issues

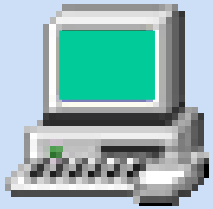


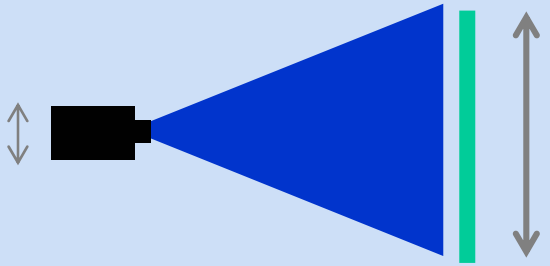
- Preprocessing:
  - Authoring
- Runtime:
  - **Identification:** Recognition of objects
    - Using markers and visual tags
  - **Registration:** Finding relative pose of display device
    - Dynamic estimate of translation and rotation
    - Render/Warp images
  - **Interaction:**
    - Widgets, Gesture recognition, Visual feedback

Advantages of Projectors

- Size of image

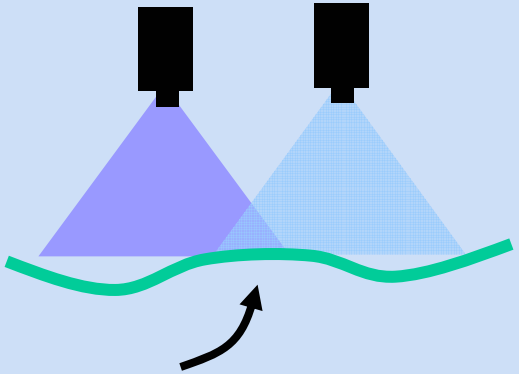
  Image can be larger than device





- Combination of images

  Images can be superimposed and added



- Shape of display surface

  Displayed images may be non-planar

Disadvantages

- Projector limitations
  - Limited depth of field
  - Shadows
  - Affected by display surface reflectance
- Challenges
  - Calibration required
  - Rendering involves complex relationships

Advantages of Spatial Augmentation (SAR)

- Augmentation of objects not view
- Wide area, High resolution

Comparison

- Body-Worn Displays
  - Better ergonomics
  - Reduced tracking requirements
- Hand-held Displays
  - Avoids ‘last foot’ problem

Visually Rich Mediums

- Objects with Shape and Appearance
  - Not just CG
  - Statues and sculptures
  - Architectural tabletop models
  - Miniature sets for movies
  - Clay prototypes (cars)

Real + Virtual

- Real
  - Intuitive interface
  - Walk to move, zoom
- Virtual
  - Easy shape and color manipulation
  - Extreme views, undo

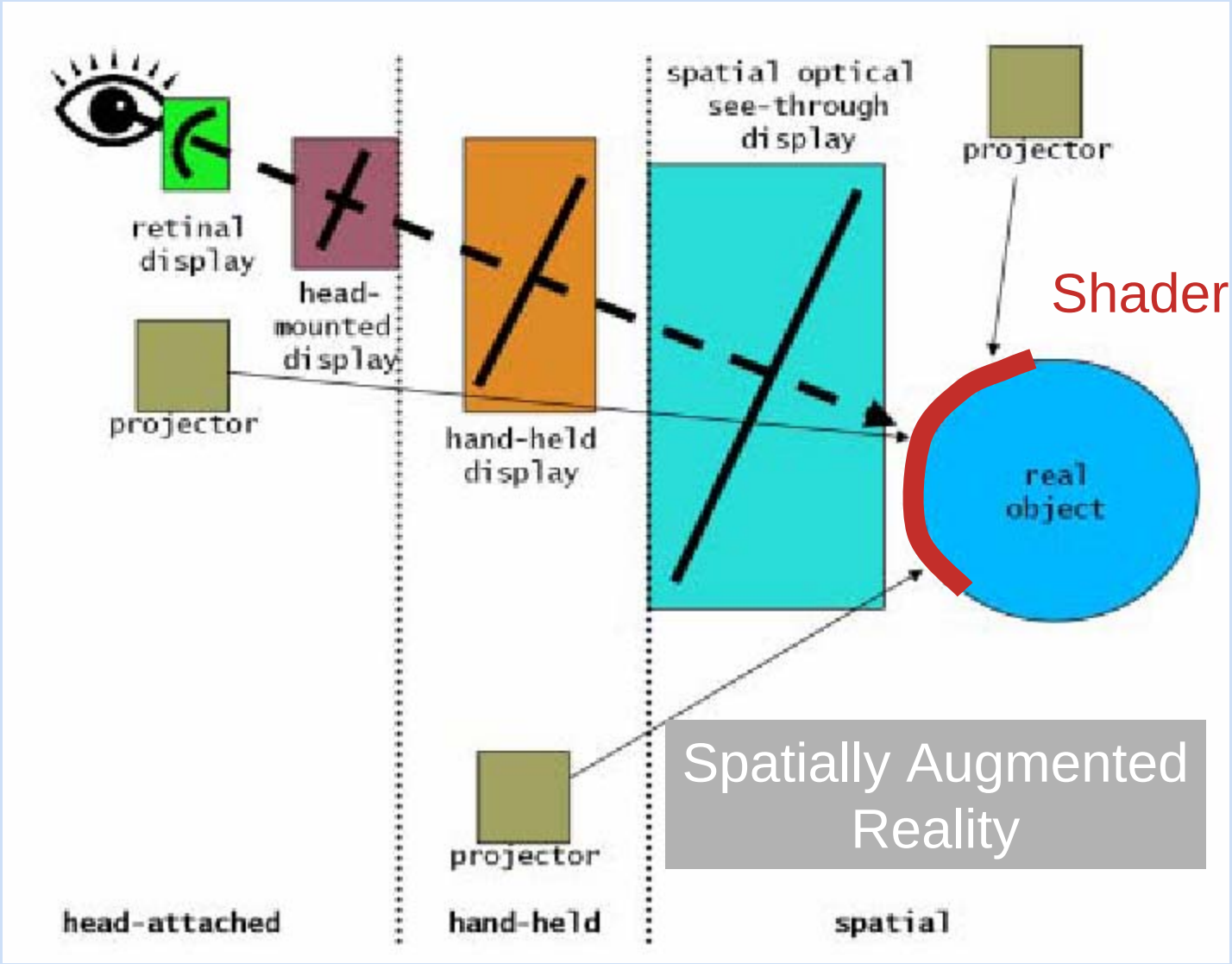
Real + Virtual

- Fidelity params
  - View (framerate, FOV, focus, stereo)
  - Shape (accuracy)
  - Color (spatial resolution, reflectance fidelity)
- Spatially Augmented Reality
  - Maintain high view fidelity
  - Approximate shape
  - Controlled reduction in color

Projector-based AR
Outline

- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

Classification of AR

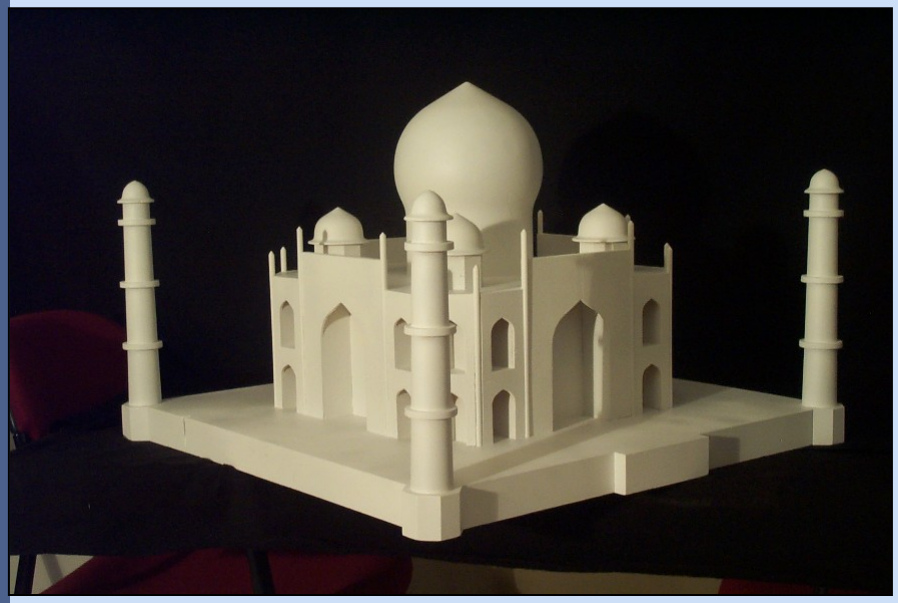


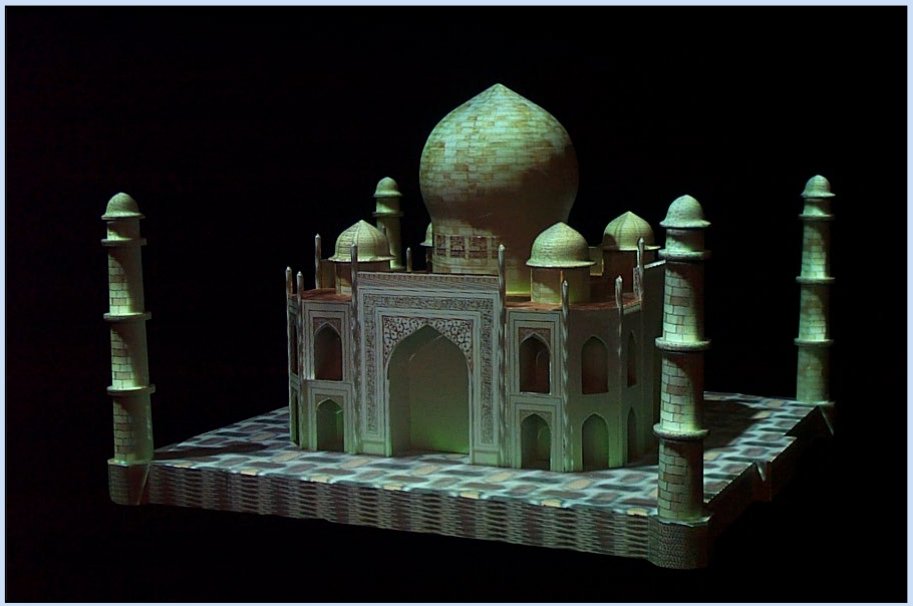
Shader Lamps
Motivation

View-dependent Appearance

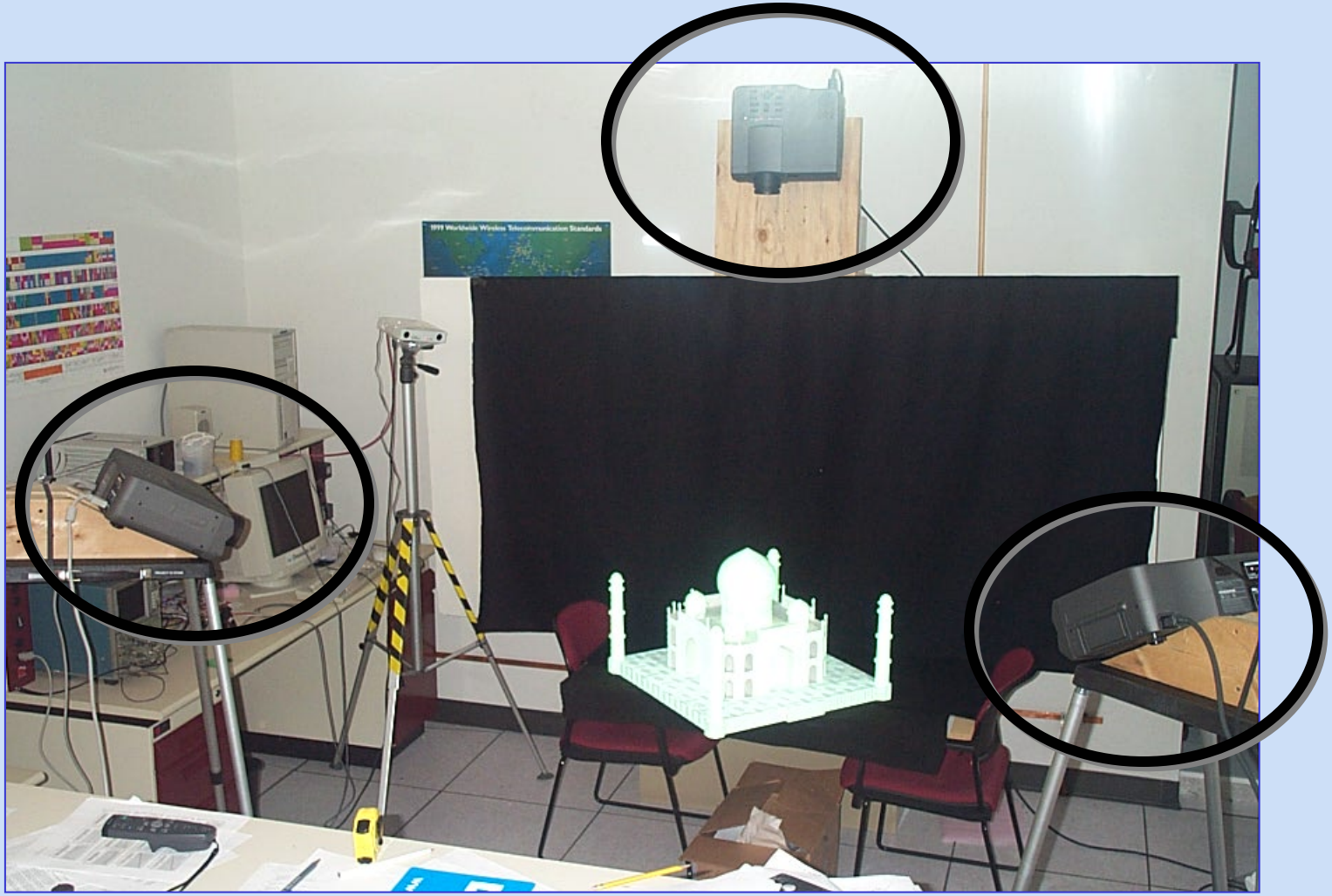
Shader Lamps
Image based Illumination

- Basic Idea
  - Render images and project on objects
  - Multiple projectors
  - View and object dependent color

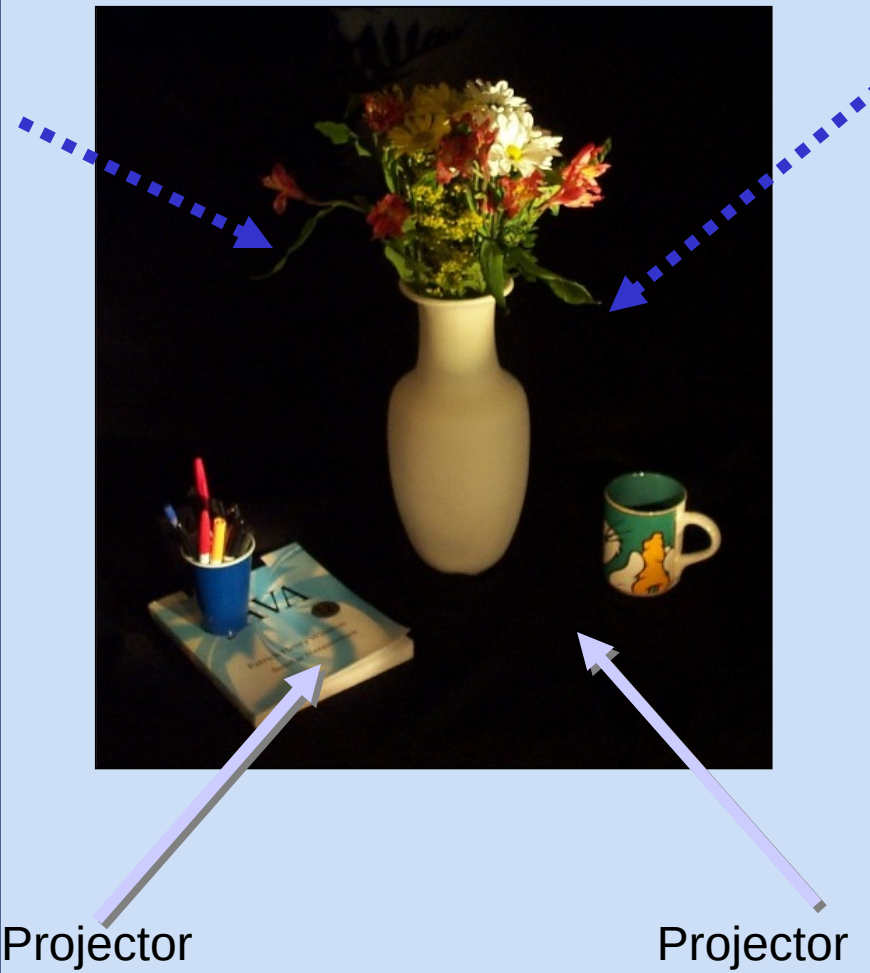




Raskar, Welch, Low, Bandyopadhyay, “Shader Lamps: Animating Real Objects with Image Based Illumination,” Eurographics Rendering Worksop (EGRW 2001)



Changing Appearance



Applications

Indoors, under controlled lighting

- Architectural models
  - Augment walk-around scaled model of buildings
  - Project and ‘paint’ surface colors, textures
  - Lighting, sunlight, seasons
  - Internal structure, pipes, wiring
- Assembly line
  - Instructional text, images and procedures
- Entertainment
  - Live shows, exhibits, demonstrations

Examples

- Son et Lumiere

Projecting slide of augmented photo

Examples

- Disney's Haunted Mansion

  Pre-recorded video

Singing busts

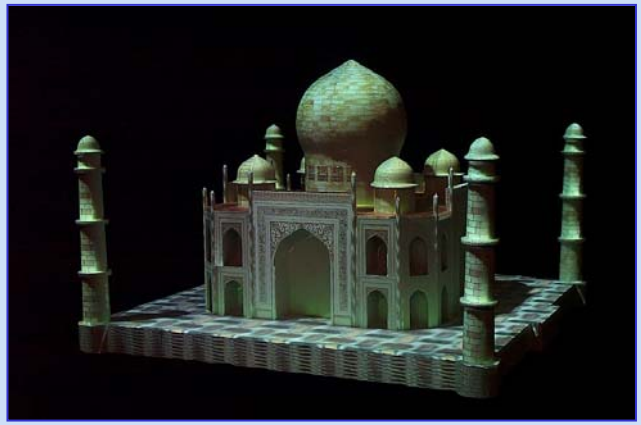
Madame Leota

- Old
  - Large, rigid installations
  - A 2D image or video projection
  - Single projector
  - Texture

- New
  - Easy setup, Non-trivial objects
  - Real time 3D animation
  - Multiple projectors
  - BRDF

Challenges

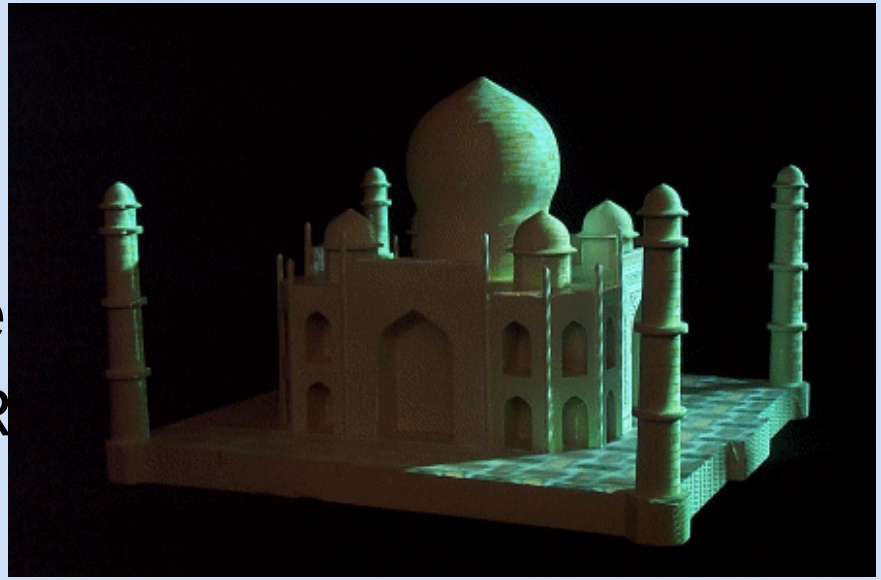
- Complete illumination
  - Image alignment
  - Special effects
    - Changing appearance and lighting
  - Complex geometry, self-occlusions
  - Merging multiple projectors



Comments ? Questions ?

Projector-based AR

Outline



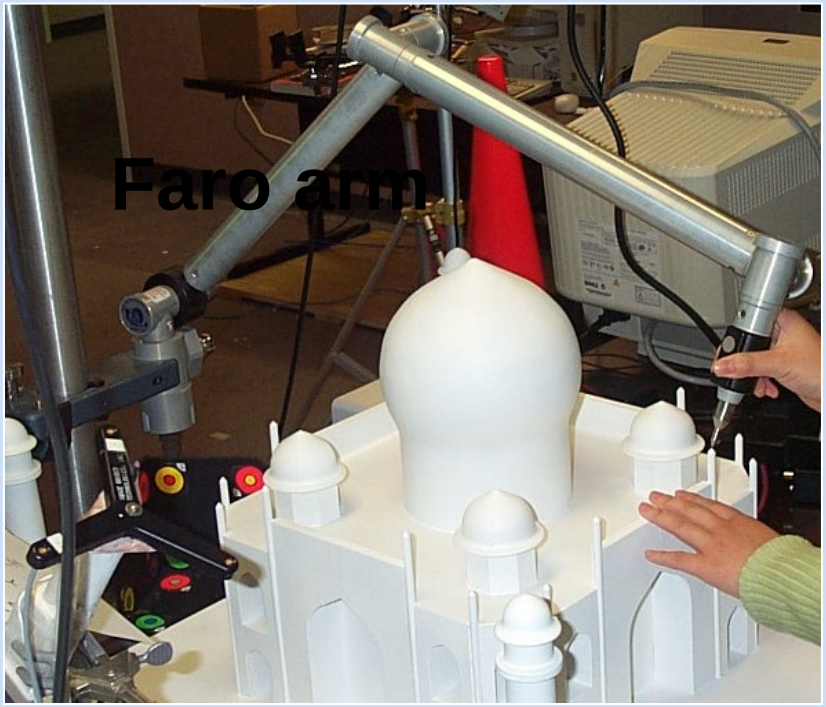
- Concepts and Hardware
  - Spatially Augmented R
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

Steps

- Preprocessing
  - Scan 3D object and create model
  - Approximately position projector(s)
  - Compute pose, ***P***
    - Find features
    - Find pixels that illuminate them
  - Compute intensity correction
- Run time
  - Render images of 3D model
  - Intensity correction for object shape
  - Feathering for projector overlap

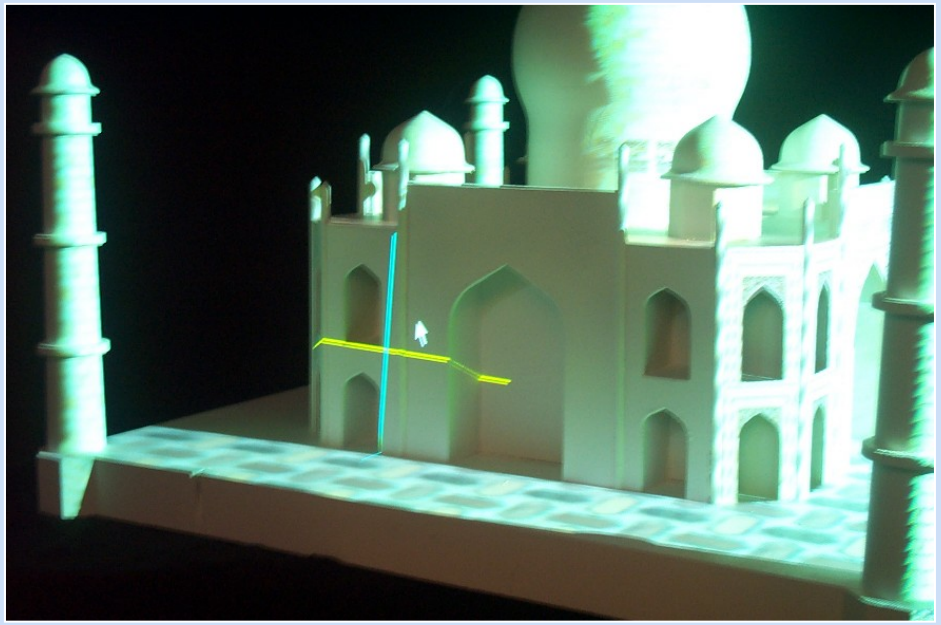
Steps

- Preprocessing
  - Scan 3D object and create virtual model, ***G***



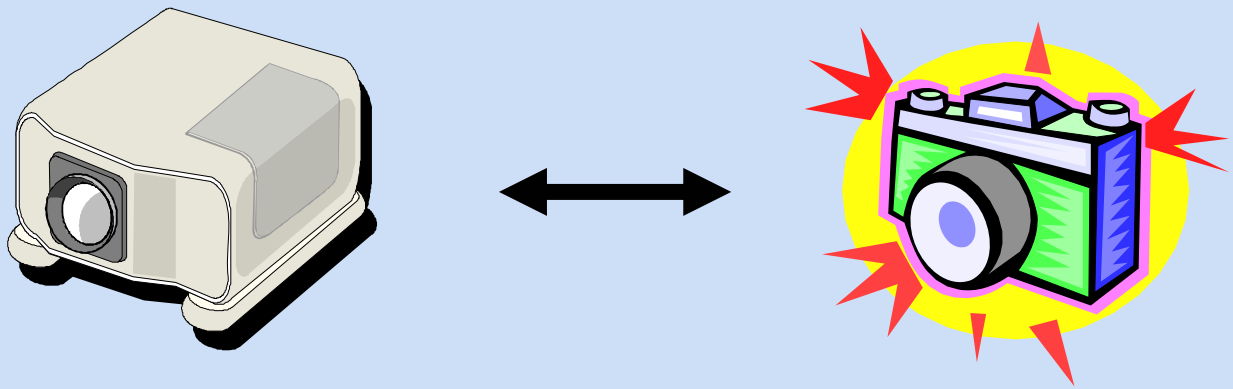
Steps

- Preprocessing
  - Scan 3D object and create model, ***G***
  - Approximately position projector(s)
  - Find pose, ***P***



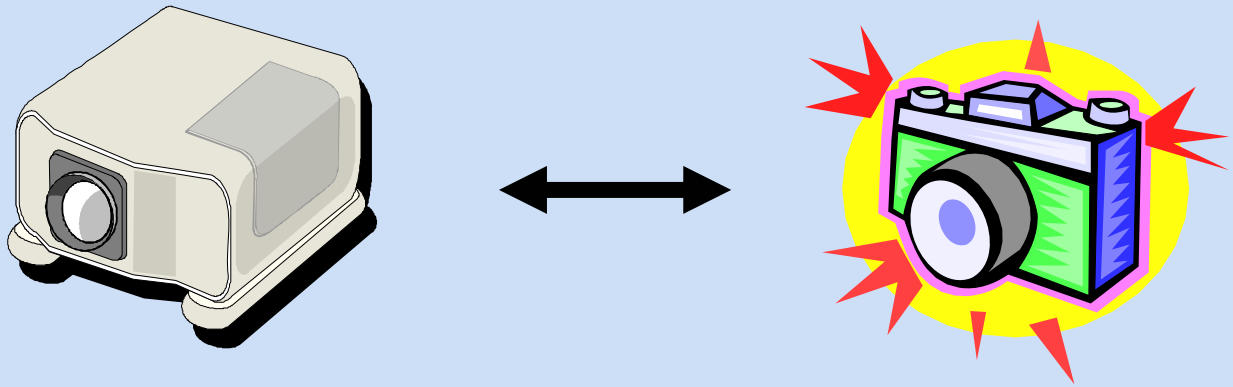
Motivation

- Projector - a 3D projection device
  - Projector is a dual of a camera
  - Relates 3D space and image in framebuffer
  - A useful abstraction : geometric projection model



Projector Model

- Pin hole model
  - Equations for perspective projection
  - Relationship between 3D and 2D
  - Intrinsic and Extrinsic Parameters



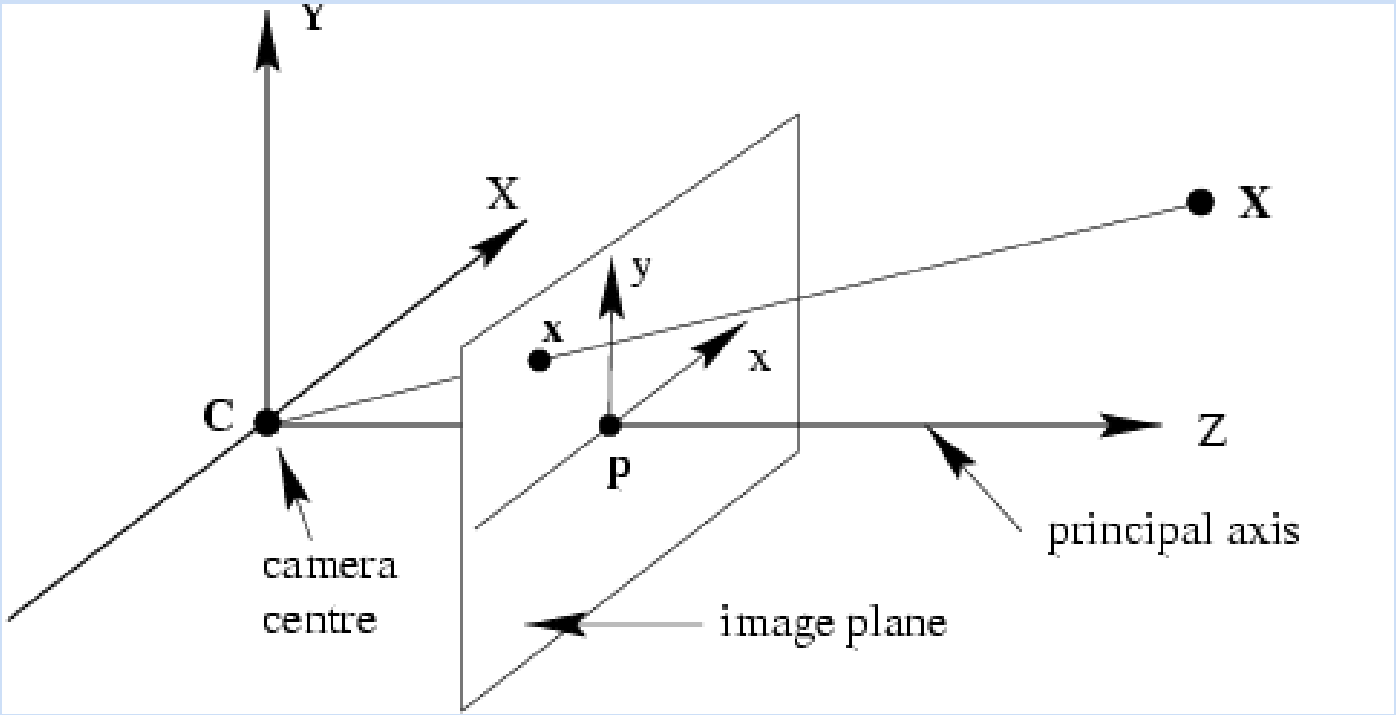
Camera (and Projector) anatomy

Camera center

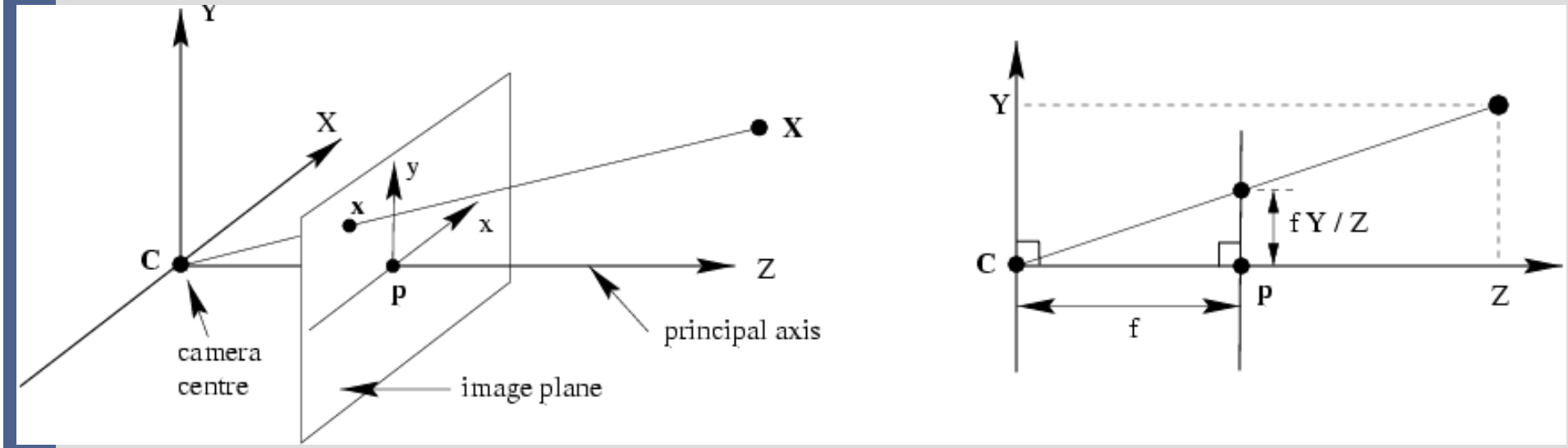
Image plane

Principal point

Principal axis



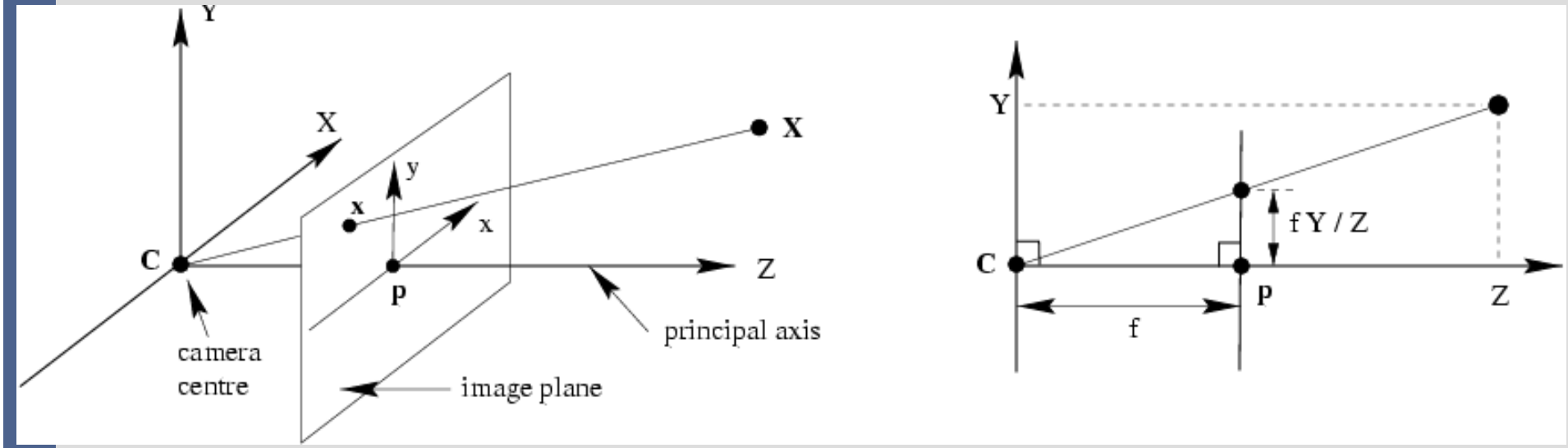
Pinhole camera (or Projector) model



$$y = f\frac{Y}{Z}$$

$$\begin{bmatrix} fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} \Rightarrow \begin{bmatrix} y \\ 1 \end{bmatrix} = \begin{bmatrix} fY \\ Z \end{bmatrix}$$

Pinhole camera (or Projector) model



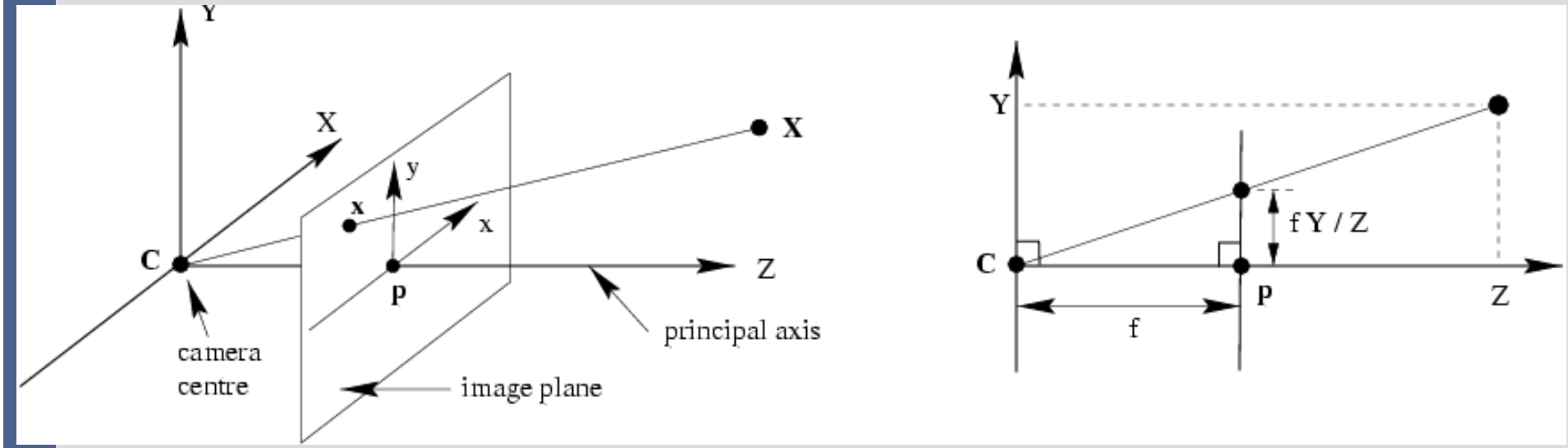
$$y = f\frac{Y}{Z}$$

$$\begin{bmatrix} fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix}$$

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$P = \mathrm{diag}(f, f, 1)[I \mid 0]$$

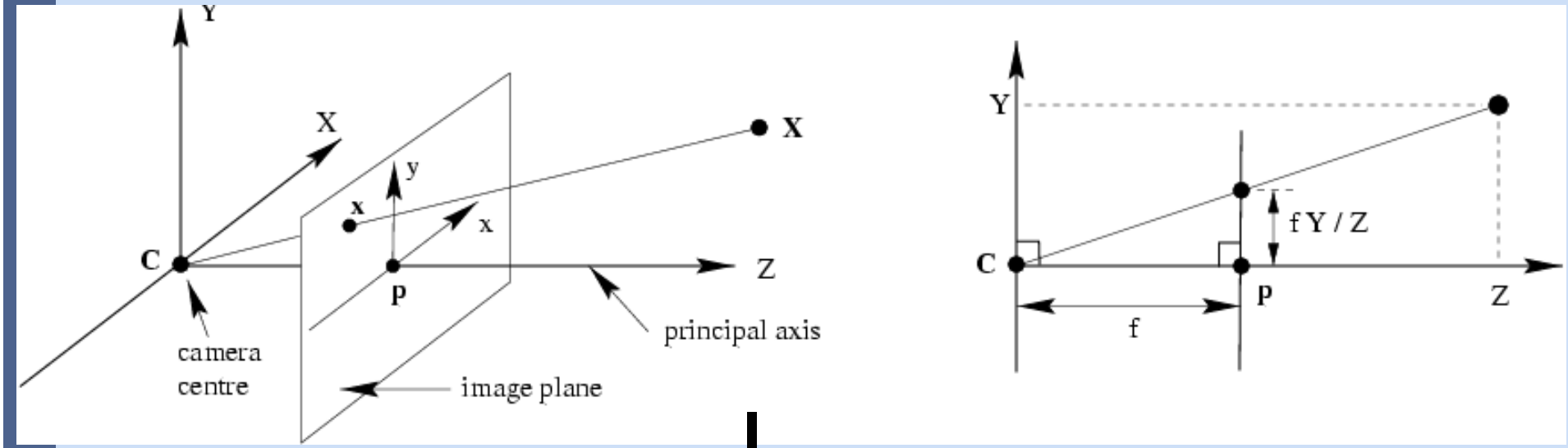
Pinhole camera (or Projector) model



$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\mathrm{x} \cong \mathrm{PX}$$

Pinhole camera (or Projector) model



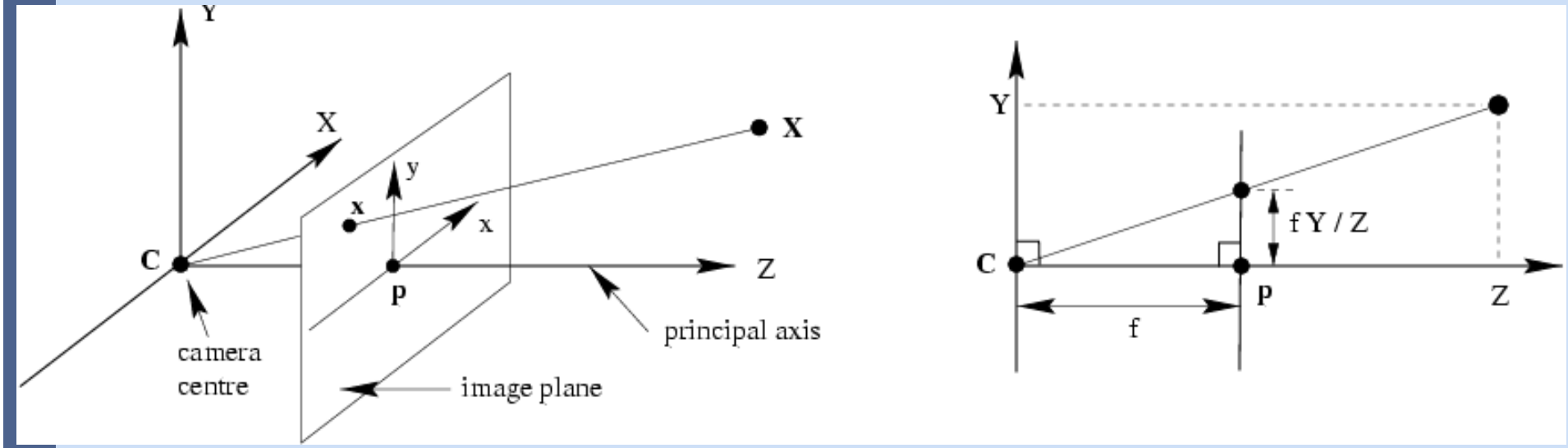
$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Vision

$$\begin{pmatrix} wx \\ wy \\ wz \\ w \end{pmatrix} = \begin{bmatrix} f & \cdot & \cdot & \cdot \\ \cdot & f & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 \\ \cdot & \cdot & 1 & \cdot \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Graphics

Pinhole camera (or Projector) model



$$\begin{pmatrix} wx \\ wy \\ wz \\ w \end{pmatrix} = \begin{bmatrix} f & \cdot & \cdot & \cdot \\ \cdot & f & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 \\ \cdot & \cdot & 1 & \cdot \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Mapping 3D to 2D

Vision

$$x = PX$$

$$P = \begin{bmatrix} K & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

Internal Matrix =

External Matrix =

Graphics

$$x = PX$$

$$P = \text{ViewPort} * \text{ProjectionMatrix} * \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

[ViewportMatrix * ProjectionMatrix]

ModelviewMatrix

Comments ? Questions ?

Projector parameter estimation method

- Calibration assistance tools
  - Calibration rigs and objects
  - Printed patterns
  - Blank walls and planes !

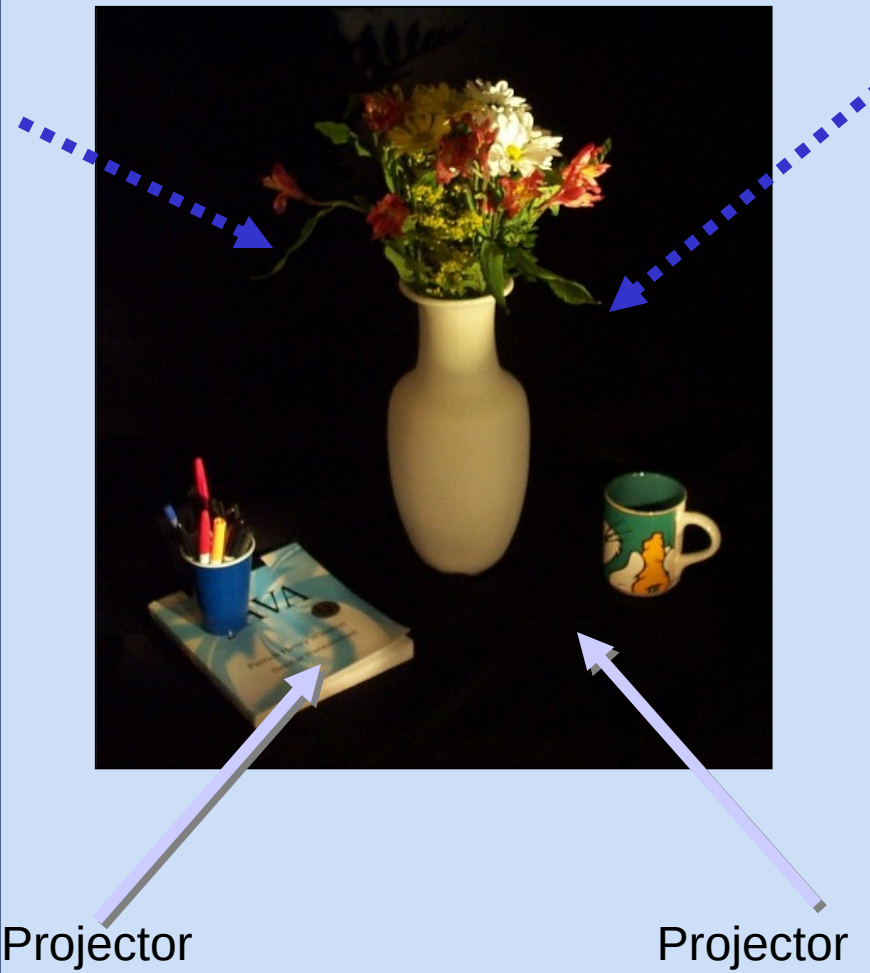
Steps

- Preprocessing
  - Scan 3D object and create model
  - Approximately position projector(s)
  - Compute pose, ***P***
    - Find fiducials
    - Find pixels that illuminate them
  - Compute intensity correction
- Run time
  - Render images of 3D model using matrix ***P***
  - Intensity correction for object shape
  - Feathering for projector overlap

Projector-based AR
Outline

- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

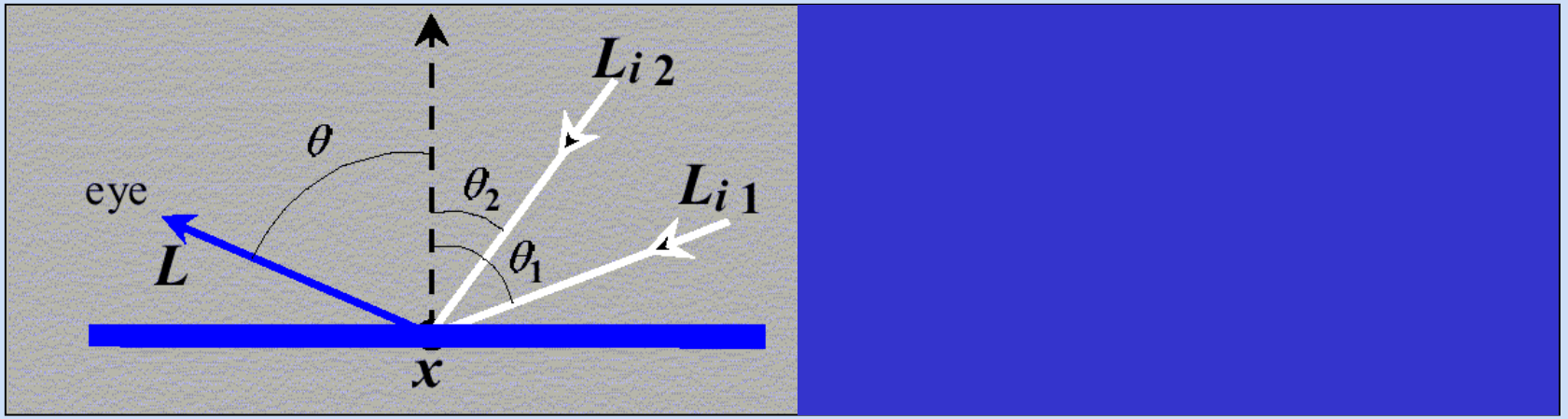
Changing Appearance



Steps

- Preprocessing
  - Scan 3D object and create model
  - Approximately position projector(s)
  - Compute pose, ***P***
    - Find fiducials
    - Find pixels that illuminate them
  - Compute intensity correction
- Run time
  - Render images of 3D model using matrix ***P***
  - Intensity correction for object shape
  - Feathering for projector overlap

Radiance Adjustment



Virtual

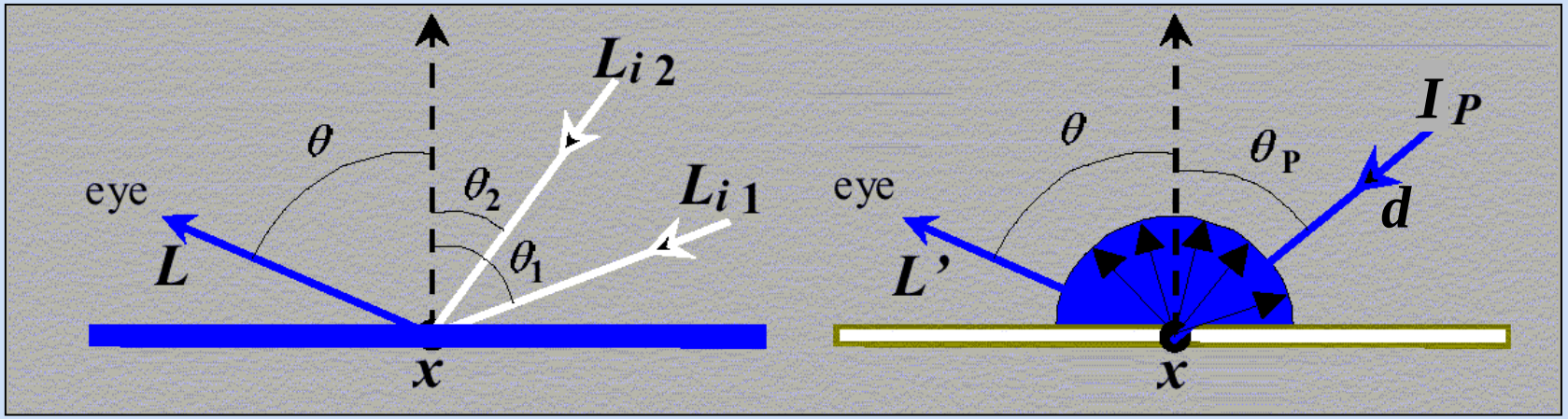
$$L(x,\theta) = \int F(x,\ \theta,\ \theta_i)\, L_i\,(x,\ \theta_i)\, d\omega_i$$

Desired radiance

BRDF

Incident radiance

Radiance Adjustment



Virtual

Real

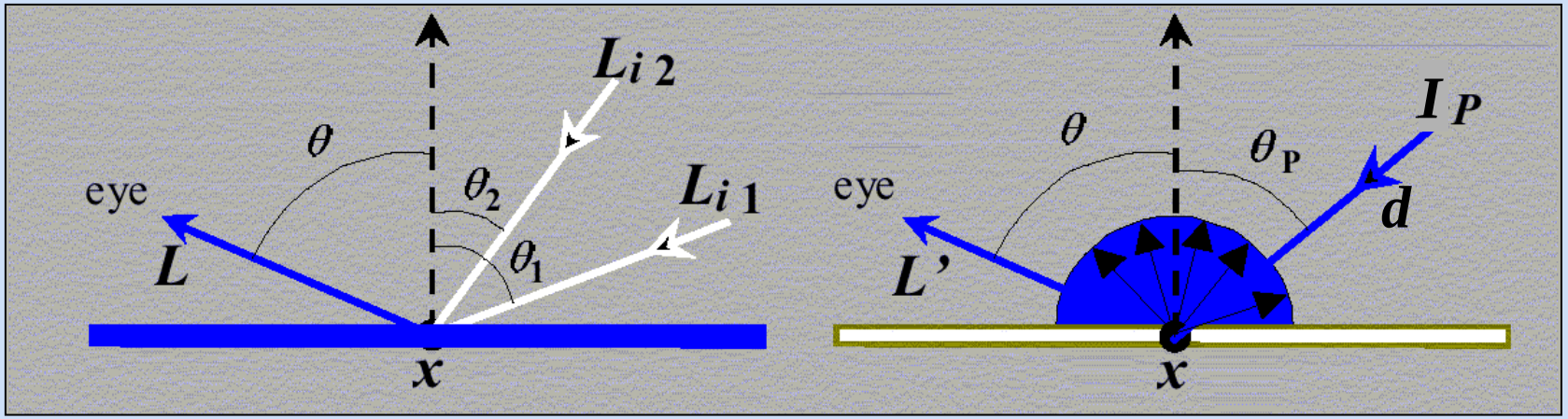
$$L(x,\theta) = \int F(x,\ \theta,\ \theta_i)\ L_i\ (x,\ \theta_i)\ d\omega_i$$

$$L'(x,\theta) = \frac{k(x)\ \cos(\theta_p)}{d(x)^2}\ I_p\ (x,\ \theta_p)$$

Resultant radiance

Pixel intensity

Radiance Adjustment



Virtual Real

$$I_p(x, \theta_p) = \frac{d(x)^2}{k(x)\ \cos(\theta_p)} L(x, \theta) \quad , \quad k(x) > 0$$

Pixel intensity

Intensity correction

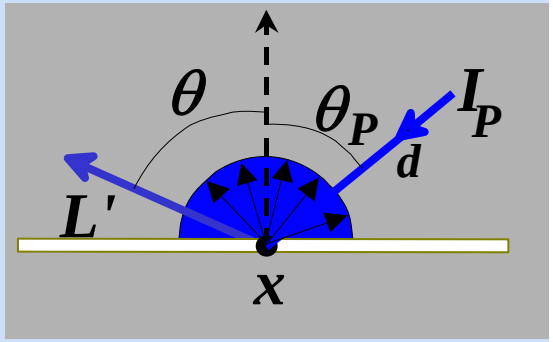
Desired radiance

Reflectance

Intensity Correction

Per-pixel factor Rendered Image

$$I_p(x, \theta_p) = \frac{d(x)^2}{k(x)\ \cos(\theta_p)} L(x, \theta)$$

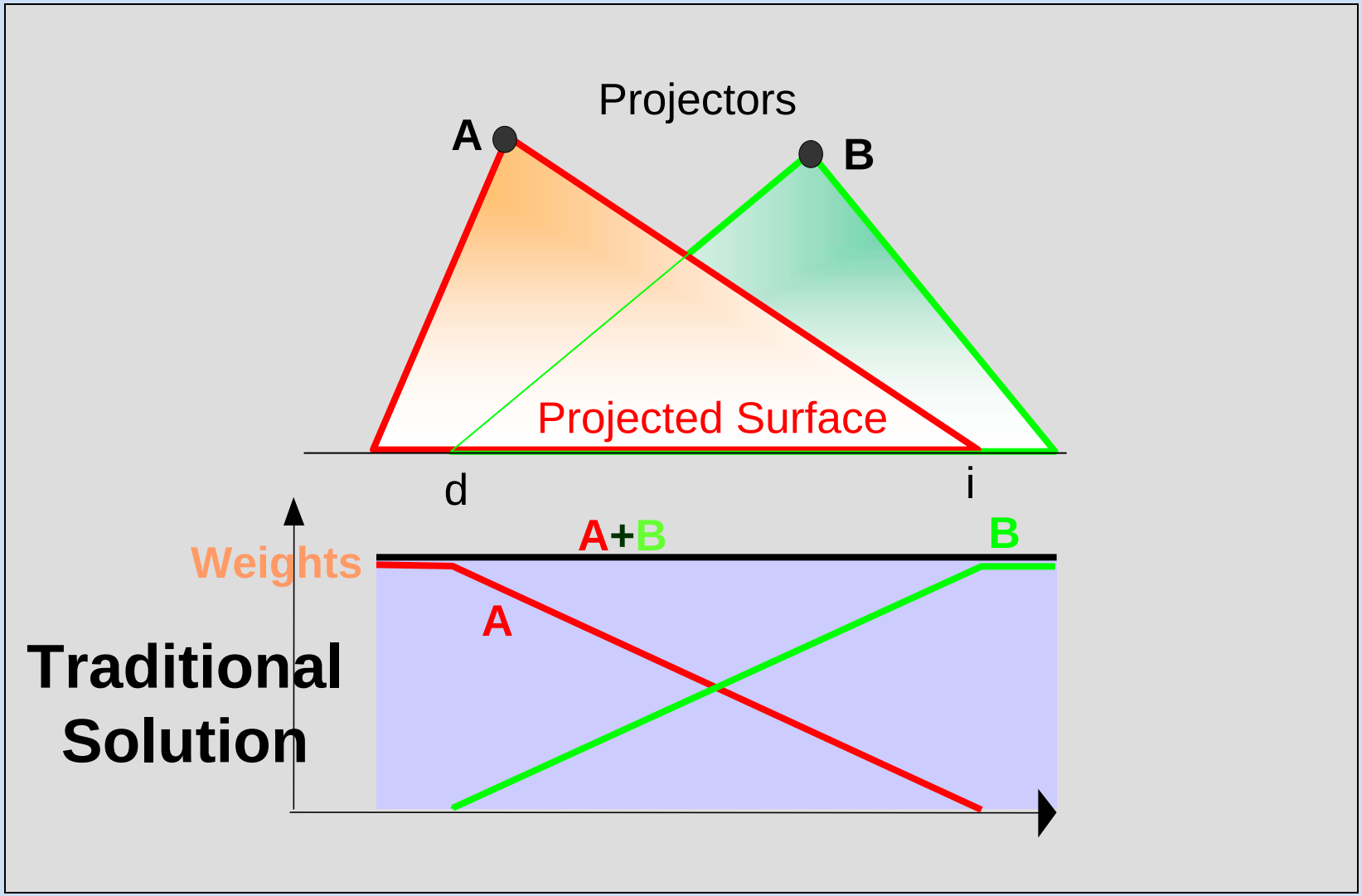


- Rendering with
  - Light at c.o.p. : $\cos(\theta_p)$
  - Diffuse reflectance : k
  - Distance attenuation : $1/\ d\ (x)^2$
  - θ_p > 60^0 cut off

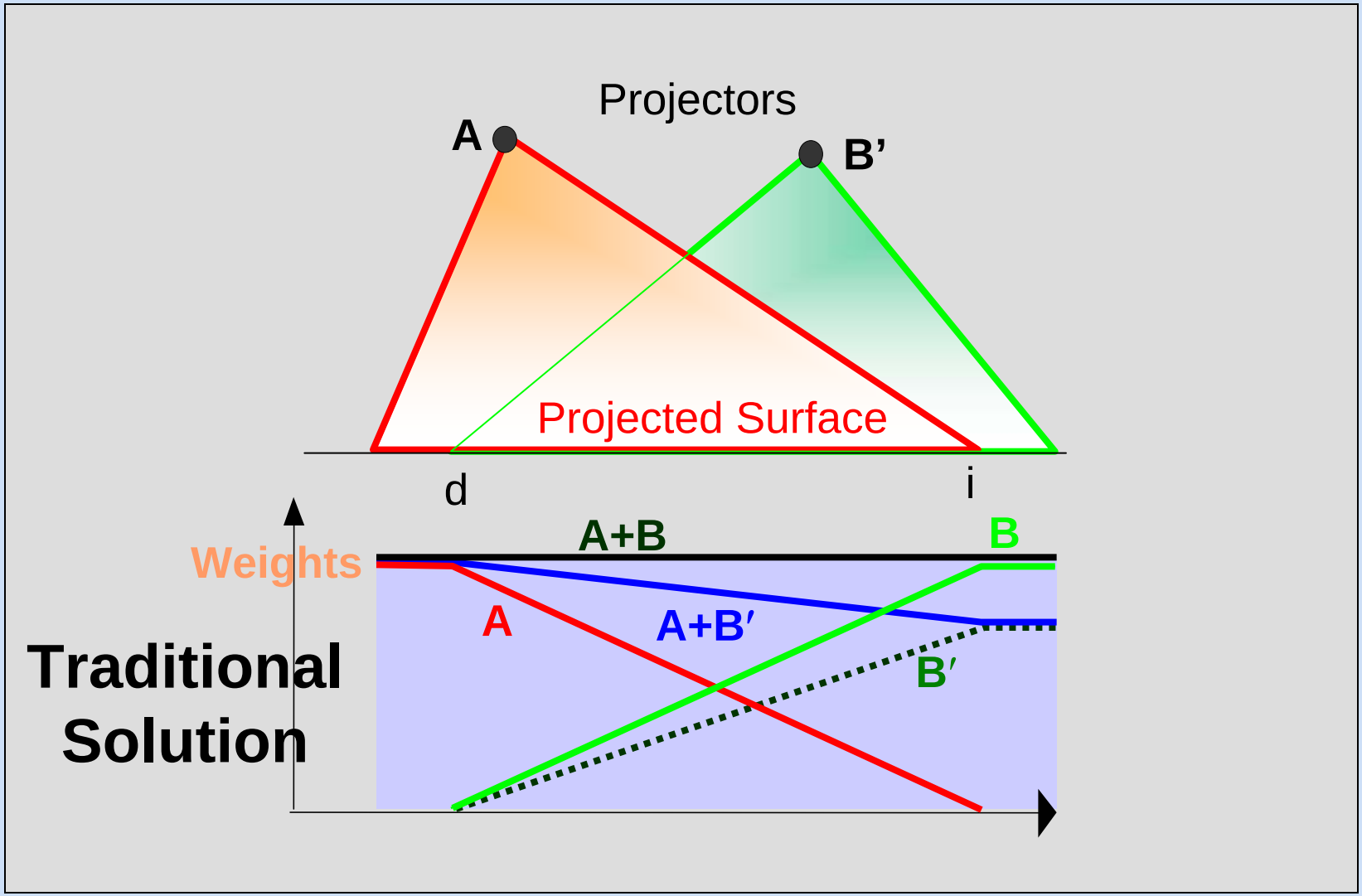
Projector-based AR
Outline

- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

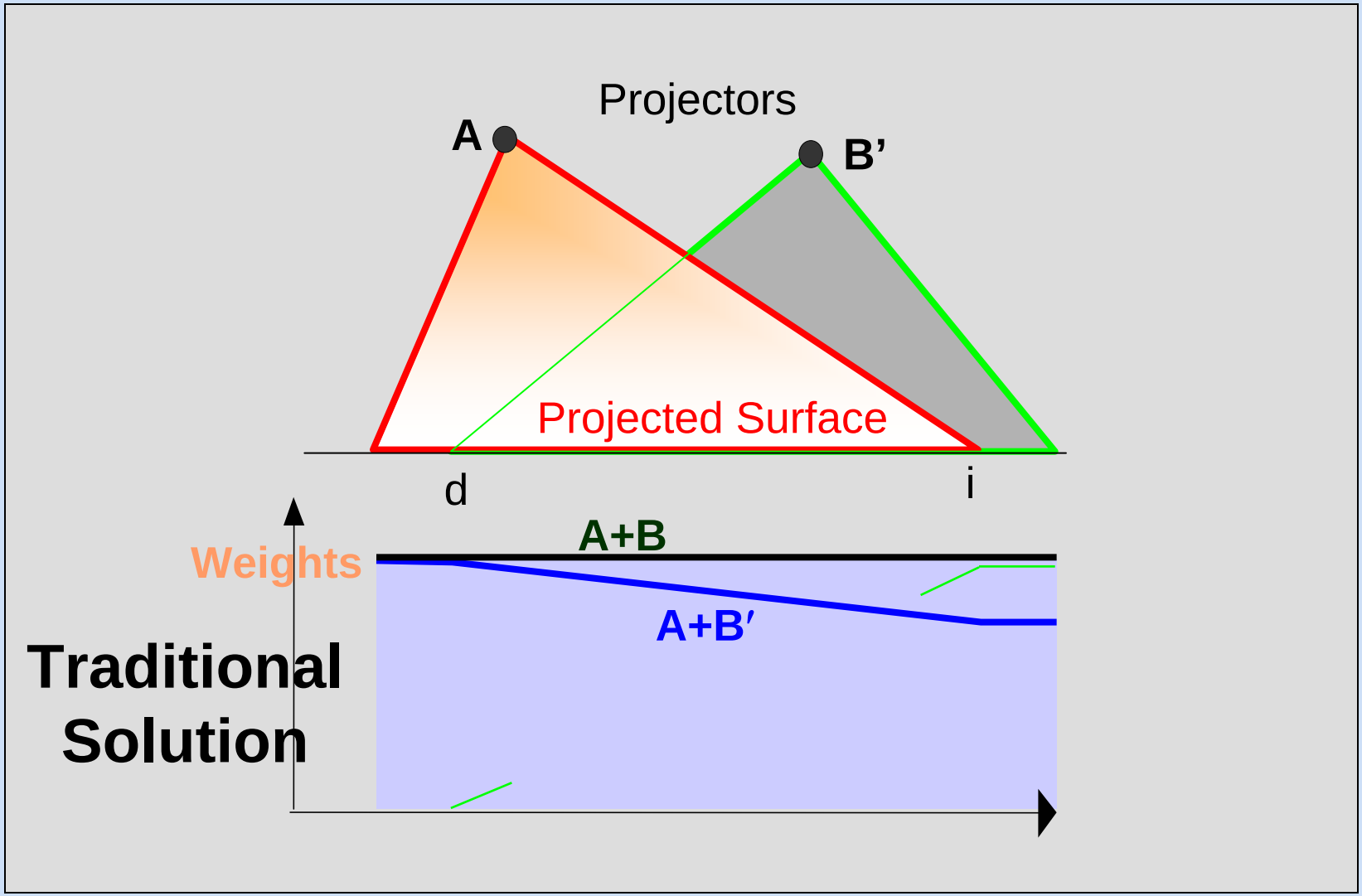
Feathering in Overlap



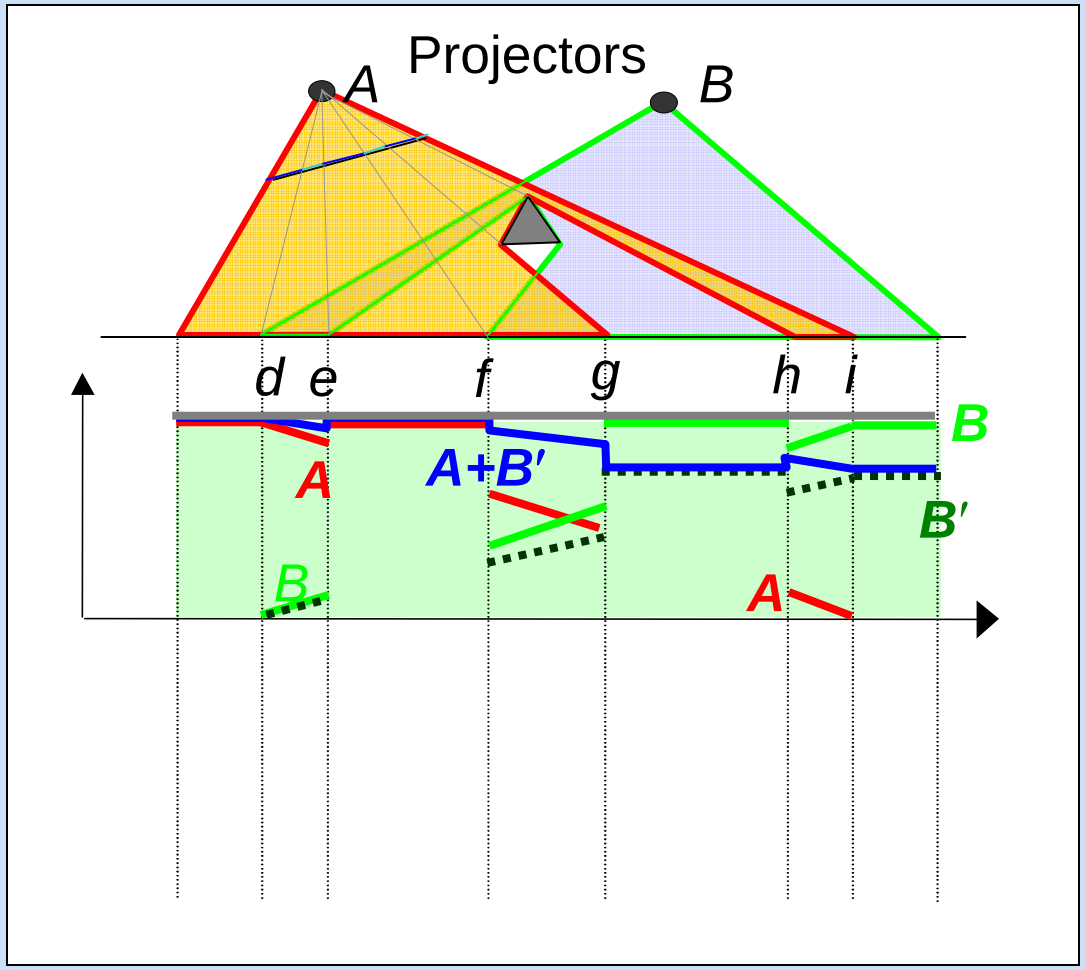
Feathering


Projectors
A
B'
Projected Surface
d
i
Weights
A+B
B
A
A+B′
B′
Traditional Solution

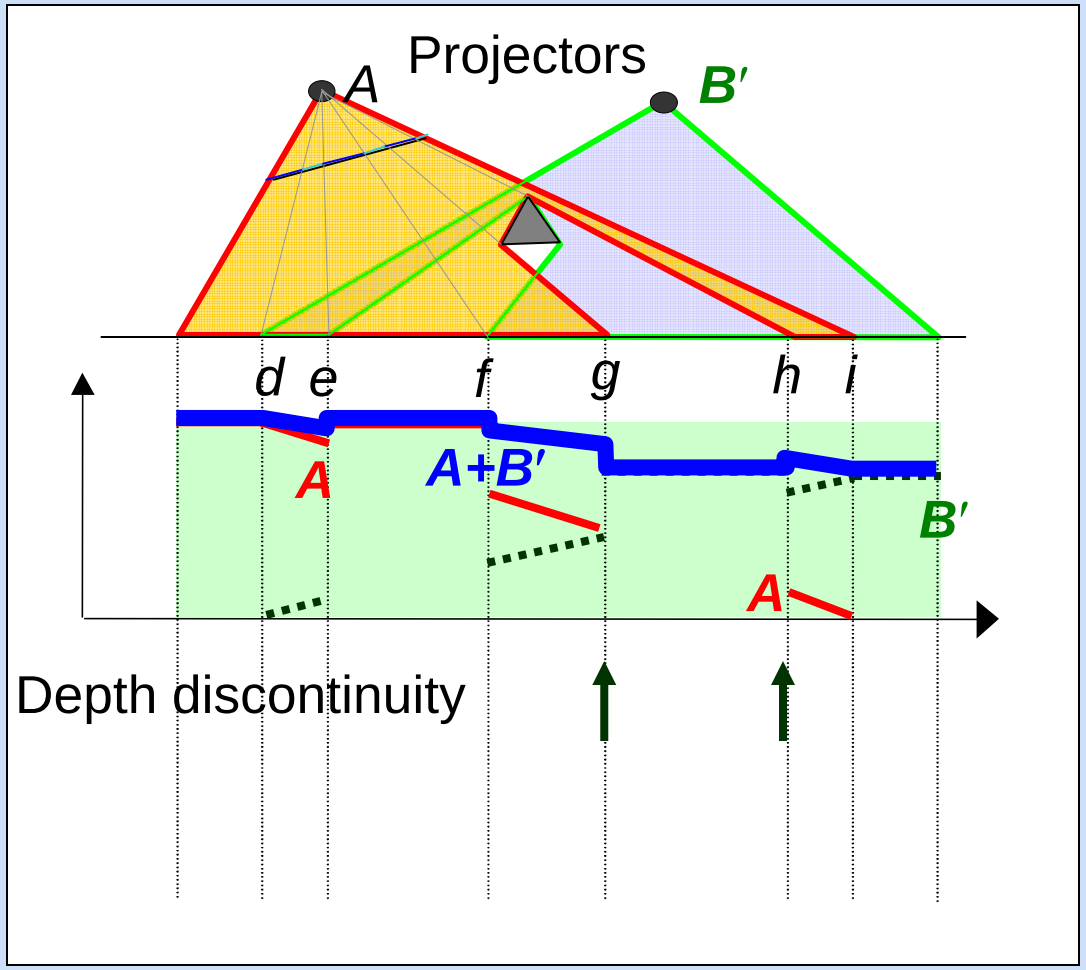
Feathering


Projectors
A
B’
Projected Surface
d
i
Weights
A+B
A+B′
Traditional Solution

Occlusions



Occlusion Problems



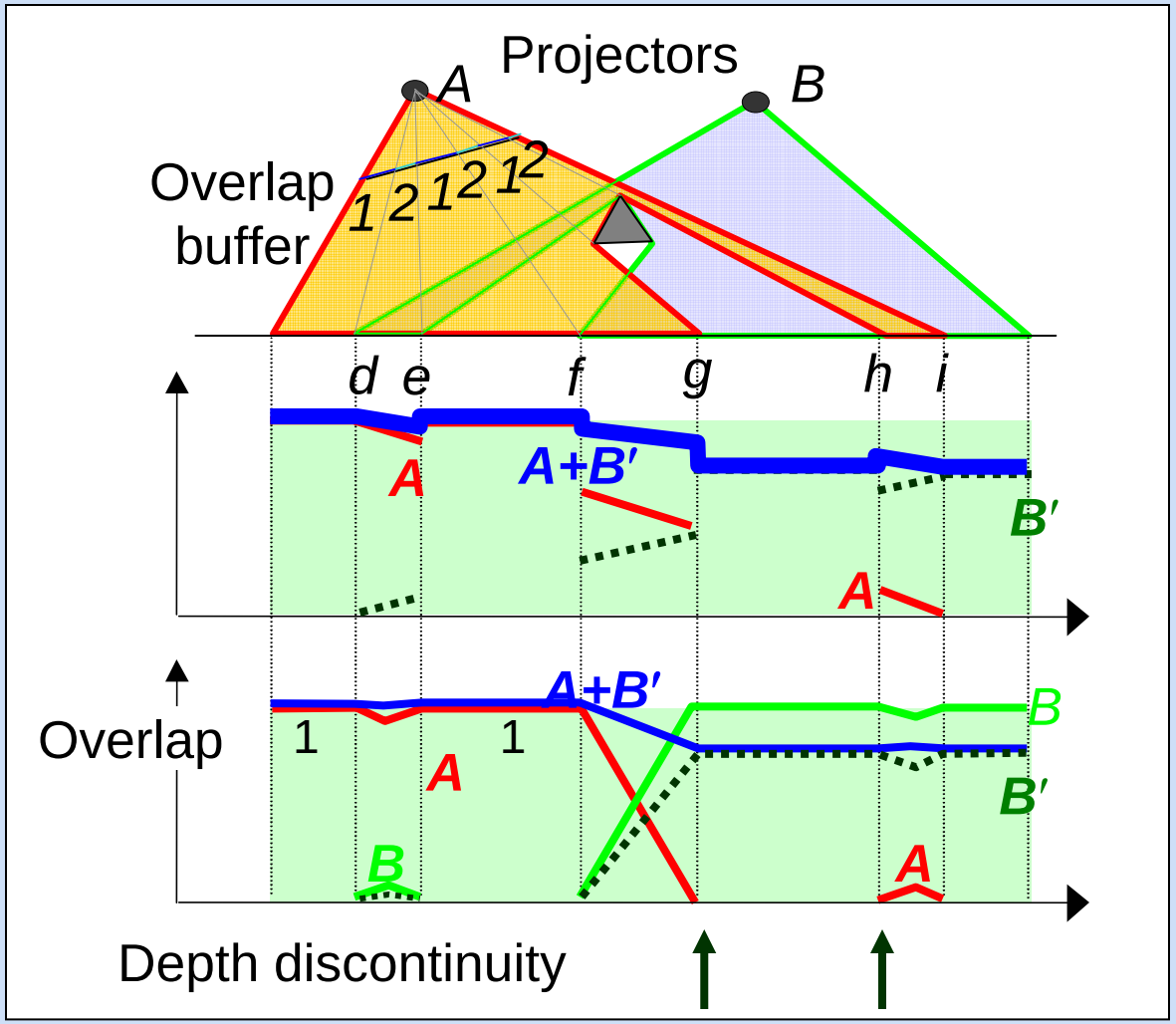
Occlusions and Shadows

- Dealing with depth discontinuity
- Goal
  - Sum of weights = 1
  - Weights along surface are smooth
  - Weights in image are smooth

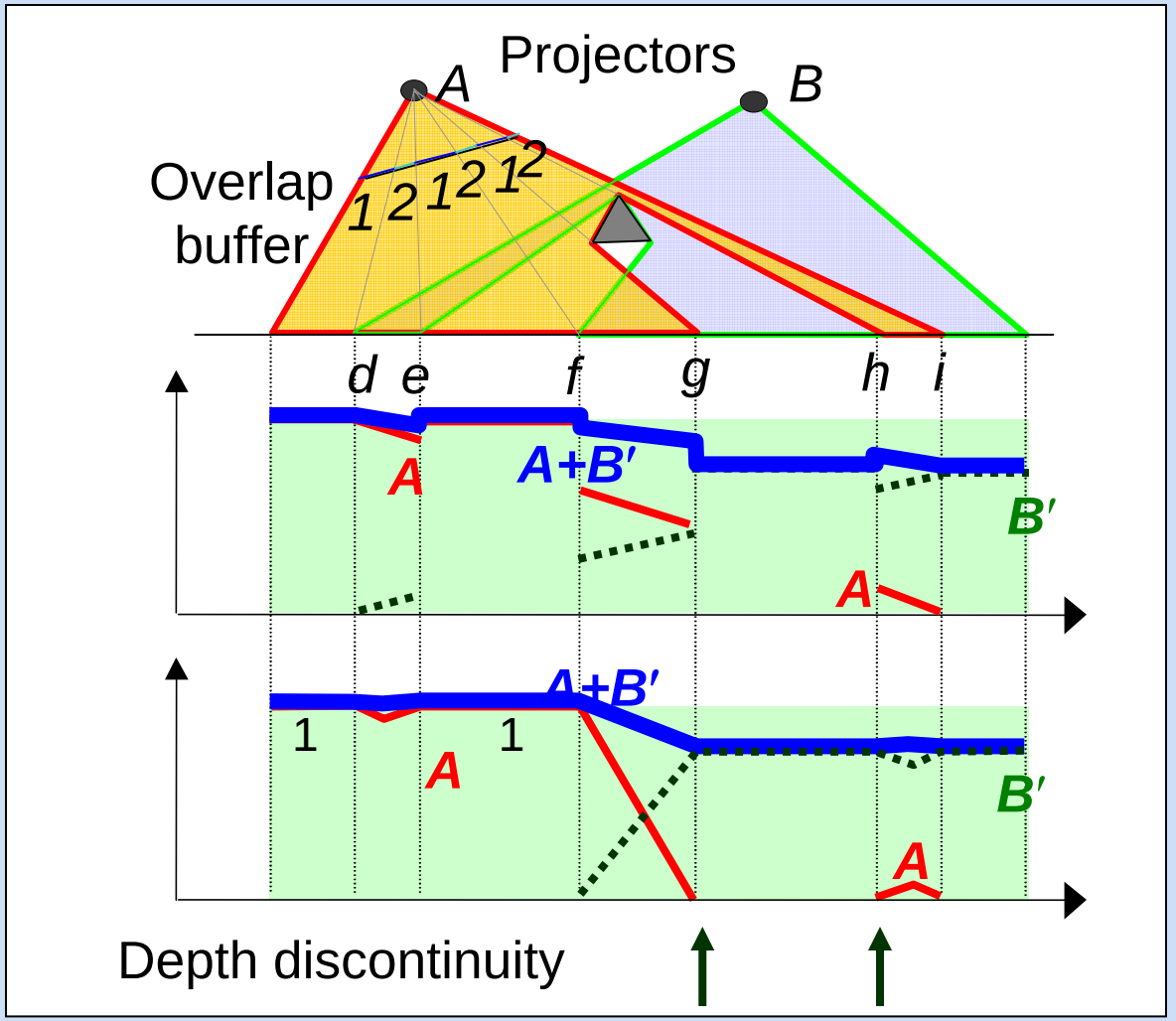
Feathering Algorithm

- Each Projector Image Space
  - Compute overlap count (0,1,2..)
  - Find depth discontinuity
  - For overlap region
    - Compute shortest distance to overlap = 1
    - Ignore paths crossing discontinuity
  - Weights = 1/(shortest distance)
- Normalize weights for corresponding pixels

New Feathering

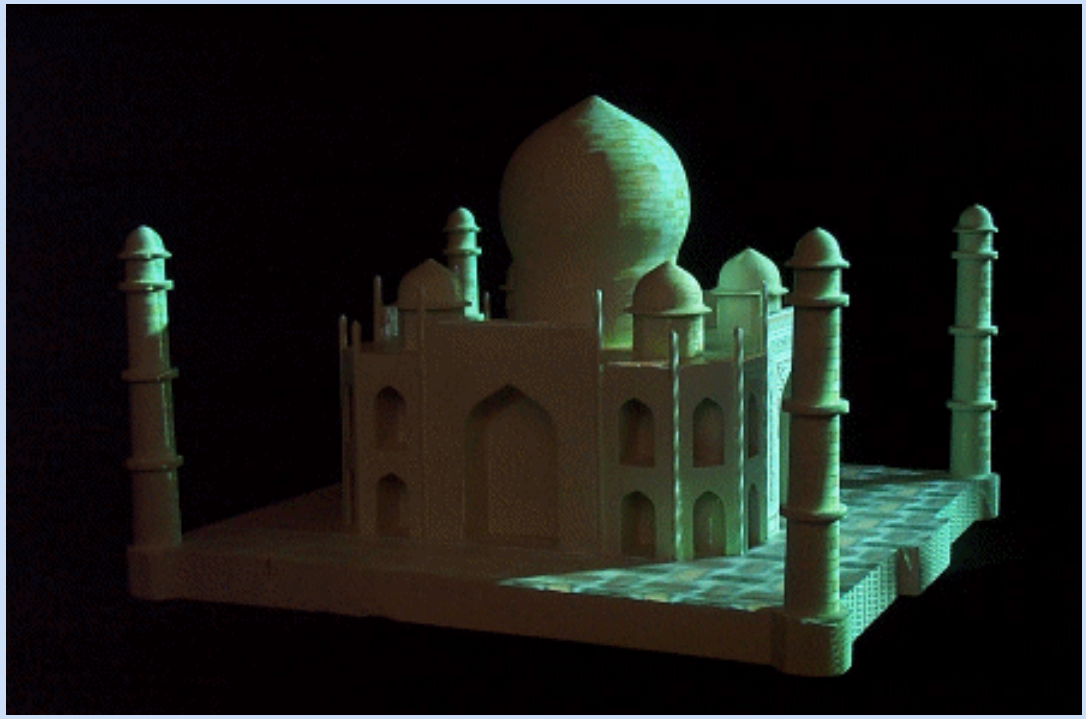


New Feathering



Virtual Illumination

Shadows, Shading and Blending



Steps

- Preprocessing
  - Scan 3D object and create model, ***G***
  - Approximately position projector(s)
  - Compute pose, ***P***
  - Compute intensity correction, α
- Run time
  - Render image [***I***]
    using model ***G*** from pose ***P***

eps

- Compute intensi

– Run time

- Render image [***I***]
- Apply intensity correction for object shape [α]*[***I***]
- Apply feathering for projector overlap, [β]*[α] *[***I***]

eps

- Compute pose
- Compute intens

– Run time

- Render image [***I***]
- Intensity correction, [α] *
- Feathering, [β] * [α] * [

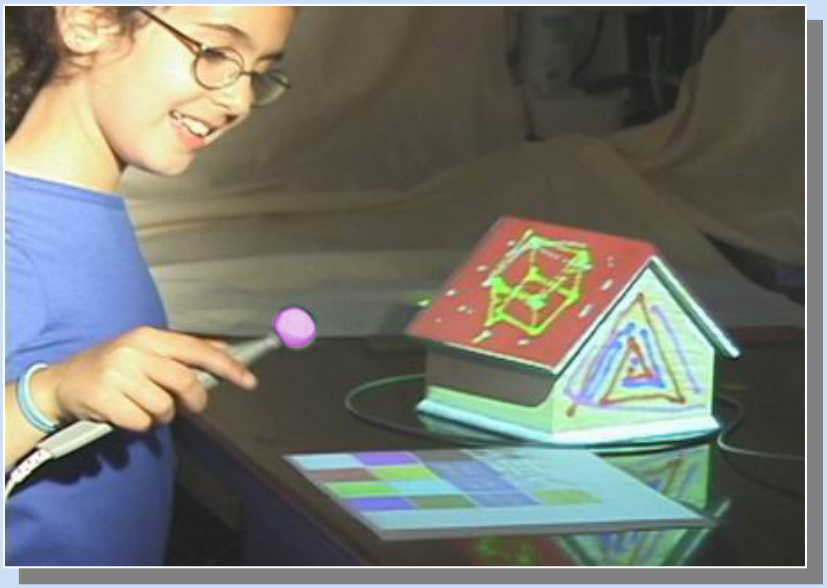
Steps

- Preprocessing
  - Scan 3D object and create model, ***G***
  - Approximately position projector(s)
  - Compute pose, ***P***
    - Find fiducials
    - Find pixels that illuminate them
    - Find projector pose
- Run time
  - Render 3D model ***G*** from ***P*** , [***I***]
  - Intensity correction for object shape [α] * [***I***]
  - Feathering for projector overlap [β] * [α] * [***I***]

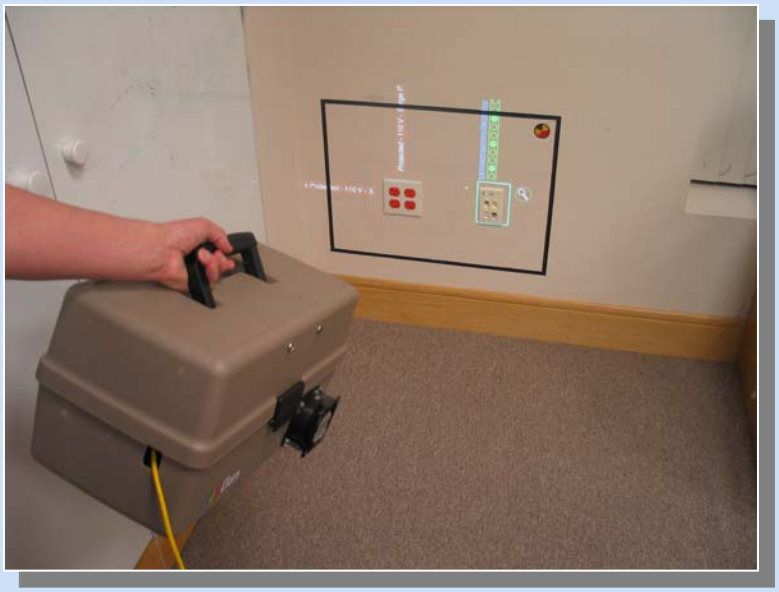
Projector-based AR
Outline

- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

Moving Objects



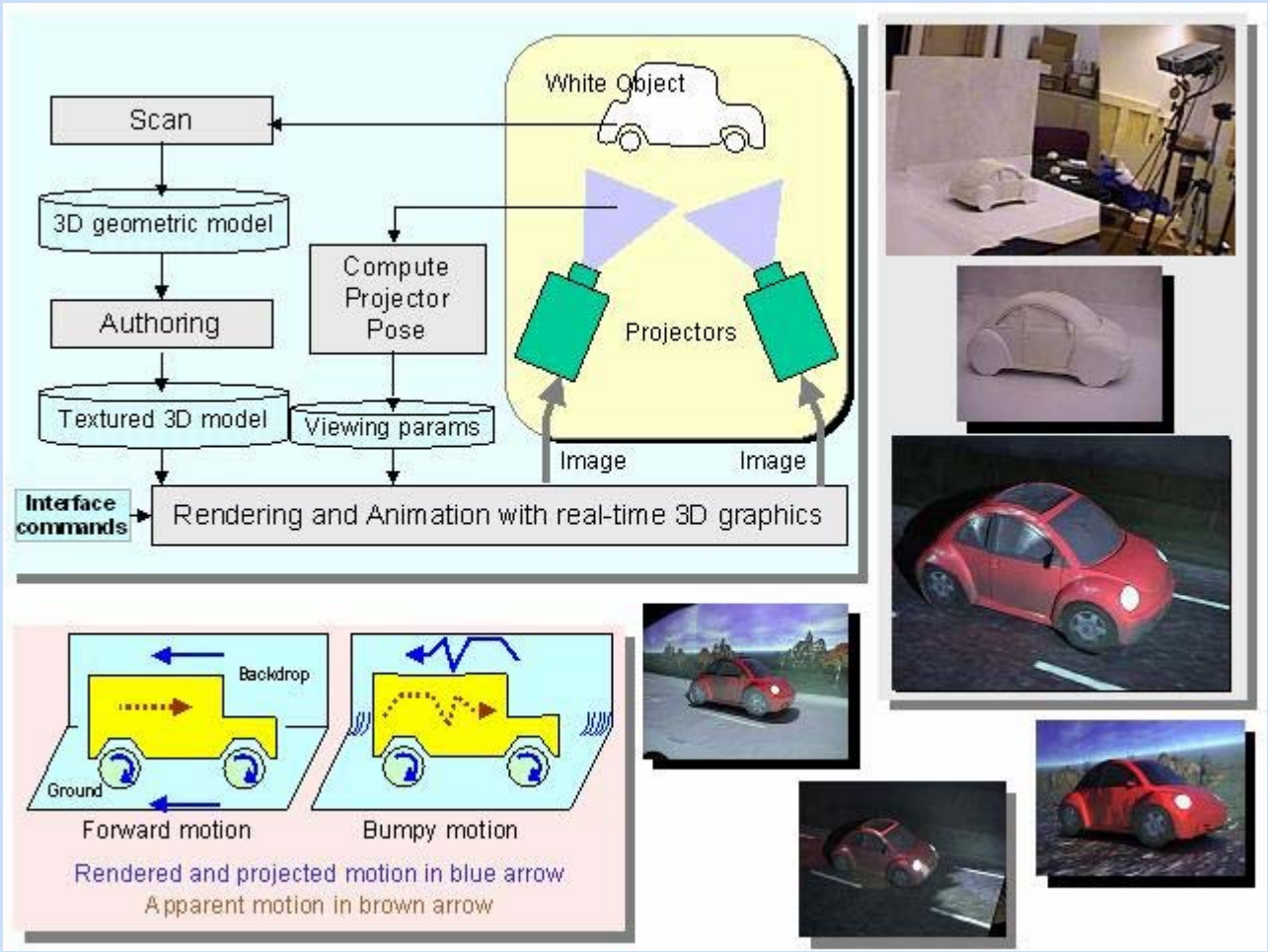
Moving Surface



Moving Projector

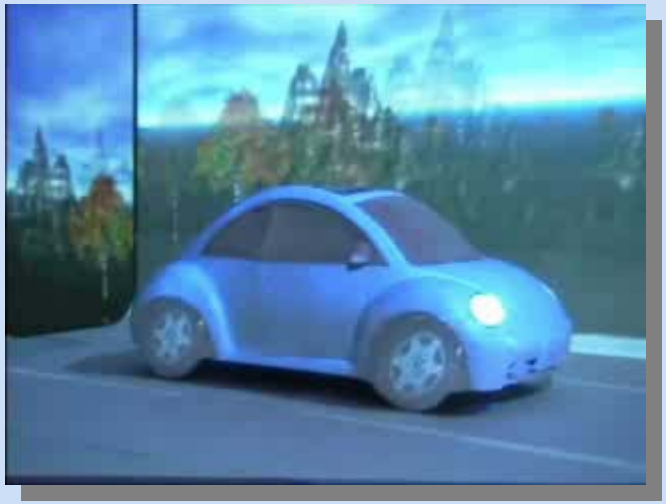
Moving Viewer

Apparent Motion



Ramesh Raskar, Remo Ziegler, Thomas Willwacher, “Cartoon Dioramas in Motion,” Proc. ACM Symposium on Nonphotorealistic Animation and Rendering (NPAR 2002)

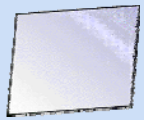
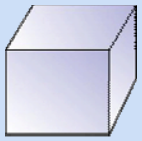
Virtual Motion



Projector-based AR
Outline

- Concepts and Hardware Prototypes
  - Spatially Augmented Reality
  - Shader Lamps
- Rendering Techniques for non-trivial projection
  - Calibration
  - Changing Surface Appearance
  - Merging Overlapping Projection
  - Moving Objects
  - Shape Adaptive Projection

Shape Adaptive Projection

- Projection Screen Geometries
  - Planar 
  - Rectilinear 
  - Cylindrical
  - Spherical
  - Irregular

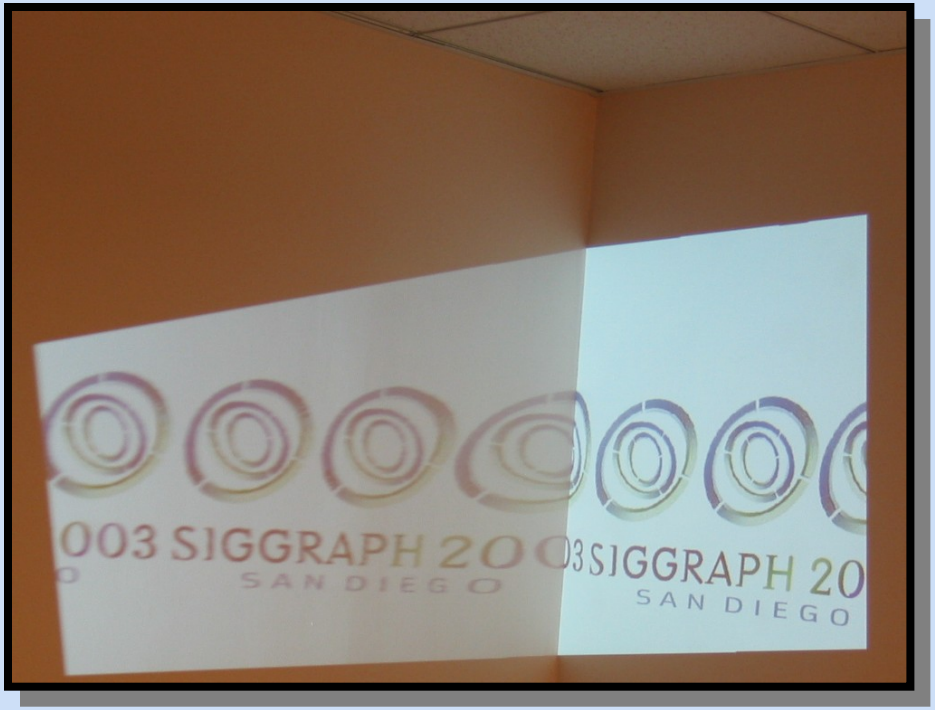
Planar Homography

Quadric image transfer

Discretized Warping

Shape Adaptive Projection

Minimum Stretch Display



‘Wallpaper the surface’

Shape cue allows perceptual unwrapping

Shape Adaptive Projection

Raskar, vanBaar, Beardsley, Willwacher, Rao, Forlines
‘iLamps: Geometrically Aware and Self-Configurable Projectors’, SIGGRAPH 2003

Shape Adaptive Projection

Problem : Given input image texture, pre-warp so that

(i) displayed image has minimum distortion

(ii) image vertical aligns with world vertical

Approach :

(i) conformal mapping

(ii) sense surface geometry, pose and vertical

1. Least square conformal mapping [Levy 2002]

Texture coords $u + iv$

Surface coords $x + iy$

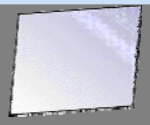
Orthogonal iso-u and iso-v contours

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$$

2. Align input texture vertical with world vertical

3. Render textured surface model from projector's pose

Projection Techniques

- Projection Screen Geometries
  - Planar 
  - Rectilinear

  Planar Homography

  - Cylindrical
  - Spherical

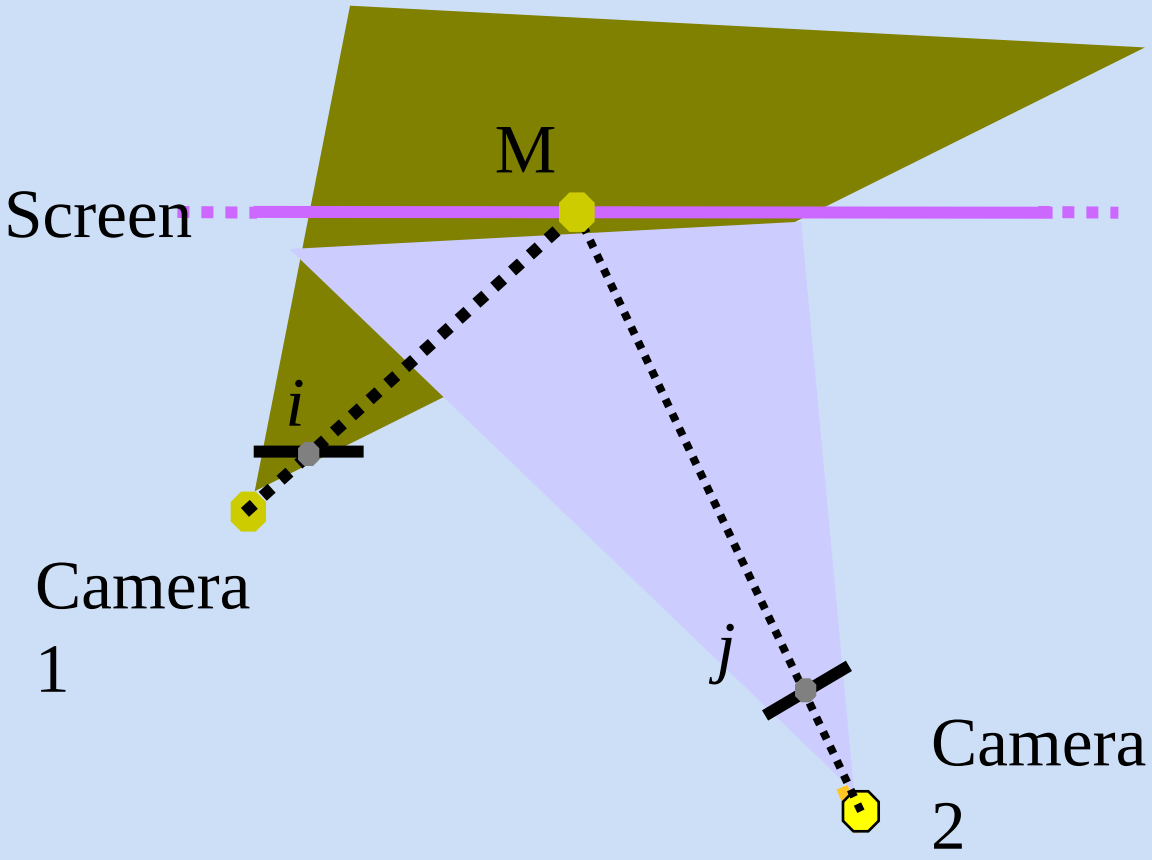
  Quadric image transfer

  - Irregular

  Discretized Warping

Planar projective transfer
What is homography ?

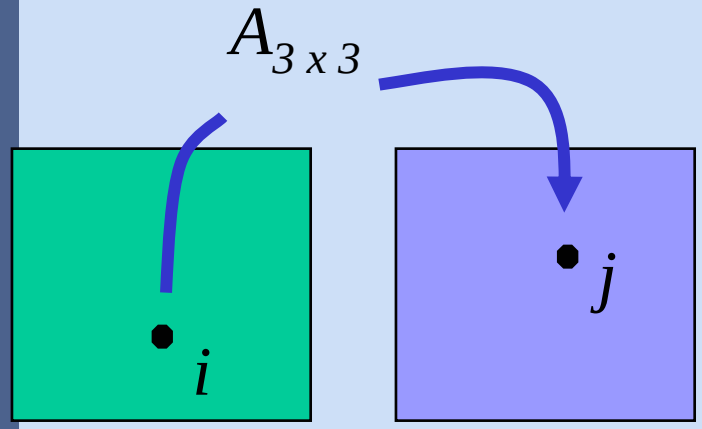
- Two images of 3D points on a plane are related by a 3x3 matrix



Planar Homography (in 2D)

Two images of 3D points on a plane

Related by a 3x3 matrix $j \cong A_{3\,x\,3}\, i$



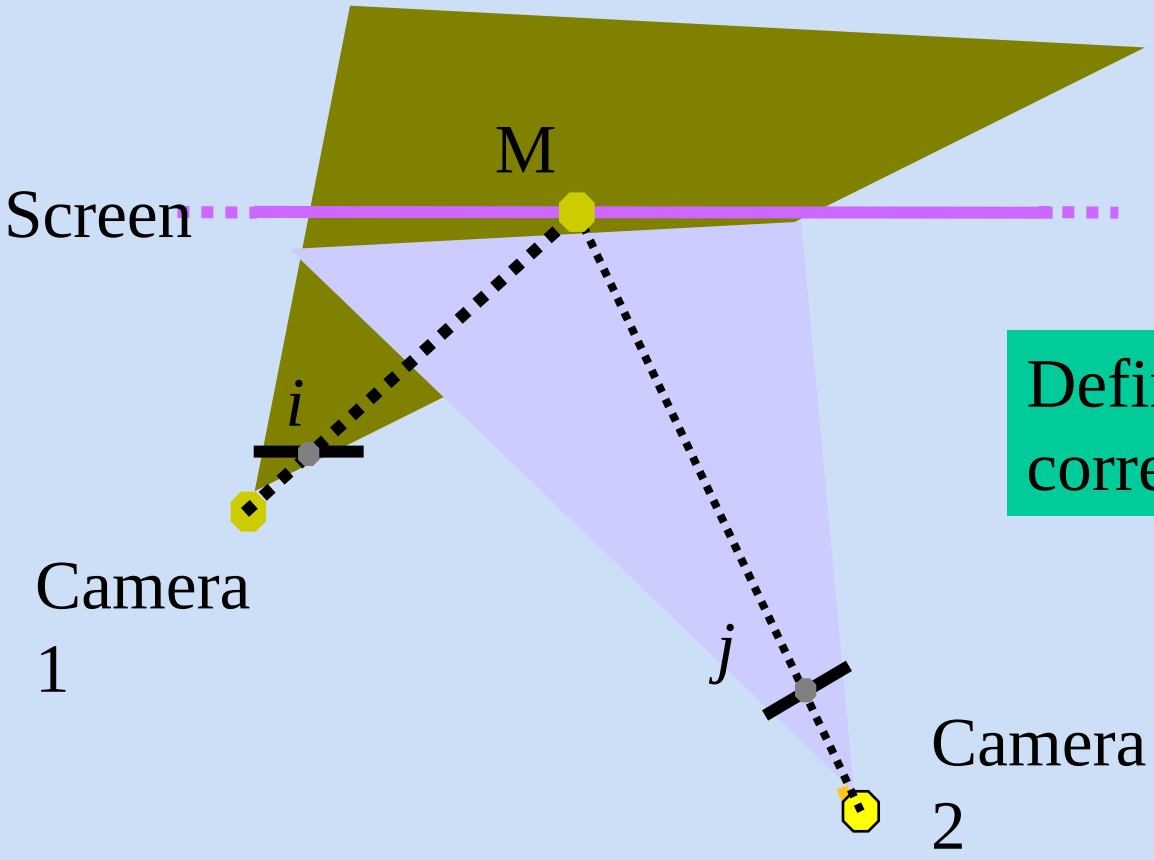
$$k \begin{bmatrix} j_x \\ j_y \\ 1 \end{bmatrix} = \begin{bmatrix} a1 & a2 & a3 \\ b1 & b2 & b3 \\ c1 & c2 & c3 \end{bmatrix} \begin{bmatrix} i_x \\ i_y \\ 1 \end{bmatrix}$$

$$j_x = (\boldsymbol{a} \cdot i) / (\boldsymbol{c} \cdot i)$$

$$j_y = (\boldsymbol{b} \cdot i) / (\boldsymbol{c} \cdot i)$$

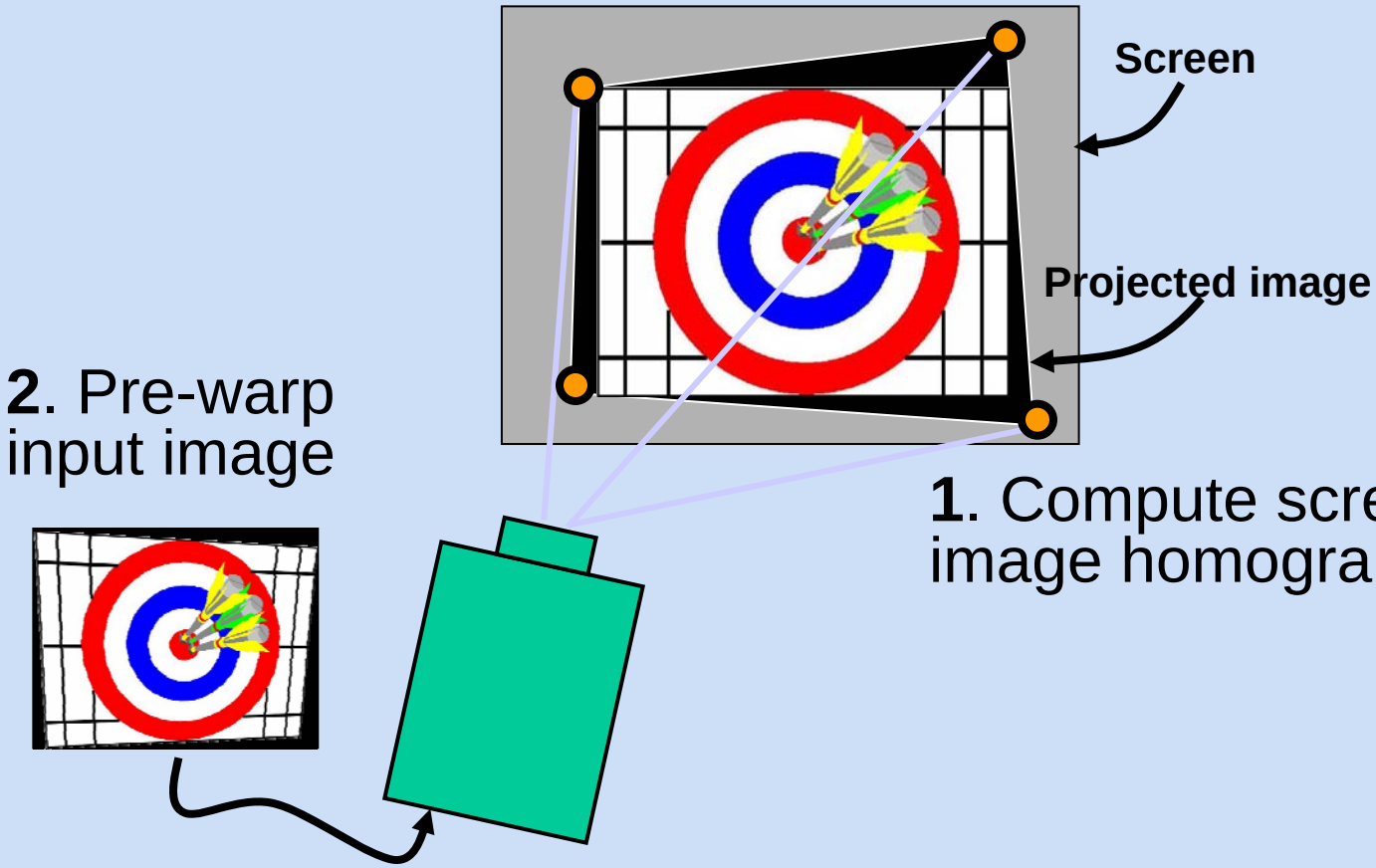
Planar projective transfer (homography)

- Two images of 3D points on a plane are related by a 3x3 matrix



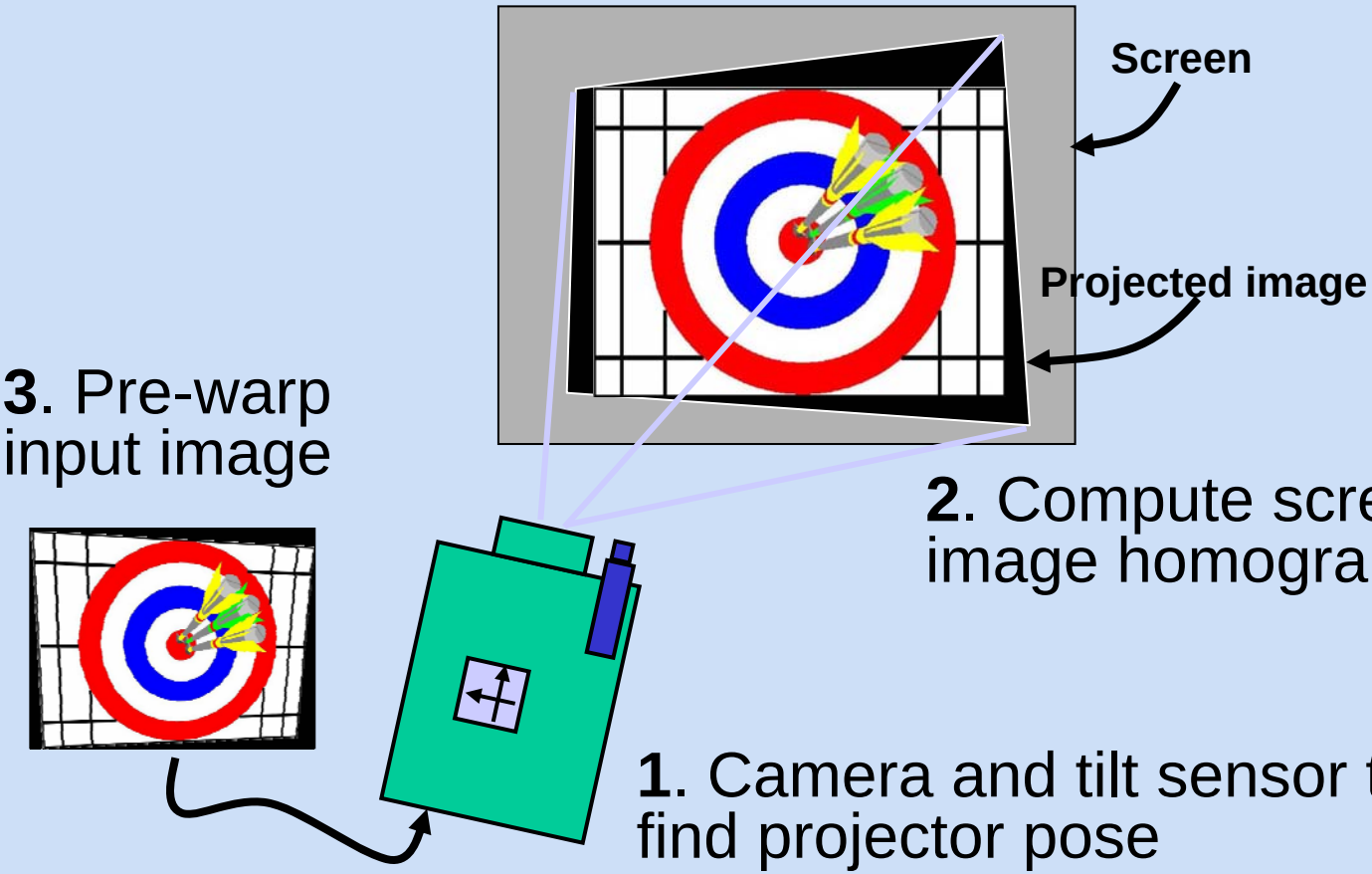
Defined by 4 or more corresponding pixels

Keystone Correction



Automatic Keystone Correction with Camera and Tilt Sensor

[Raskar and Beardsley01]

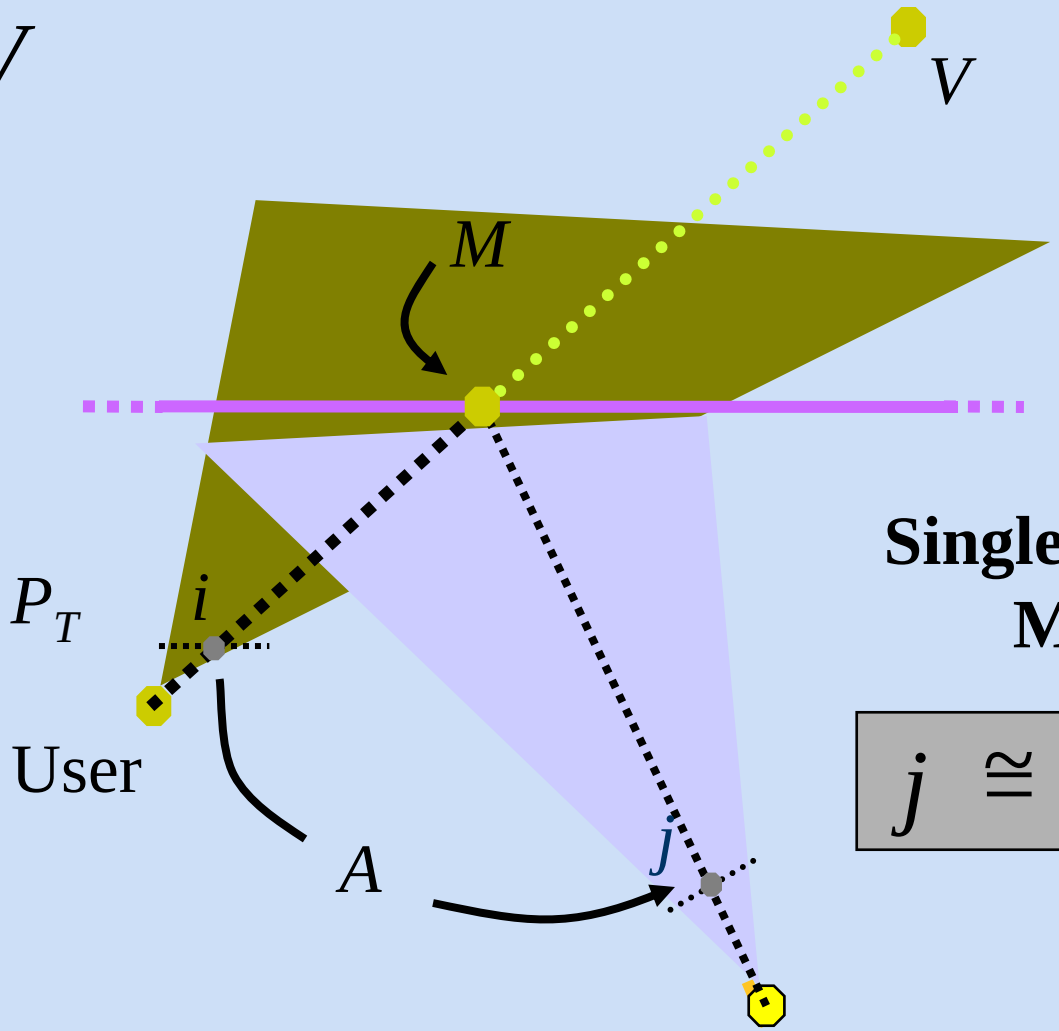


Planar display surface

Use homography (A_{3x3})

$$i \cong P_T V$$

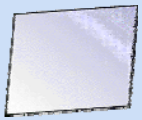
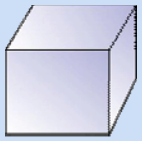
$$j \cong A\, i$$



Single Projection Matrix !

$$j \cong [A\, P_T]\, V$$

Projection Techniques

- Projection Screen Geometries
  - Planar 
  - Rectilinear 

Planar Homography

  - Cylindrical
  - Spherical

Quadric image transfer

  - Irregular

Discretized Warping

Quardic curved shape Displays

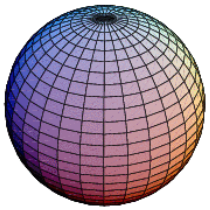
Planetarium

Sim/Viz Center

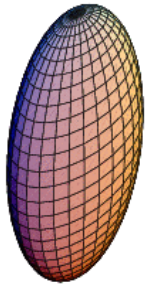
Raskar, vanBaar, Willwacher, Rao
‘Quadric Transfer for Immersive Curved Displays’,
EuroGraphics 2004

Curved projective transfer
Quadric classification

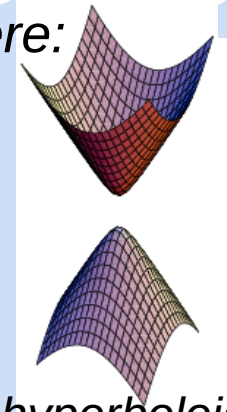
Projectively equivalent to *sphere:*



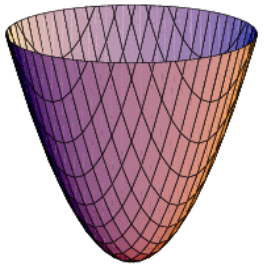
sphere



ellipsoid

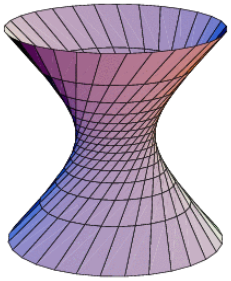


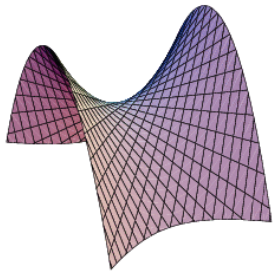
hyperboloid of two sheets



paraboloid

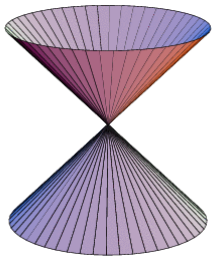
Ruled quadrics:



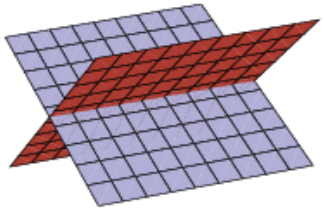


hyperboloids of one sheet

Degenerate ruled quadrics:



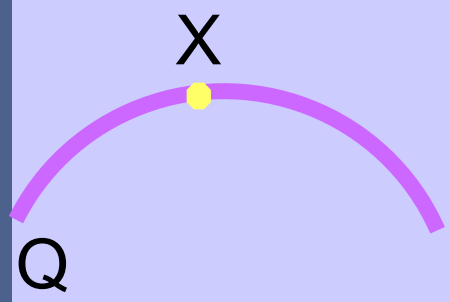
cone



two planes

Quadrics

$X^T Q X = 0$ For 3D points X on Quadric

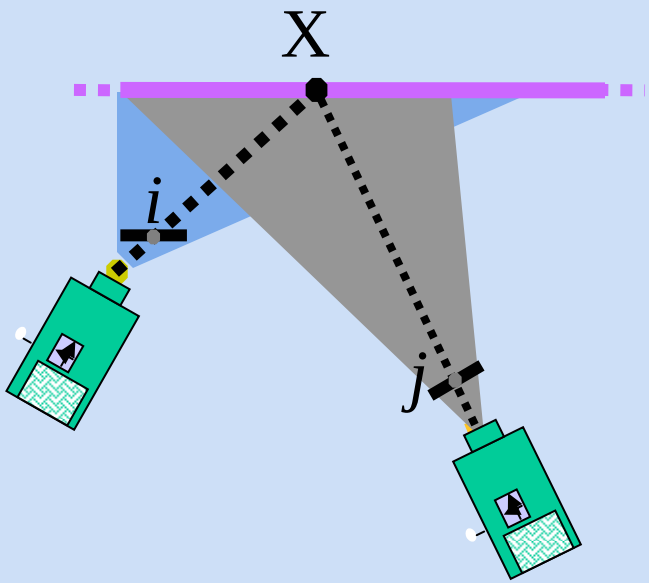


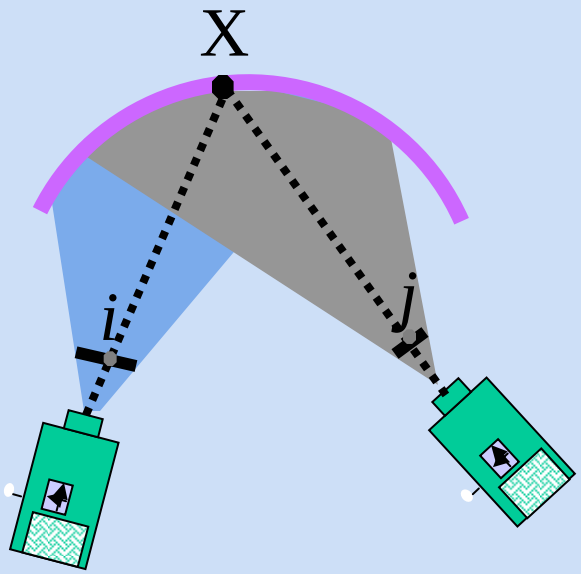
Q : 4x4 symmetric matrix, $Q = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet \\ \circ & \bullet & \bullet & \bullet \\ \circ & \circ & \bullet & \bullet \\ \circ & \circ & \circ & \bullet \end{bmatrix}$

Nine d.o.f

In general 9 points in 3D define quadric

Parametric Image Transfer





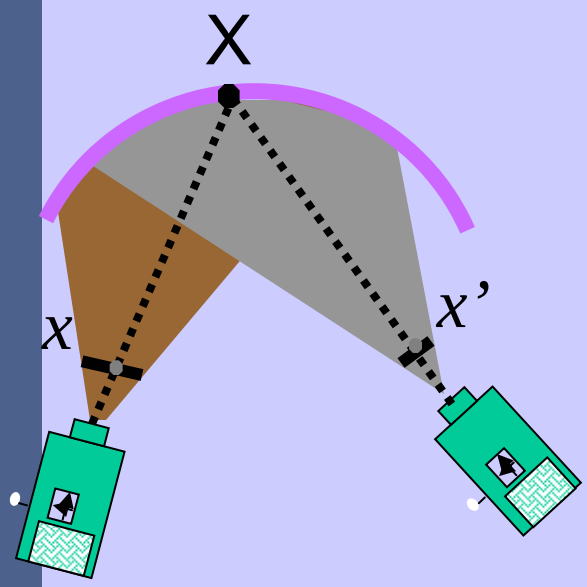
Planar Homography

$$j \cong A_{3x3}\, i$$

Quadric Transfer

$$j \cong A_{3x3}\, i \pm \left(\sqrt{i^T E\, i}\right) e$$

Simplified Quadric Image Transfer

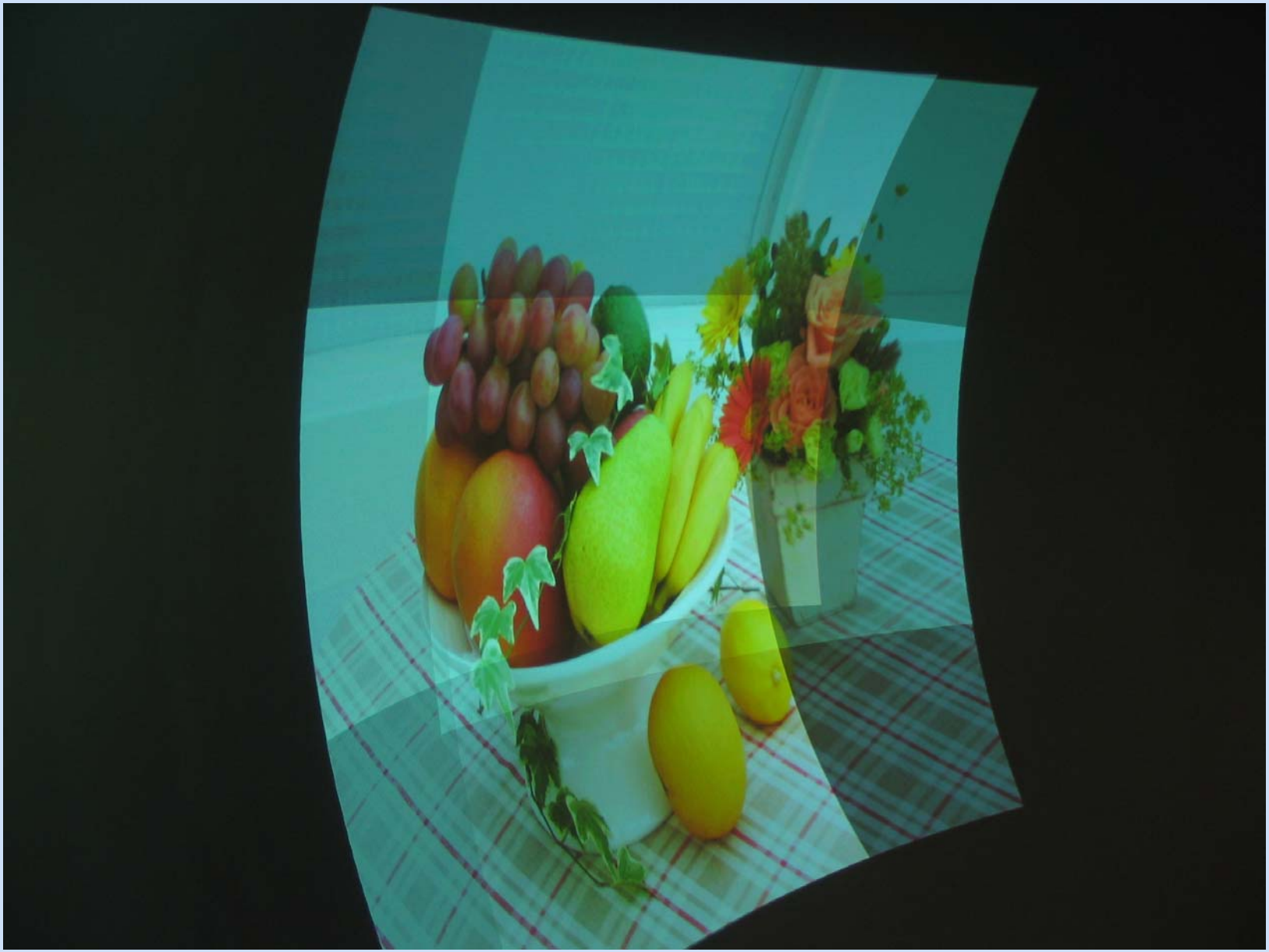


$$x' \cong Ax \pm \left(\sqrt{x^T Ex} \right) e$$

17 param warp

Planar homography:	4 corresponding pixels
Quadric transfer:	9 corresponding pixels

Overlap on Quadric Screens

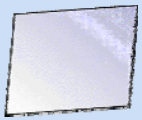
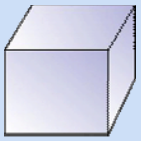


Vertex Shader for Quadric Transfer in Cg

```
vertout main( appin IN,  uniform float4x4 modelViewProj,
uniform float4 constColor, uniform float3x3 A,  uniform float3x3 E,
      uniform float3 e) {
    vertout OUT;
    float4 m1 = float4(IN.position.x, IN.position.y, IN.position.z,
1.0f );
    float4 m, mi ;   float3 m2,mp;   float scale;

    m = mul( modelViewProj, m1);
    m2.x = m.x/m.w;        m2.y = m.y/m.w;        m2.z = 1;
    scale = mul(m2, mul(E,m2));
    mp = mul(A,m2) + sqrt(scale)*e;
    mi.x = m.w * (mp.x)/(mp.z);
    mi.y = m.w * (mp.y)/(mp.z);
    mi.zw = m.zw;
    OUT.position = mi;
    OUT.color0 = IN.color0; // Use the original per-vertex color
specified
    return OUT;
}
```

Projection Techniques

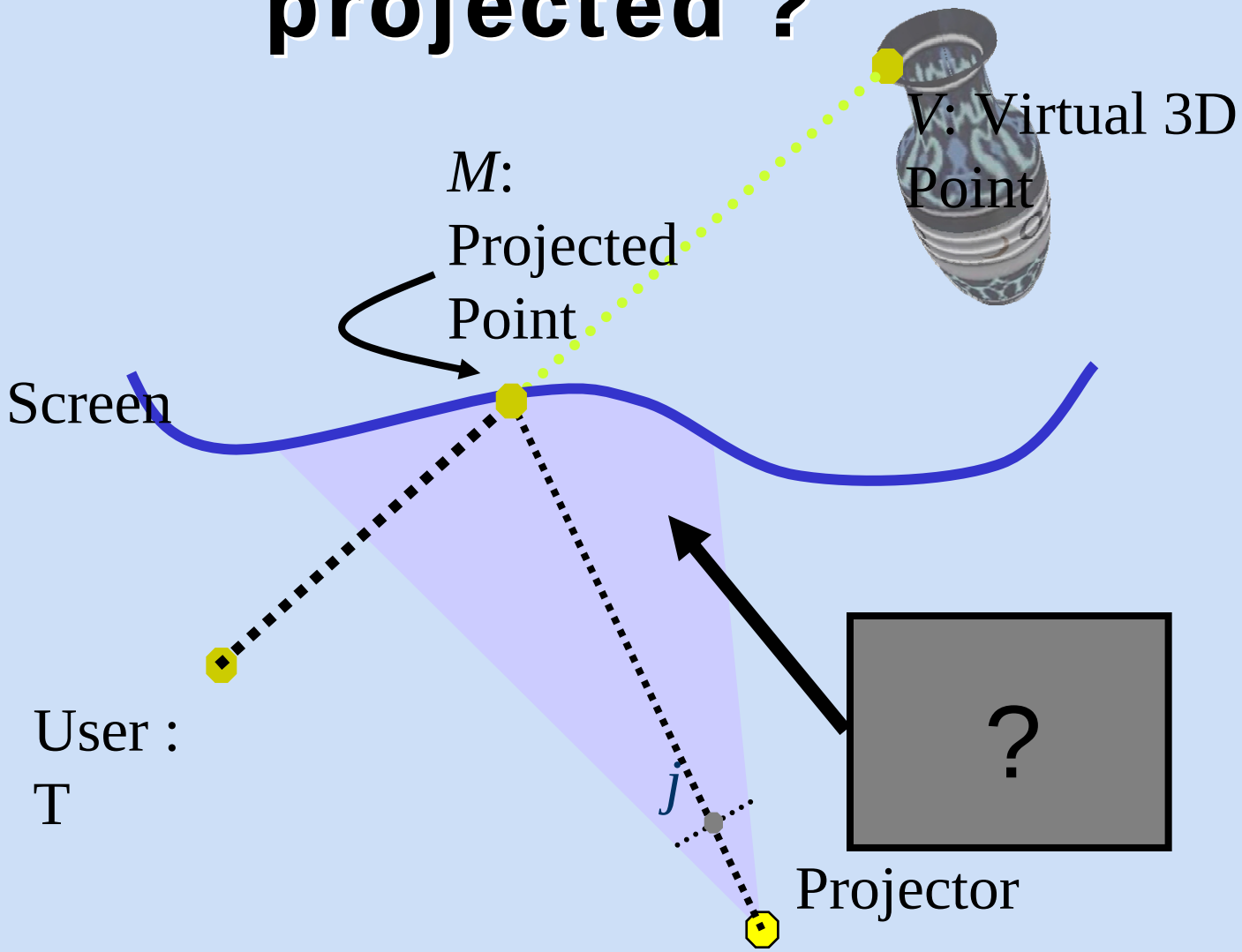
- Projection Screen Geometries
  - Planar 
  - Rectilinear 
  - Cylindrical
  - Spherical
  - Irregular

Planar Homography

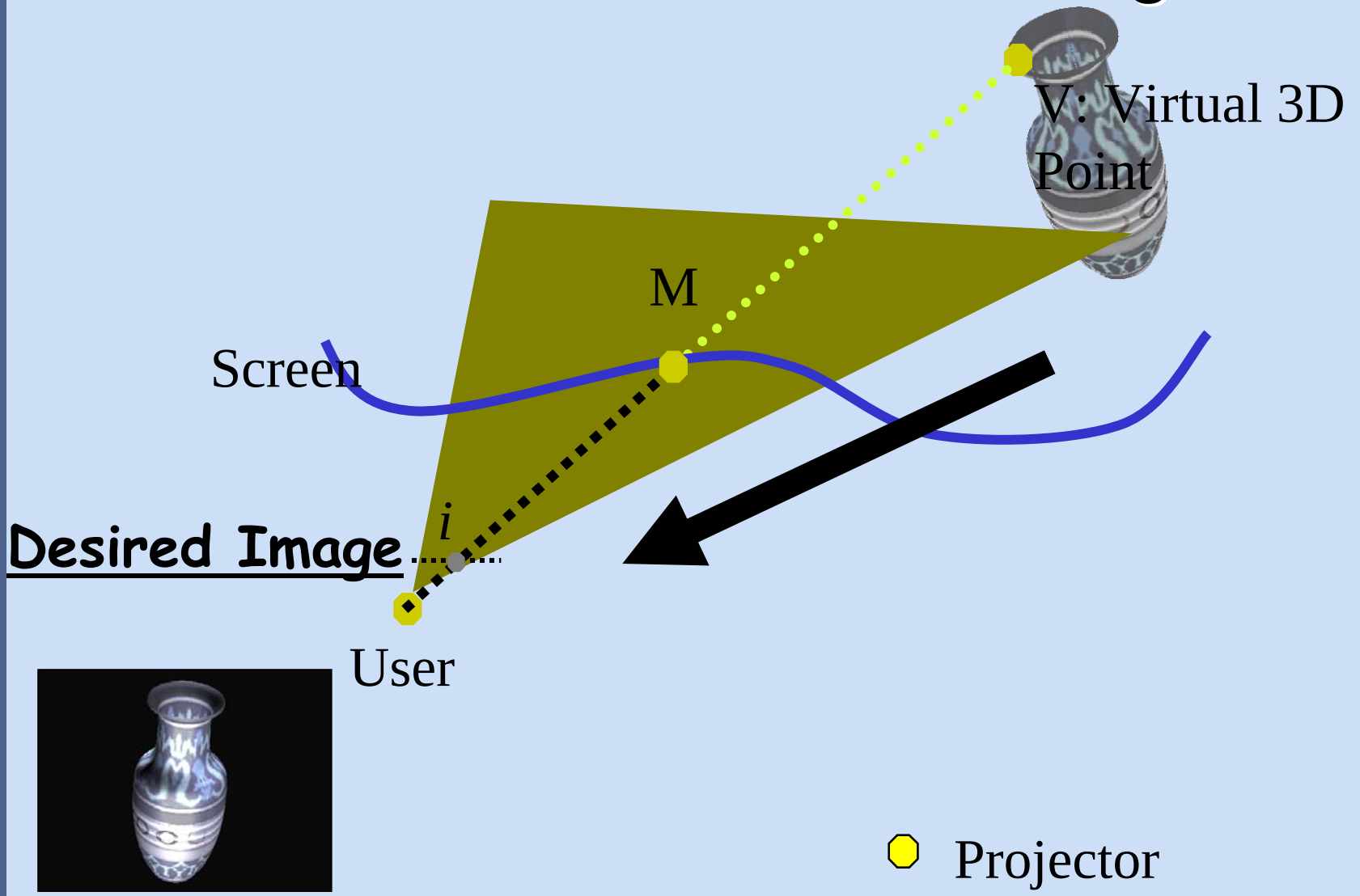
Quadric image transfer

Discretized Warping

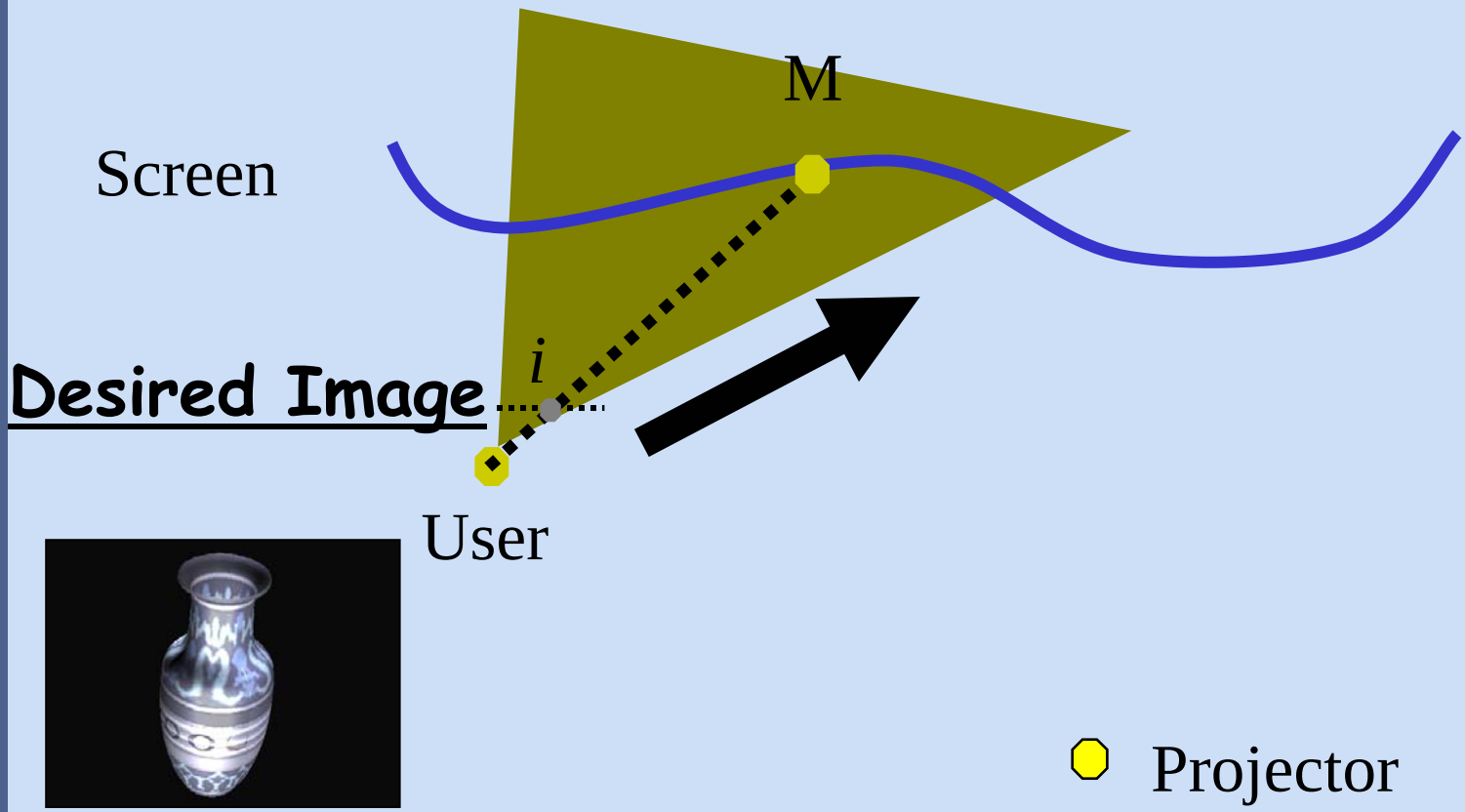
What image should be projected ?



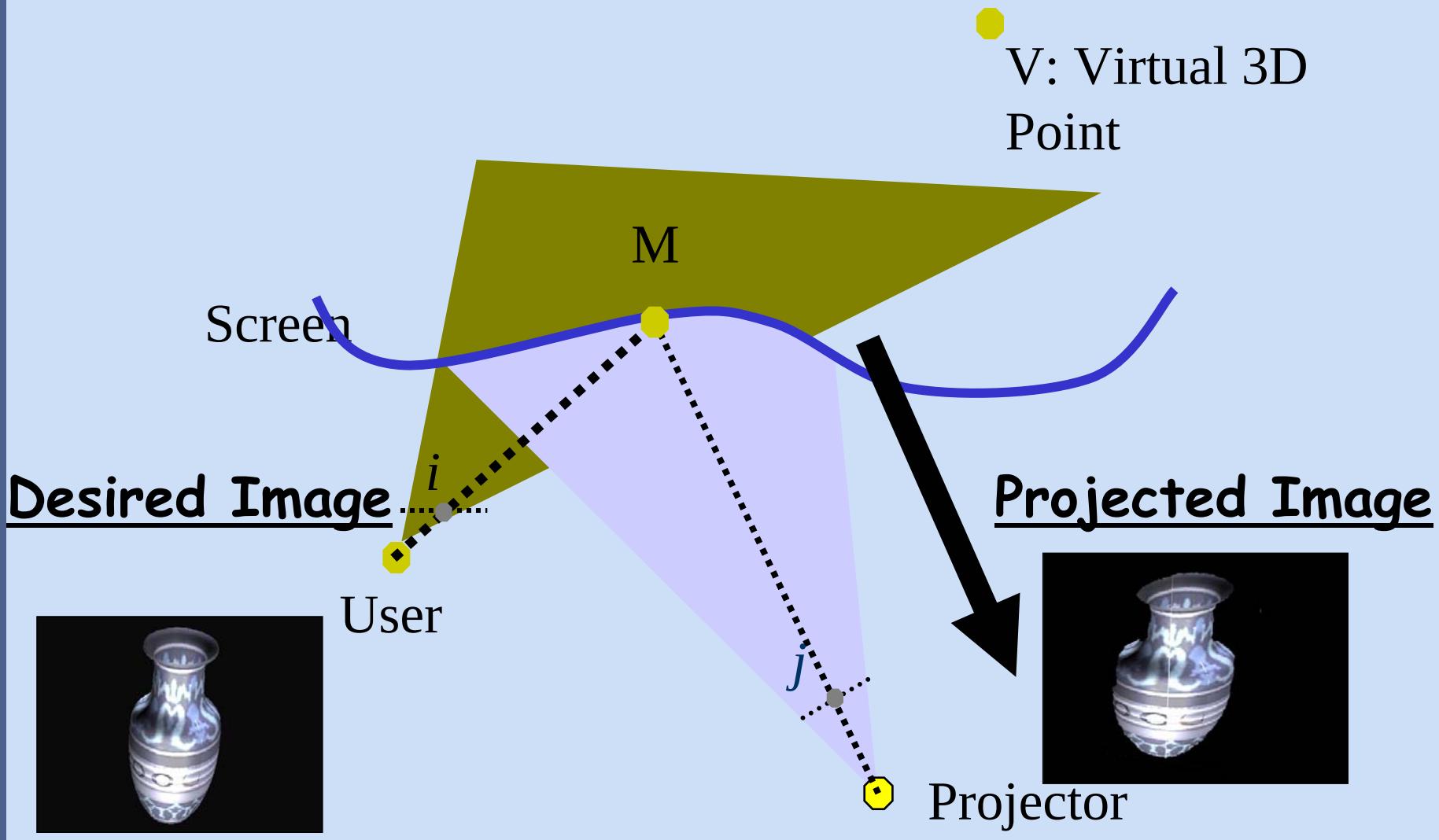
Step I : Calculate 'desired' image



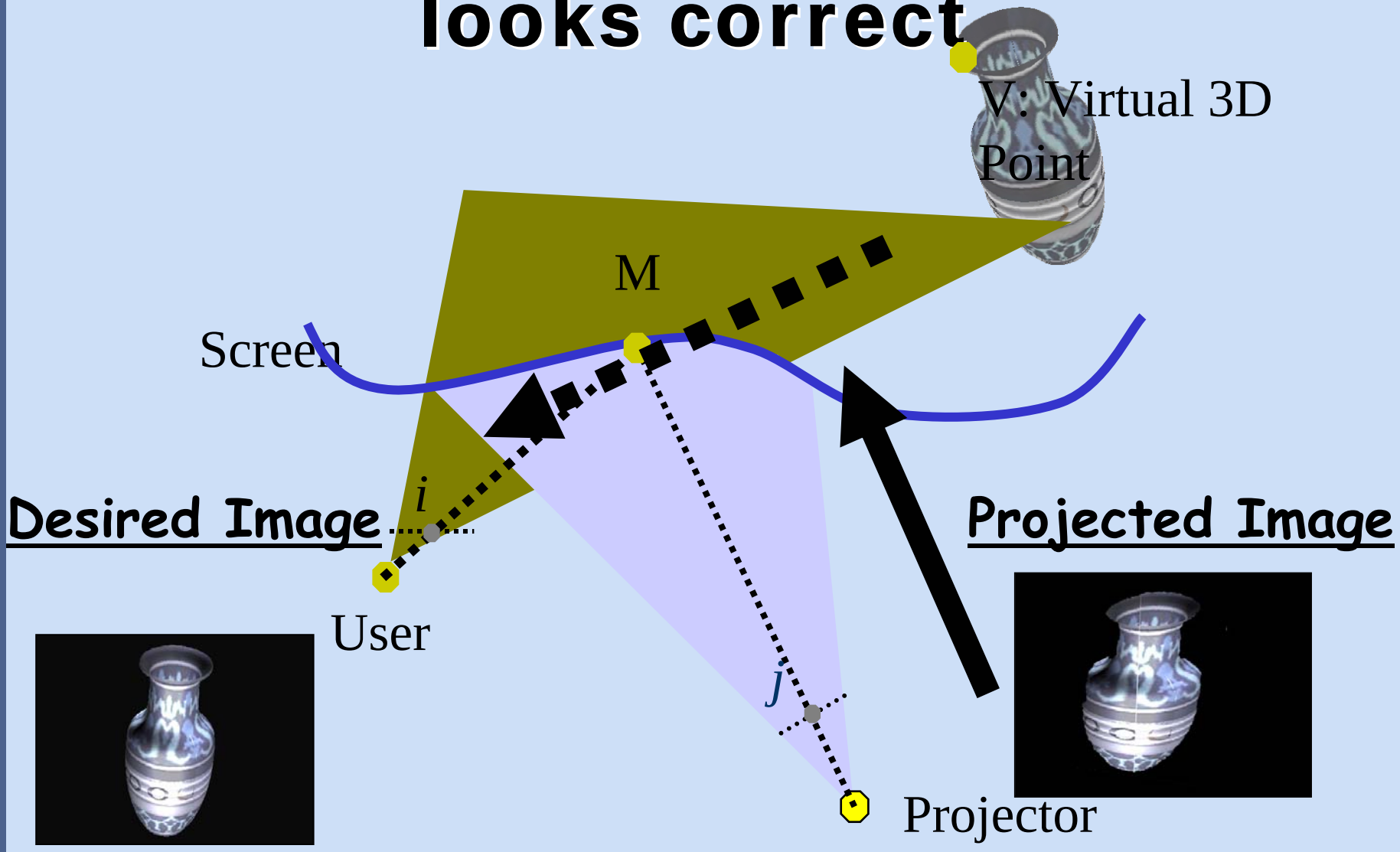
Step II :
‘Project’ the desired image from T



Step II :
Render this scenario from *P*



Result: Projecting a pre-warped image, so it looks correct



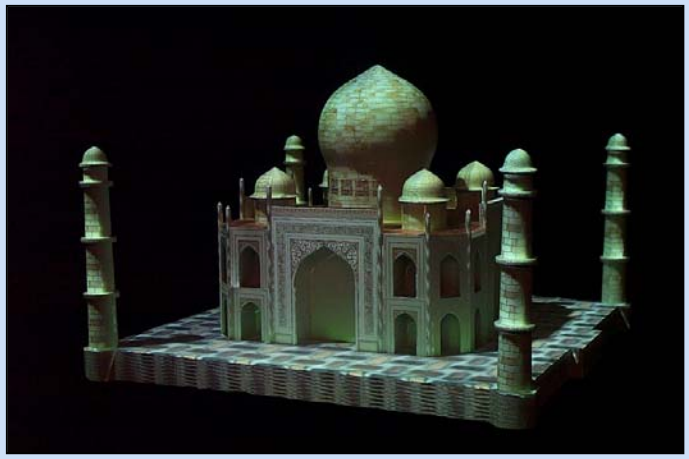
Non-planar Display

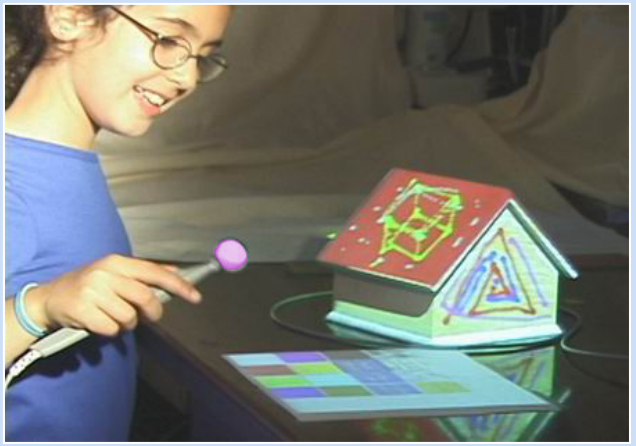
- Perspectively correct image for head-tracked user
- Details in [Raskar et al 'Office of the Future' Siggraph 1998]

- Step I
  - Compute desired image
  - Load in texture memory
- Step II :
  - With *projective texture,* map on display portal
  - Render projector's view of display portal

Projector-based Augmentation

- More Examples ..

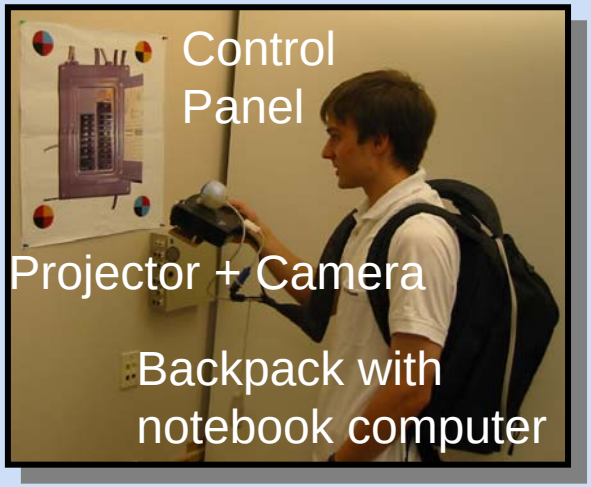


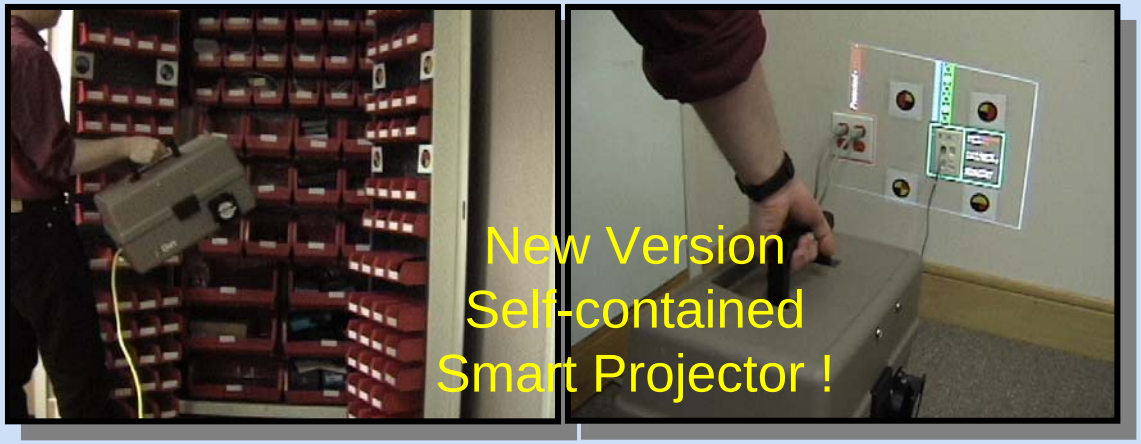


More info : www.ShaderLamps.com , Code available

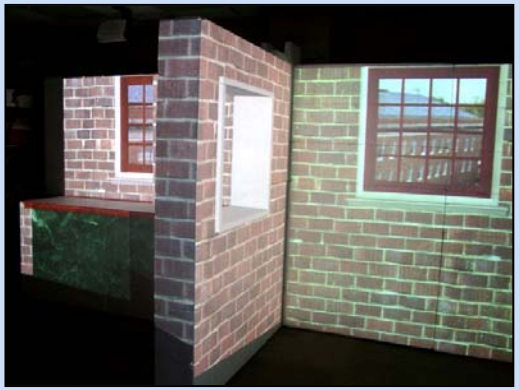
Training and Maintenance
(Projector-based Augmented Reality)
Raskar, Beardsley, Forlines

- Automatically add projected information
  - Training videos
  - Instruction manuals
- Detect pose and identity from pie-codes

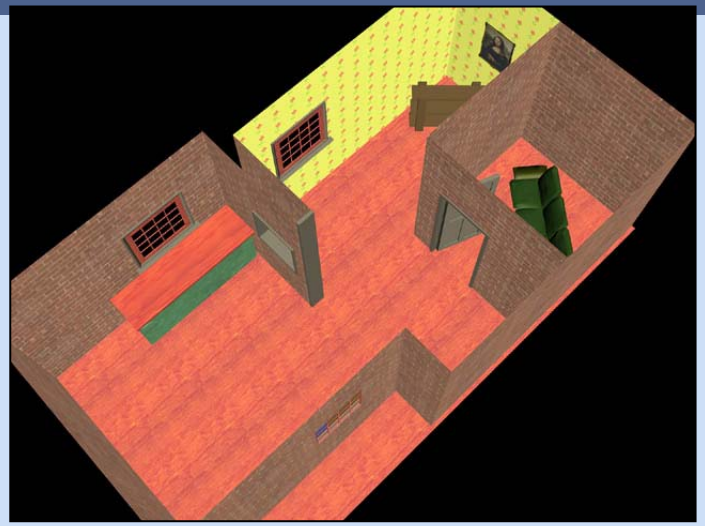




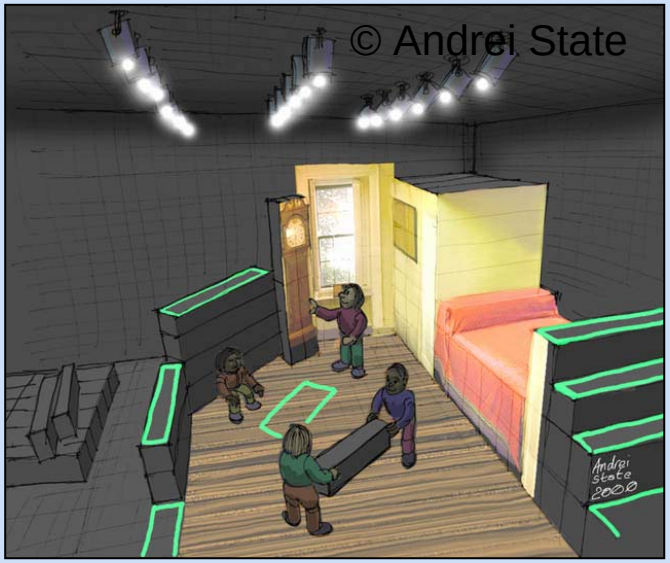
- Recreate Large Environments
  - ‘BeingThere’, walk-around
  - Human sized environments
  - Museums, Exhibitions



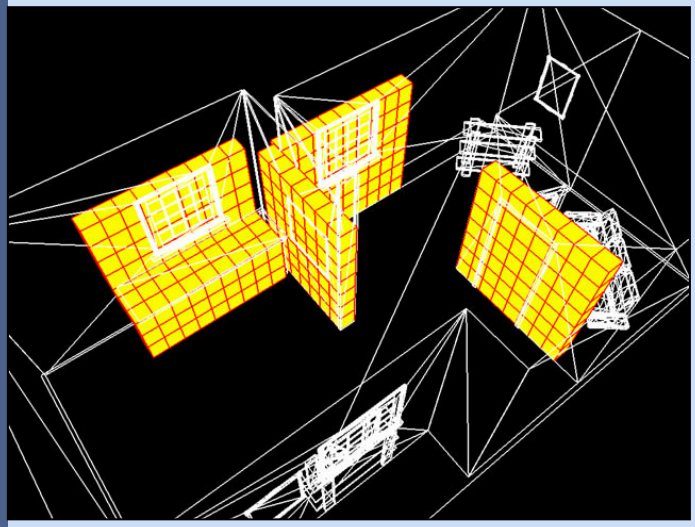
Kok-Lim Low, Greg Welch, Anselmo Lastra, Henry Fuchs. “Life-Sized Projector-Based Dioramas,” Proc. ACM Symposium on Virtual Reality Software and Technology 2001 (VRST 2001), November 2001.

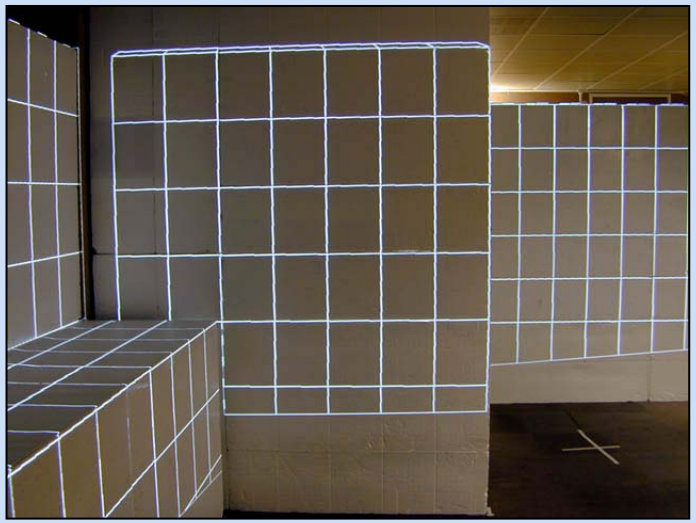


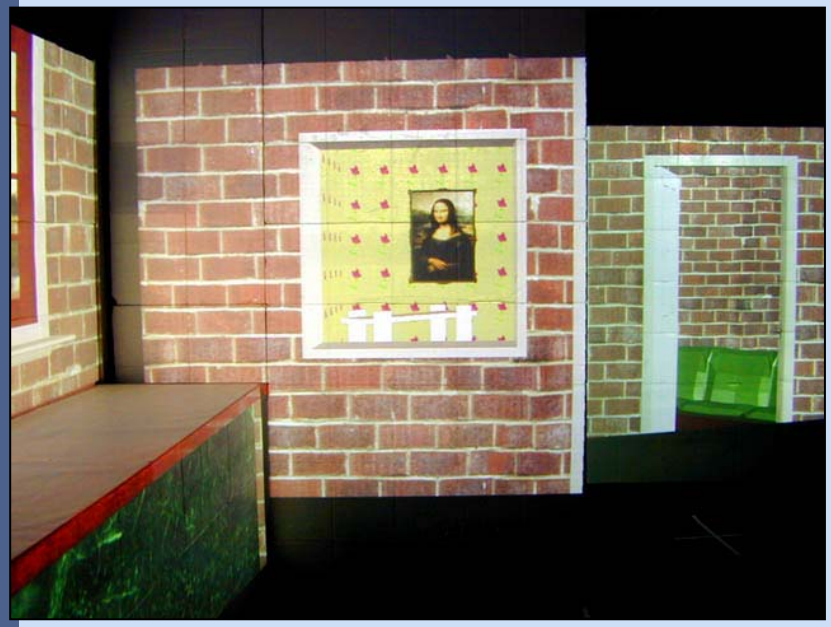
Desired Virtual Model

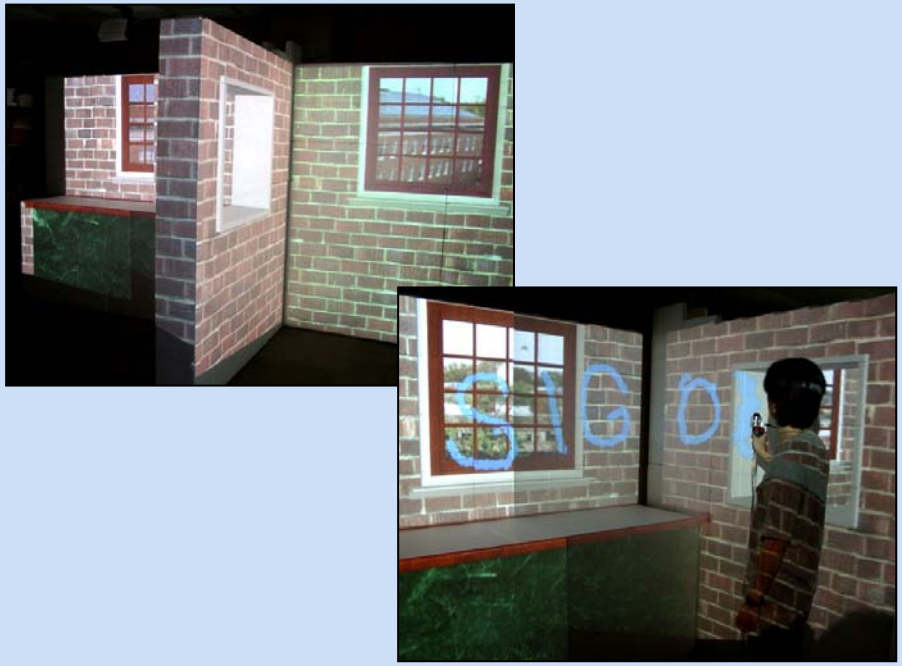


Projected Guidance for Placement

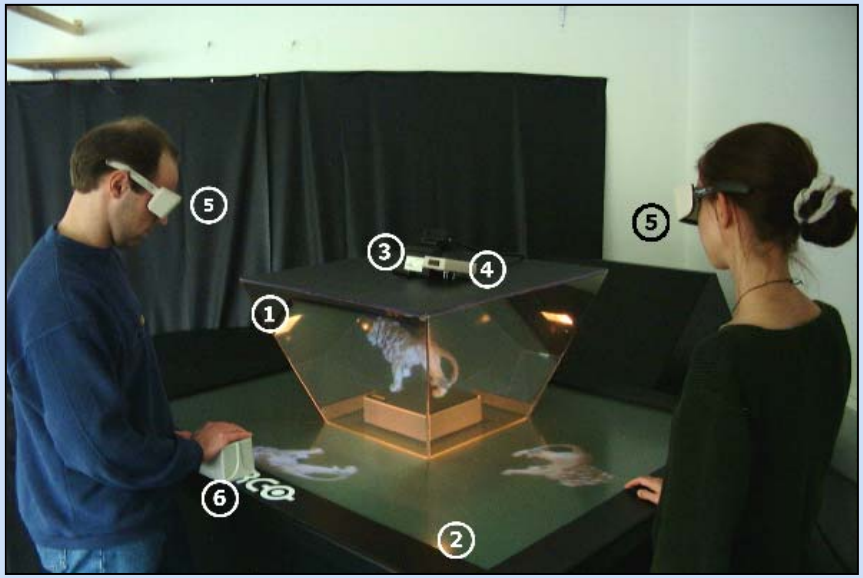


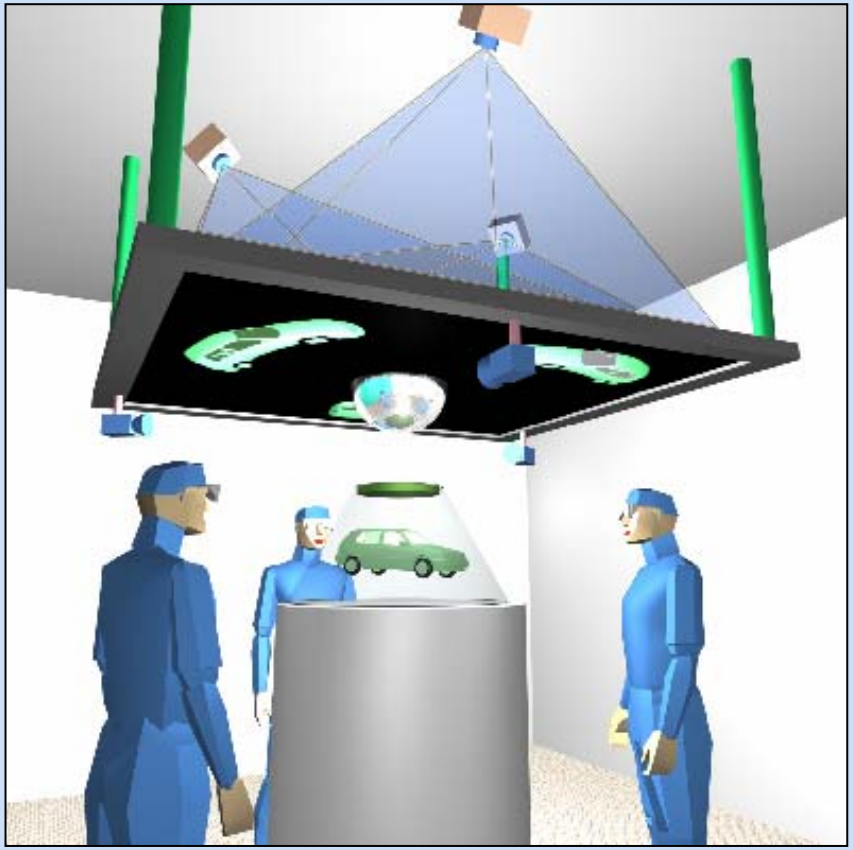






Projector-based AR





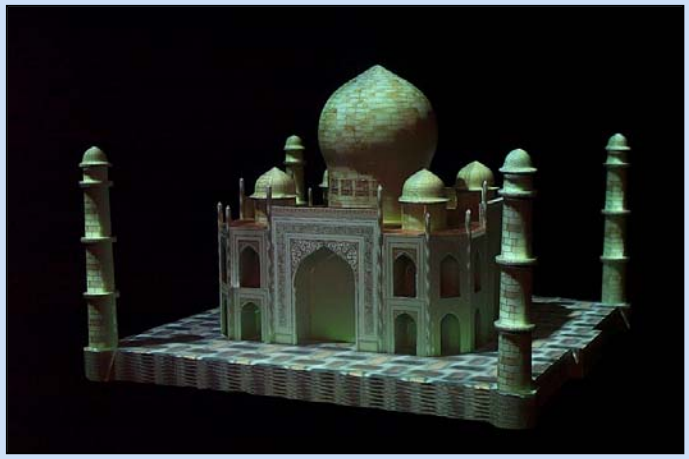
Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. 'The Virtual Showcase'. *IEEE Computer Graphics & Applications*, vol. 21, no.6, 2001.

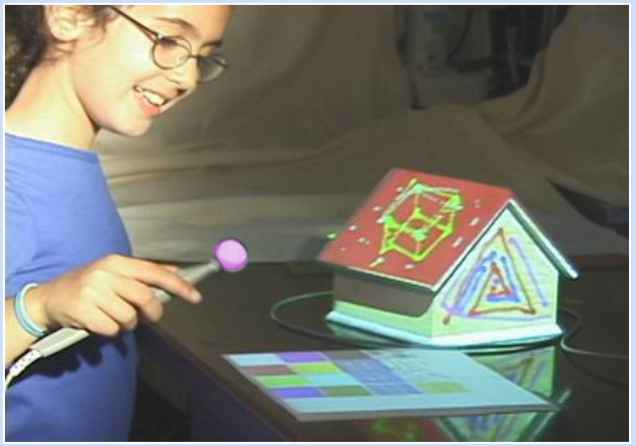
Acknowledgements

- MERL
  - Jeroen van Baar, Paul Beardsley, Remo Ziegler, Thomas Willwacher, Srinivas Rao, Cliff Forlines, Paul Dietz
- Office of the Future group at UNC Chapel Hill
  - Greg Welch, Kok-lim Low, Deepak B'padhyay, Aditi Majumder, Michael Brown, Ruigang Yang
  - Henry Fuchs, Herman Towles
  - Wei-chao Chen
- Mitsubishi Electric, Japan
  - Yoshihiro Ashizaki, Masatoshi Kameyama, Masato Ogata, Keiichi Shiotani
- Images
  - Oliver Bimber (Virtual Showcase images)
  - Marc Pollefeys (UNC Chapel Hill)
  - Apologies
    - (Not able to include recent work by others)

Projector-based Augmentation

- Useful paradigm for 3D graphics
- New methods to make it practical
- Many open problems and applications



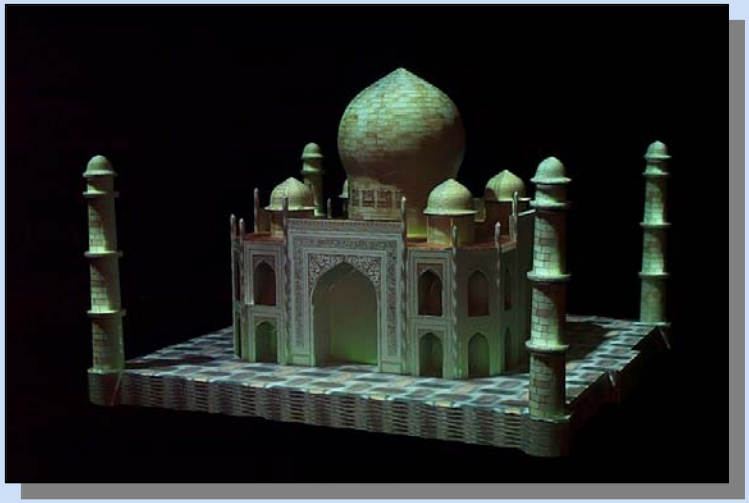


More info : www.ShaderLamps.com , Code available

Projector-based AR

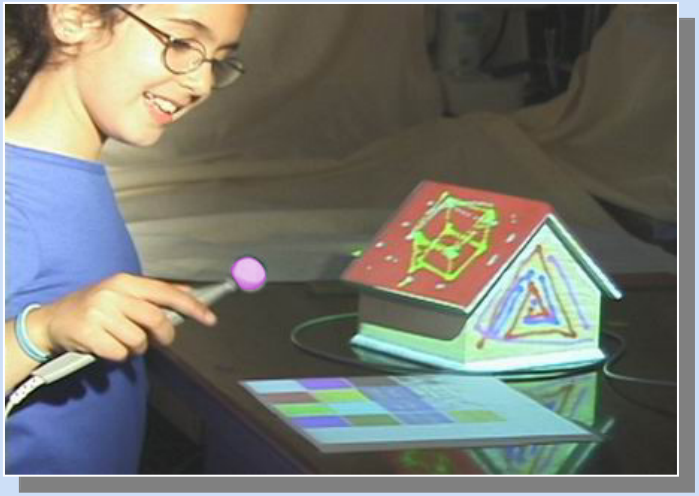
Virtual Reflectance

Virtual Illumination



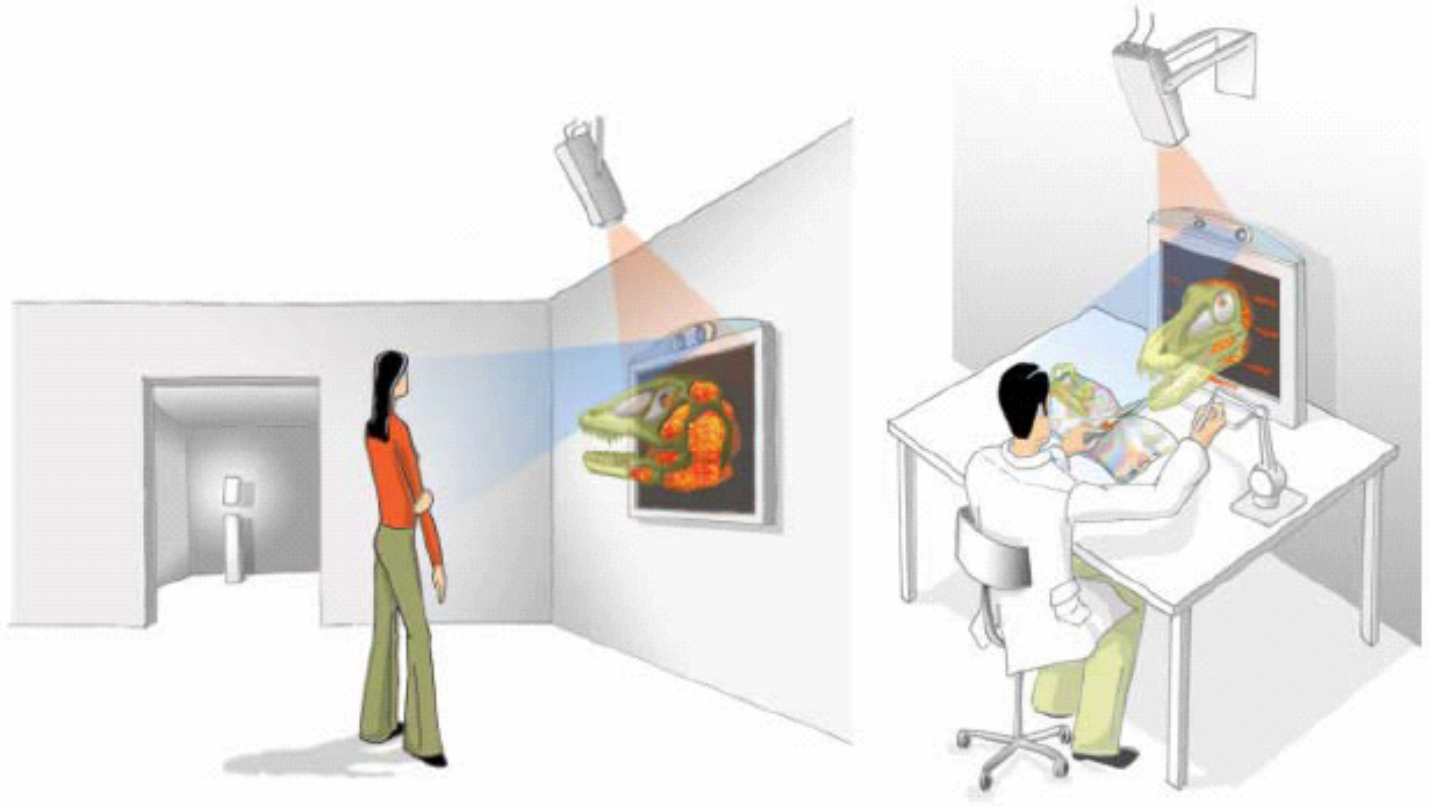
Virtual Motion

Interaction



www.ShaderLamps.com

HoloGraphics

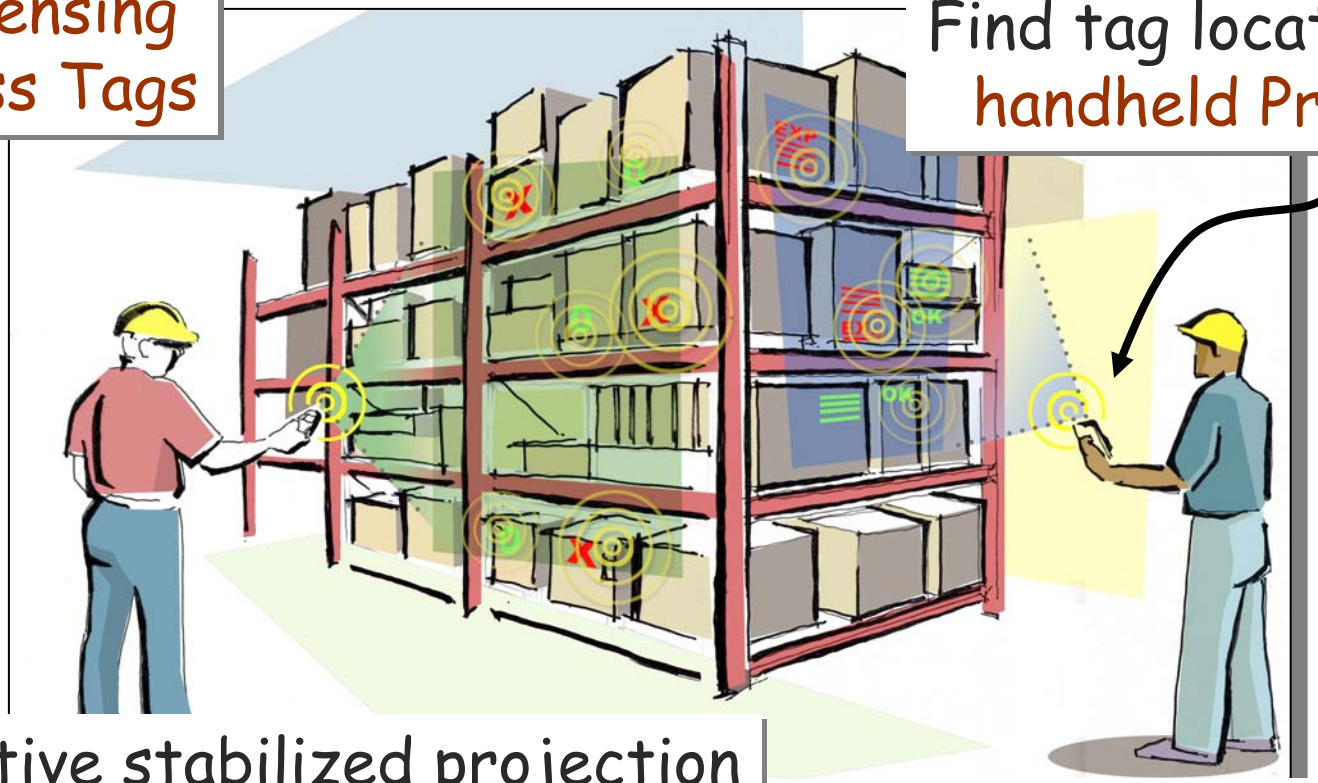


RFID ➡ **RFIG**

(Radio Frequency Id & Geometry)

Photosensing Wireless Tags

Find tag location using handheld Projector



Interactive stabilized projection

Many geometric ops

Multi-Projector Techniques for Real-Time Visualizations in Everyday Environments

O. Bimber

Outline

Introduction
Motivations and Applications

Geometric Correction
Planar, Non-Trivial, Complex Surfaces

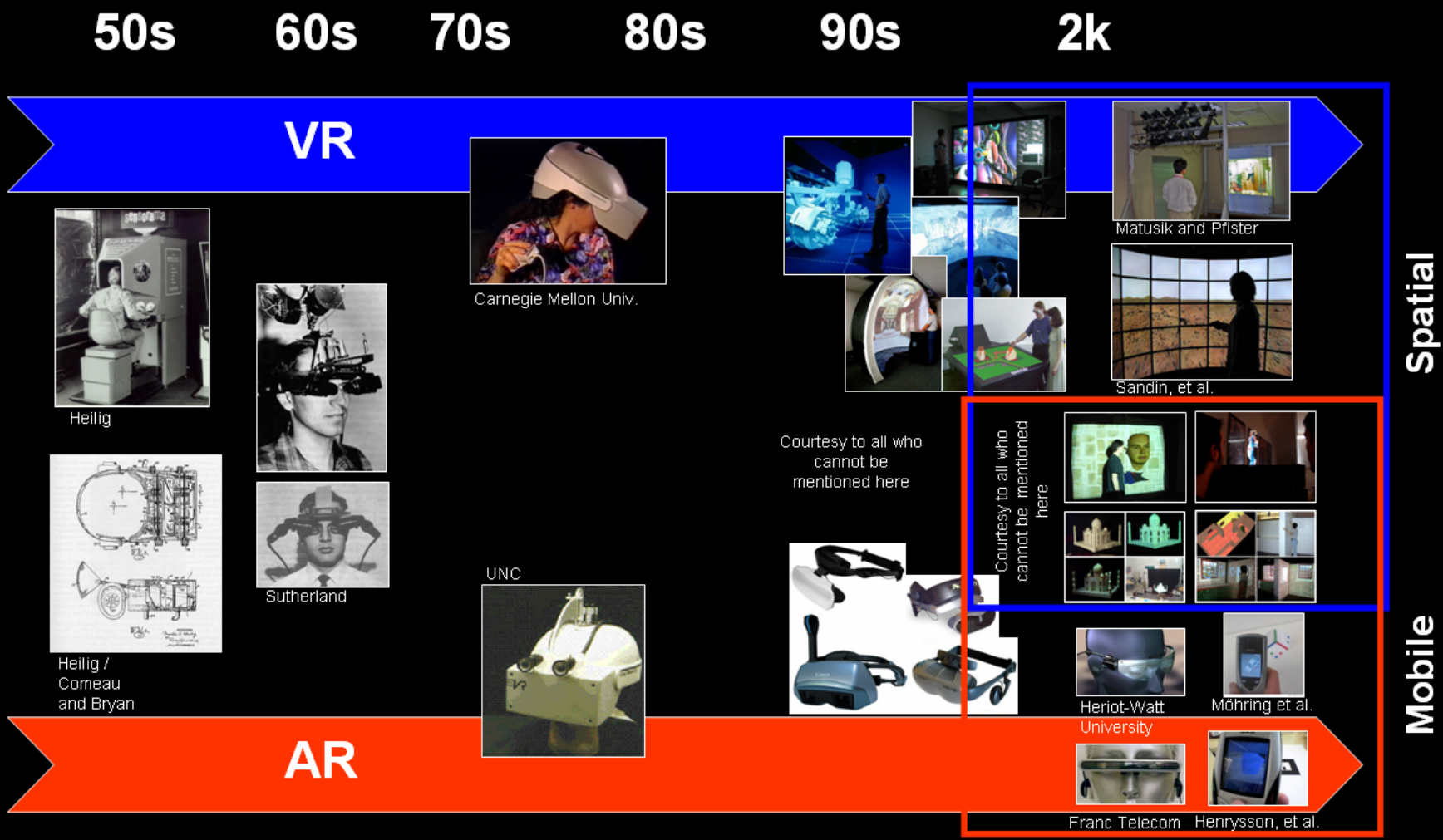
Radiometric Compensation
Local and Global Light Effects

Advanced Techniques
View-Dependence, Multi-Focal Projection, Light Transport

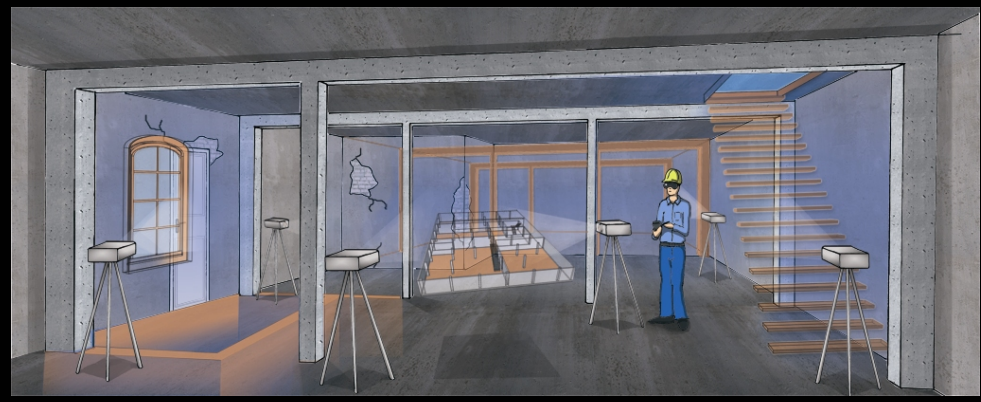
Outlook
Limitations and Future Work

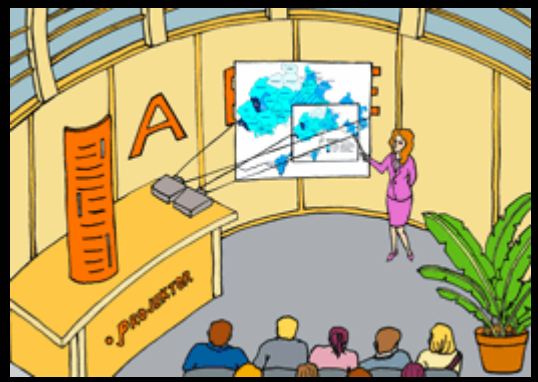
Introduction

Evolving Evolution

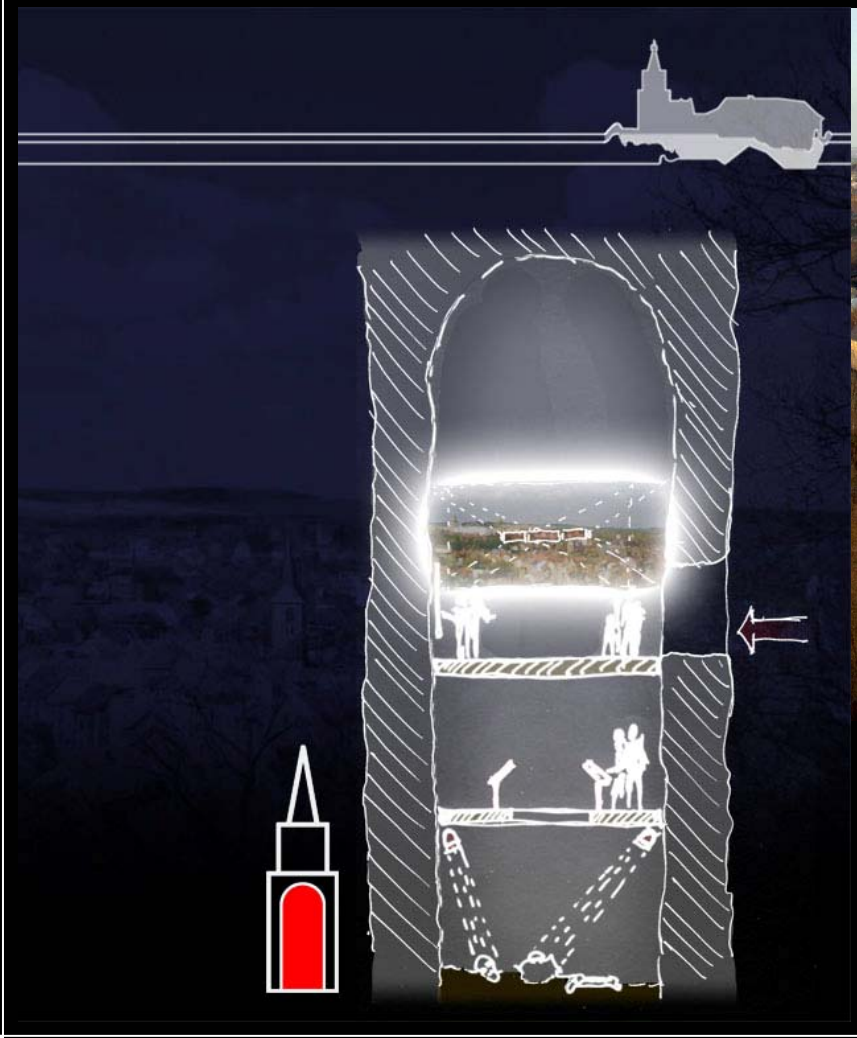


Motivation: Projection on Everyday Surfaces



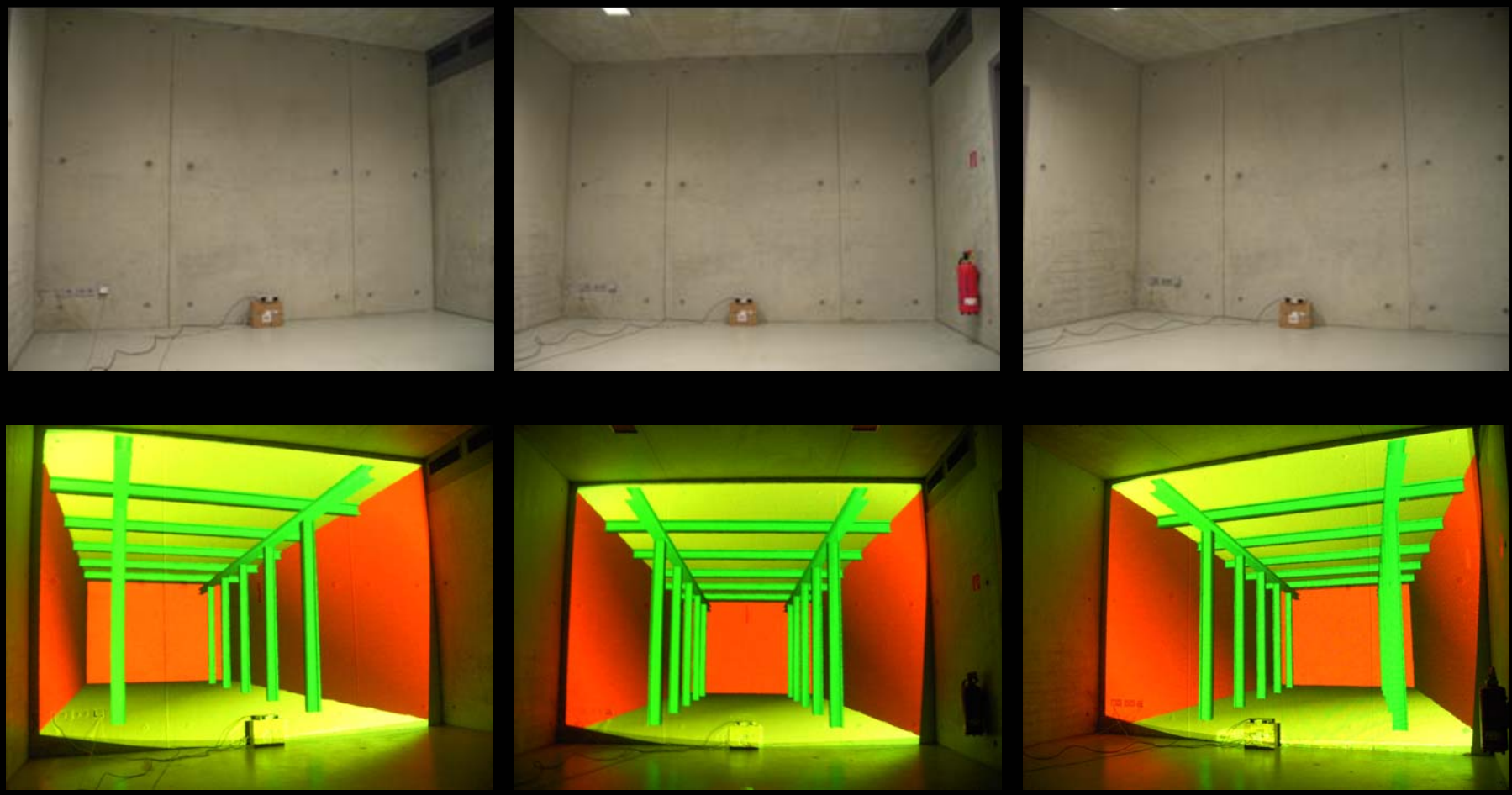


Application: Historic Sites and Museums



360° Surround Projection in Castel Tower
(Running project in coop. with Bennert Group)

Application: Architectural Visualization



Bimber et al, IEEE/ACM ISMAR 2005

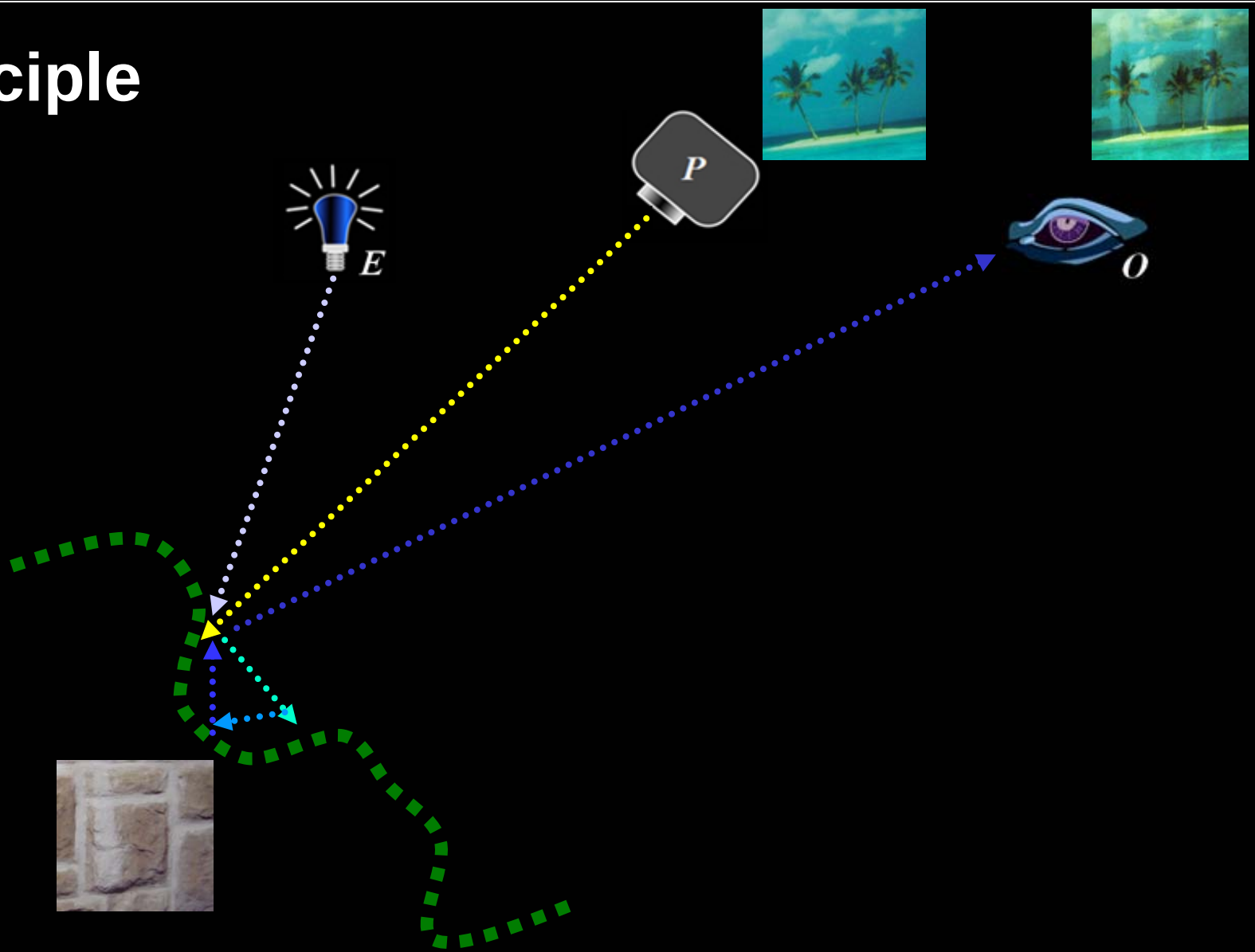
On-Site Architectural Visualizations
(Running project in coop. with Architecture Faculty, BUW)

Application: Pocket Projectors

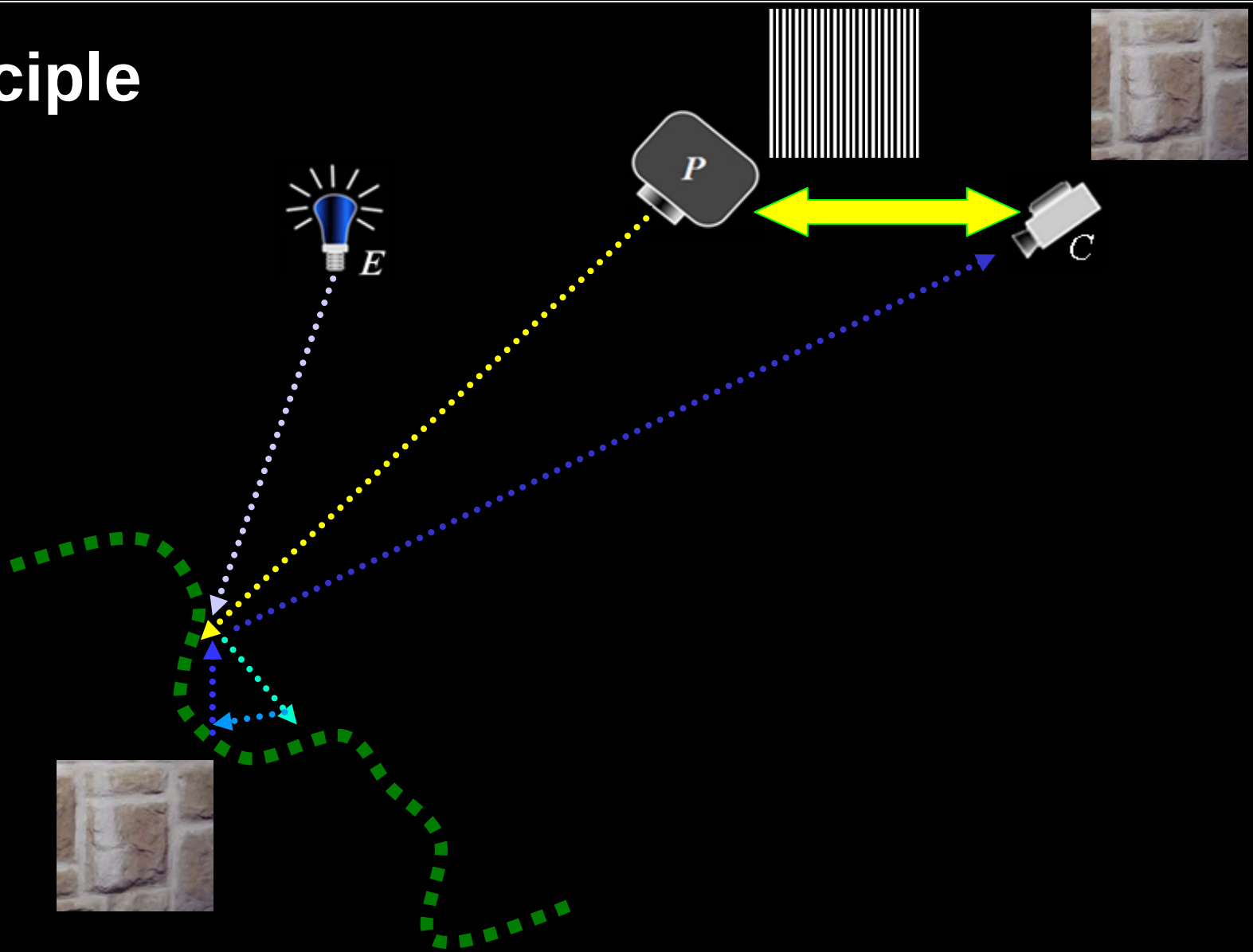
Courtesy: InFocus

Courtesy: Siemens

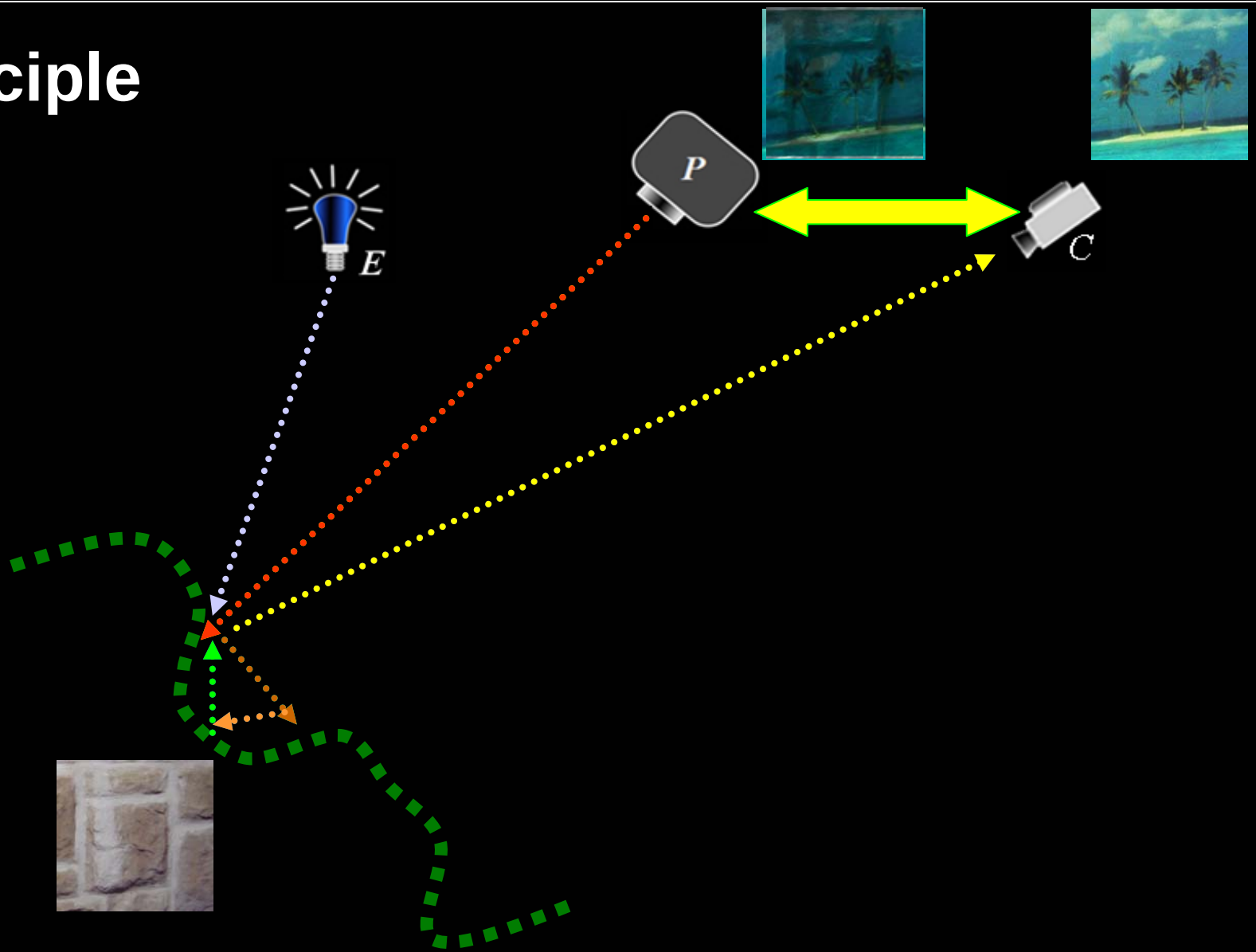
Principle



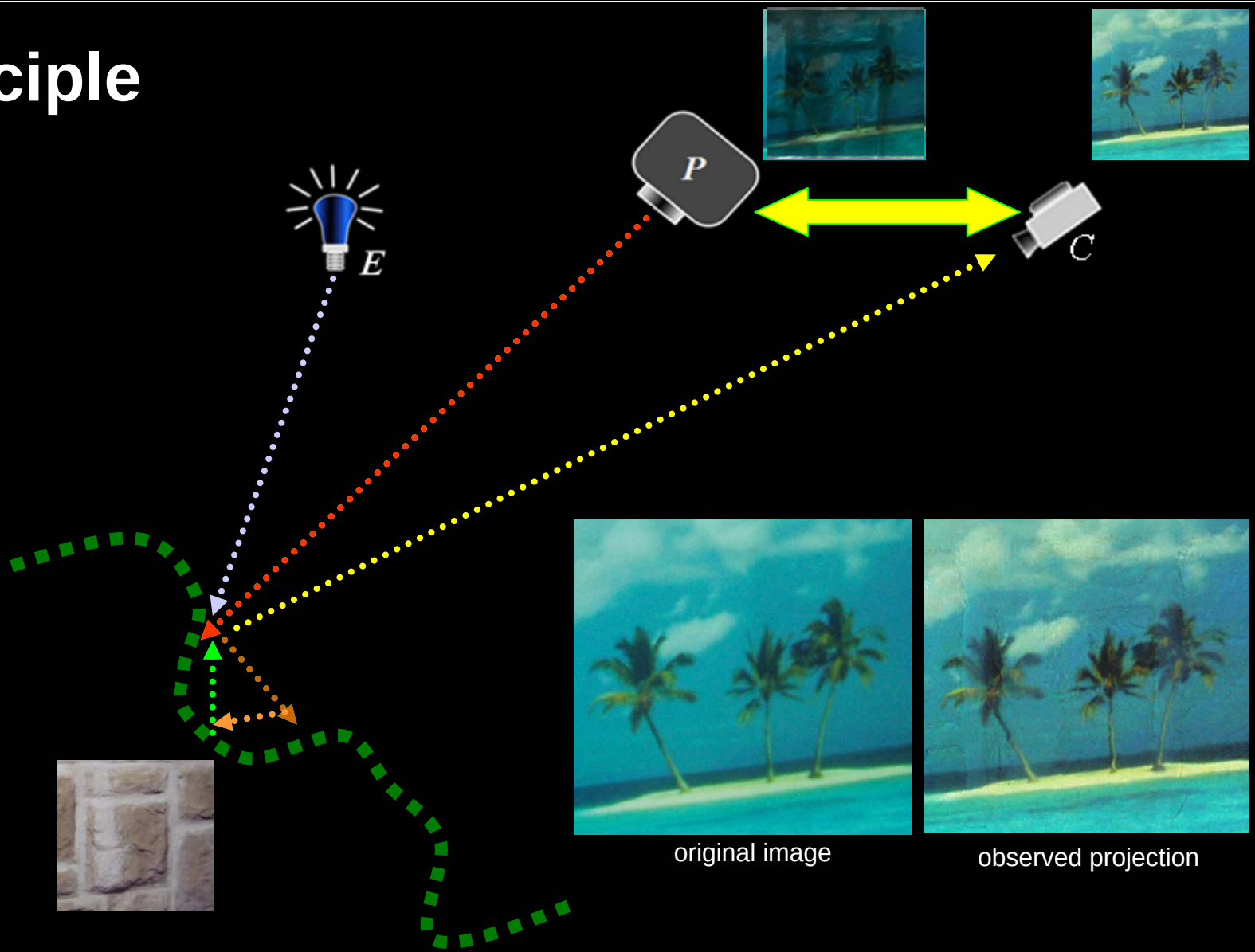
Principle



Principle



Principle



original image

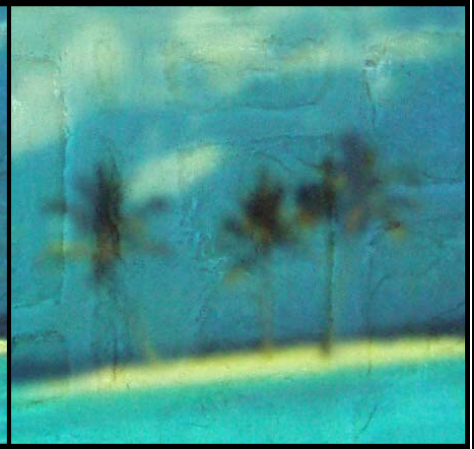
observed projection

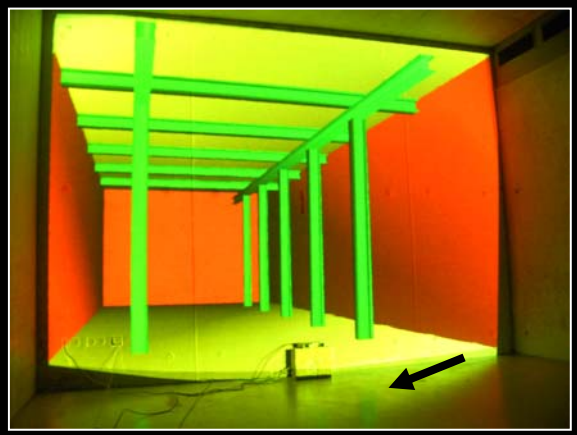
Some Challenges

color blending

geometric warping

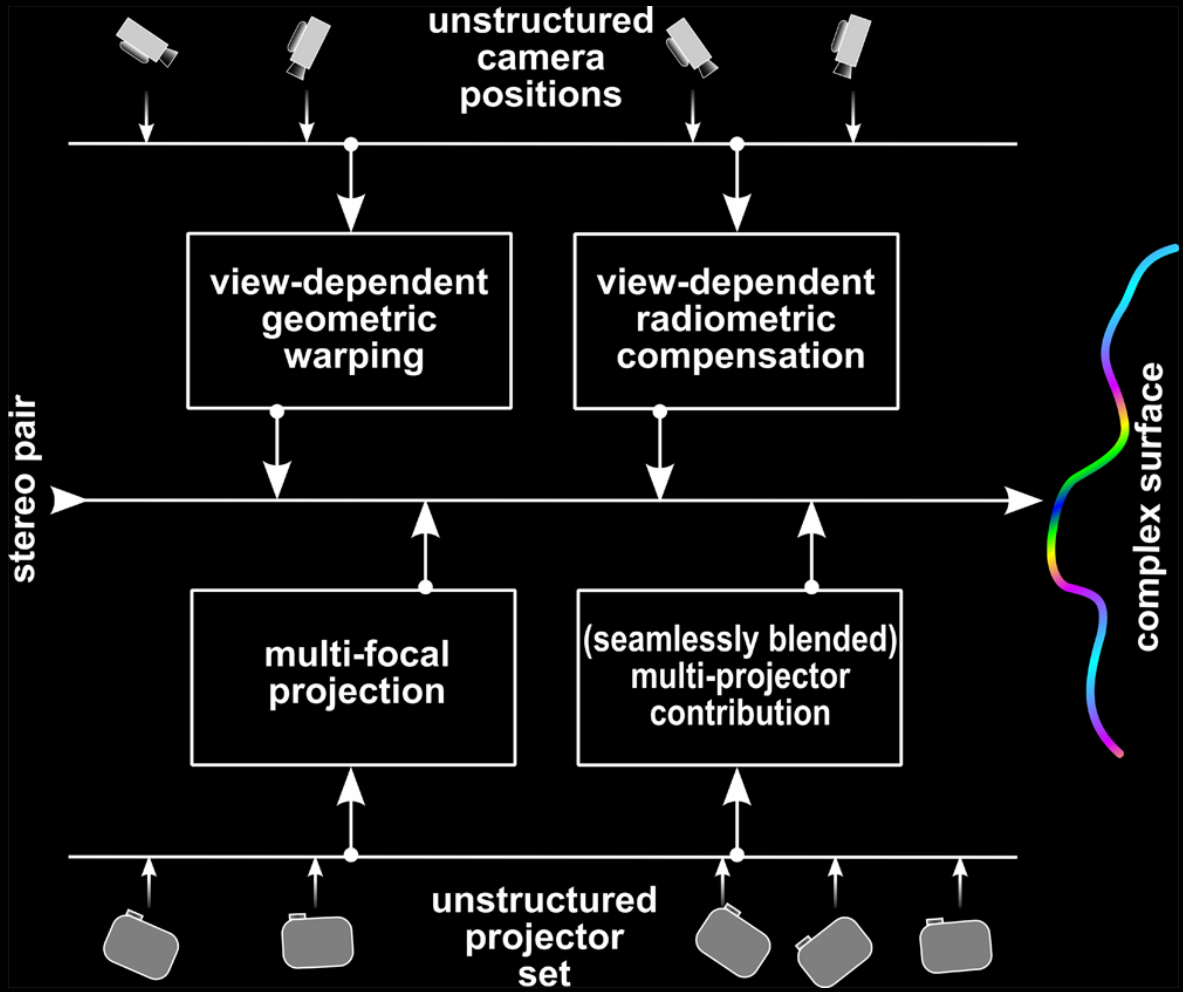
misregistration


regional defocus


scattering

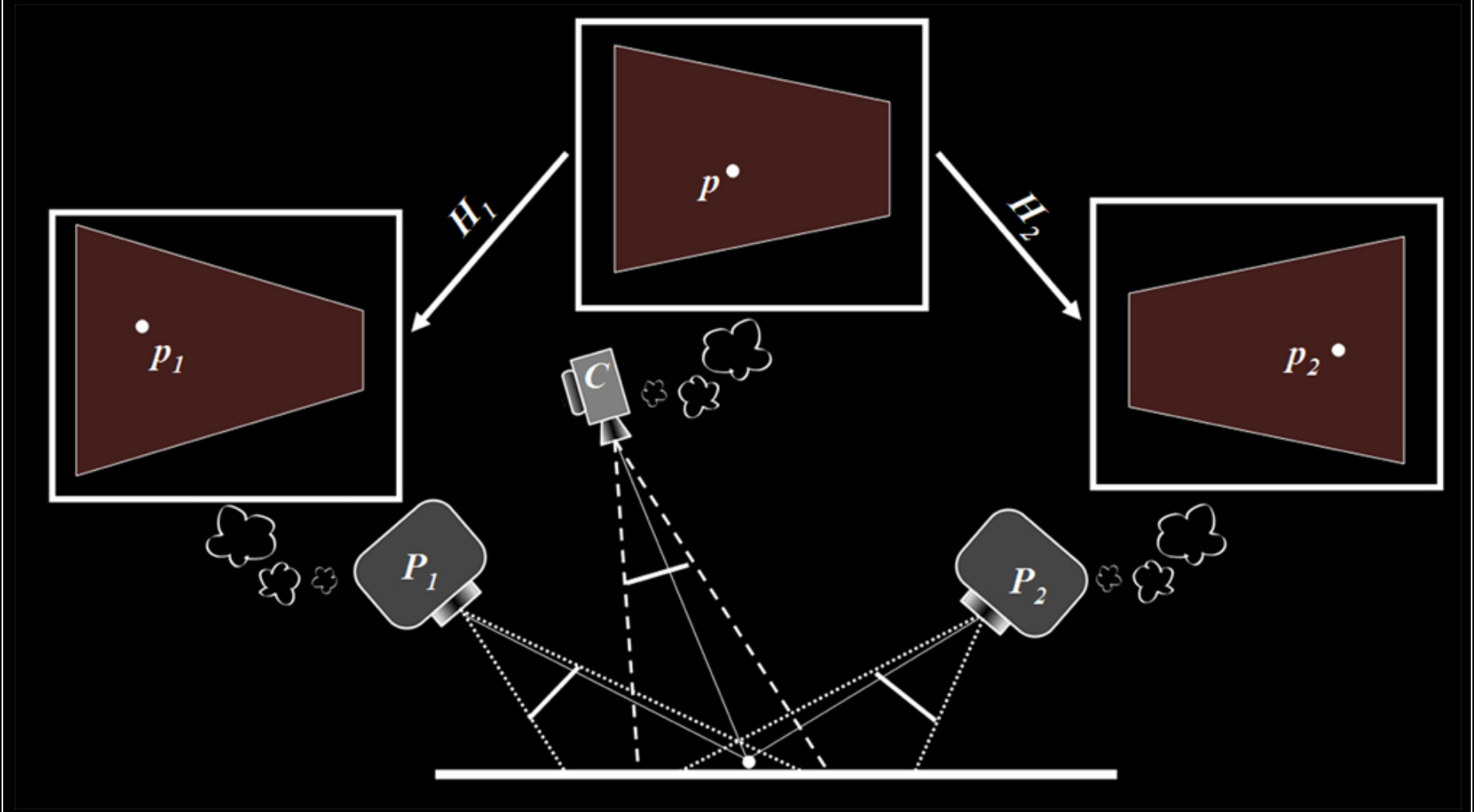
specular reflection, refraction, sub-surface scattering, inter-reflections, dispersion, diffraction, etc.

A Multi-Projector-Camera Approach



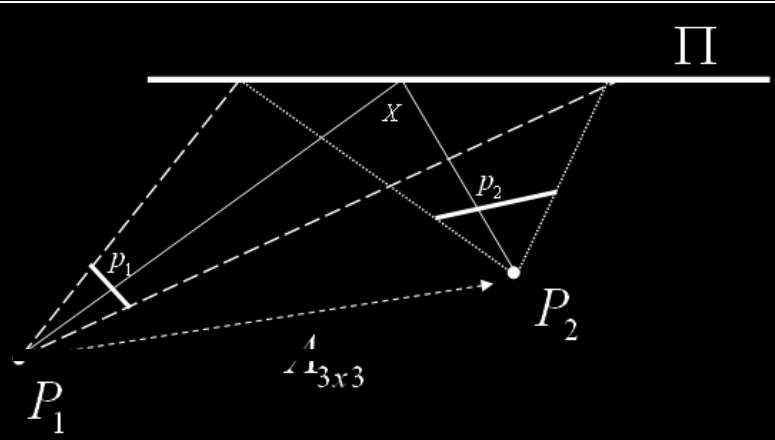
Geometric Correction

Planar Surfaces



Homography

- homography is a mapping between two projections over a plane
- can map pixel coordinates from one perspective to another
- equation system has to be solved to determine 8 parameters of matrix A
- can be used directly in transformation pipeline by multiplying the following matrix after projection (without perspective division):



$$p_2 \cong A_{3x3} p_1$$

$$\begin{bmatrix} p_{2x} \\ p_{2y} \\ 1 \end{bmatrix} \cong \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix} \begin{bmatrix} p_{1x} \\ p_{1y} \\ 1 \end{bmatrix}$$

$$A_{4x4} = \begin{bmatrix} a_{11} & a_{12} & 0 & a_{13} \\ a_{21} & a_{22} & 0 & a_{23} \\ 0 & 0 & 1 & 0 \\ a_{31} & a_{32} & 0 & 1 \end{bmatrix}$$

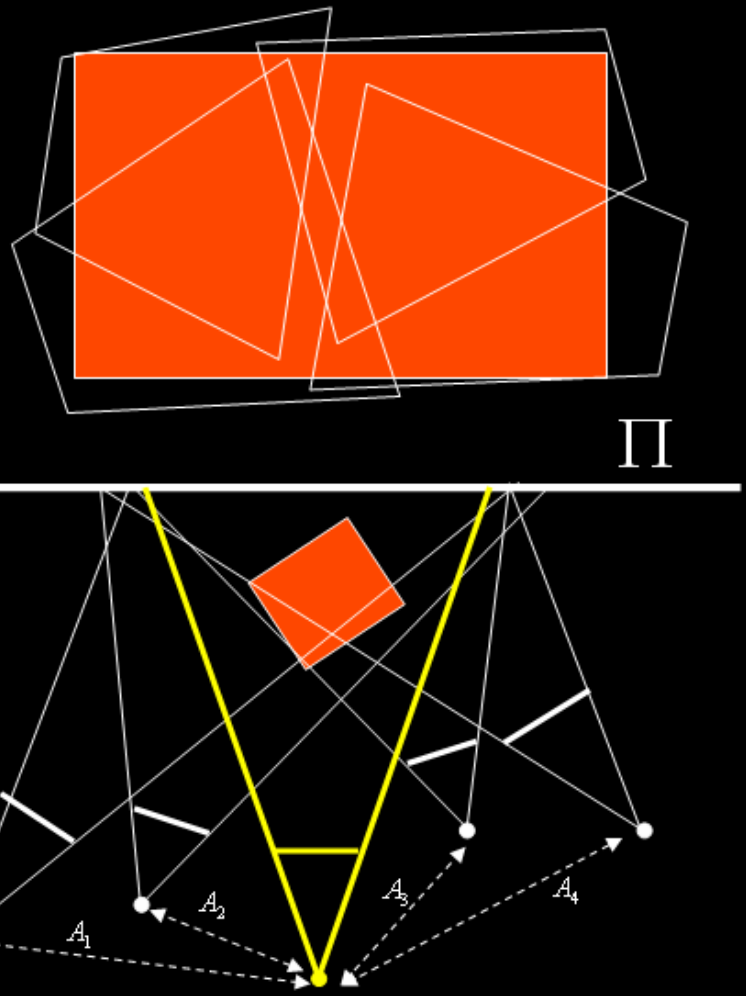
- ensure intact depth values with (approximately)

$$p_{2z} = \frac{p_{1z}}{(a_{31}p_{1x} + a_{32}p_{1y} + p_{1w})} \in [-1,1]$$

$$A_{4x4} = \begin{bmatrix} a_{11} & a_{12} & 0 & a_{13} \\ a_{21} & a_{22} & 0 & a_{23} \\ 0 & 0 & 1-|a_{31}|-|a_{32}| & 0 \\ a_{31} & a_{32} & 0 & 1 \end{bmatrix}$$

Multi-Projector Registration

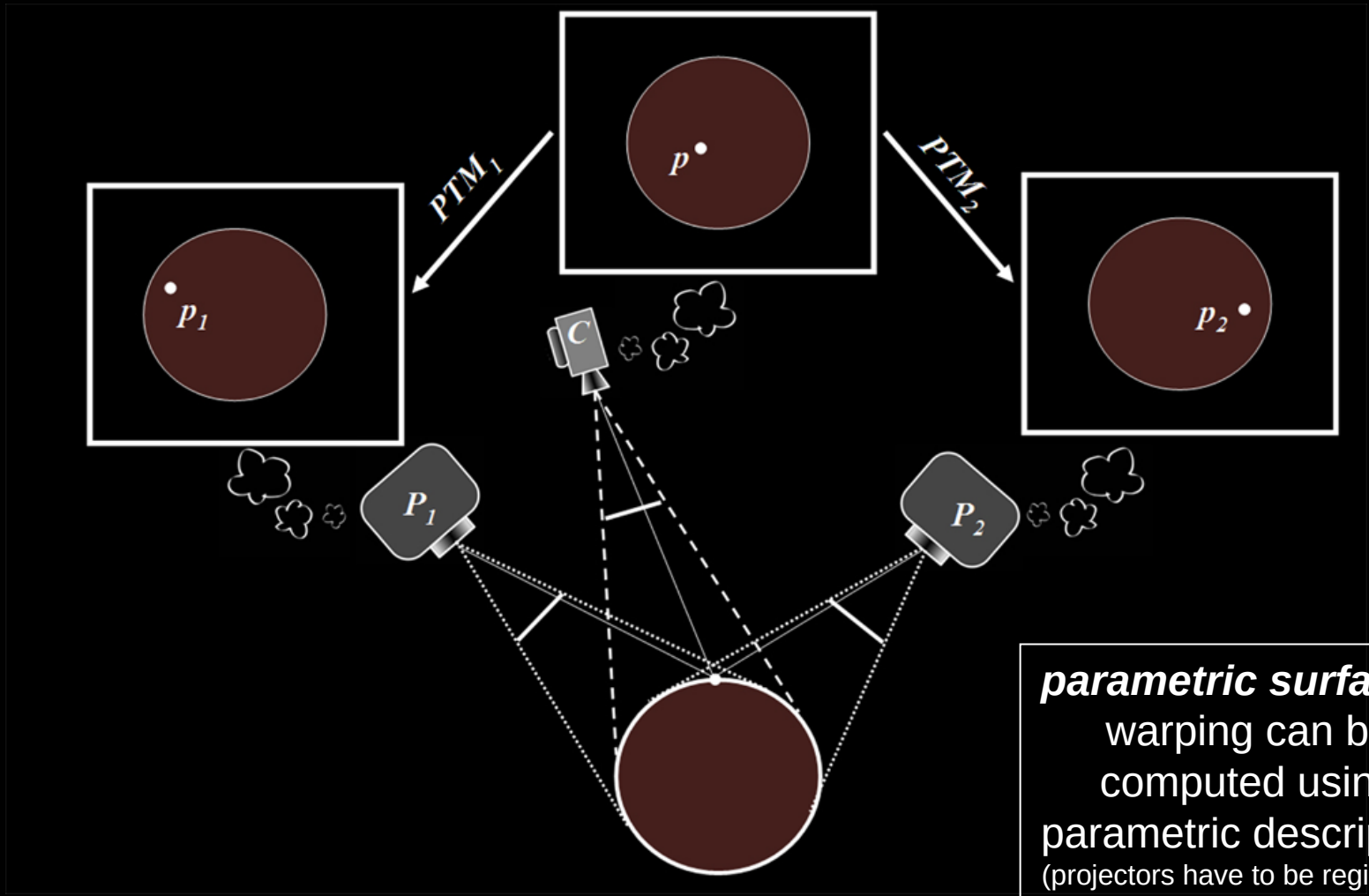
- registering multiple projectors onto a common planar surface
- map all perspective into a single target perspective via homographies
- target perspective can be camera perspective
  - automatic determination of matrix parameters via structured light
- rendering
  - render image for target perspective (if target perspective is **orthogonal** to plane, then it can be done with an off-axis projection of an observer!)
  - map pixels into individual projector views (i.e., multiply 4x4 version of homography matrix onto matrix stack [after projection] and ensure that depth values remain intact!)



Example: Tiled Projection Screens

Courtesy: Brown, et al.,
IEEE TVCG, 2005

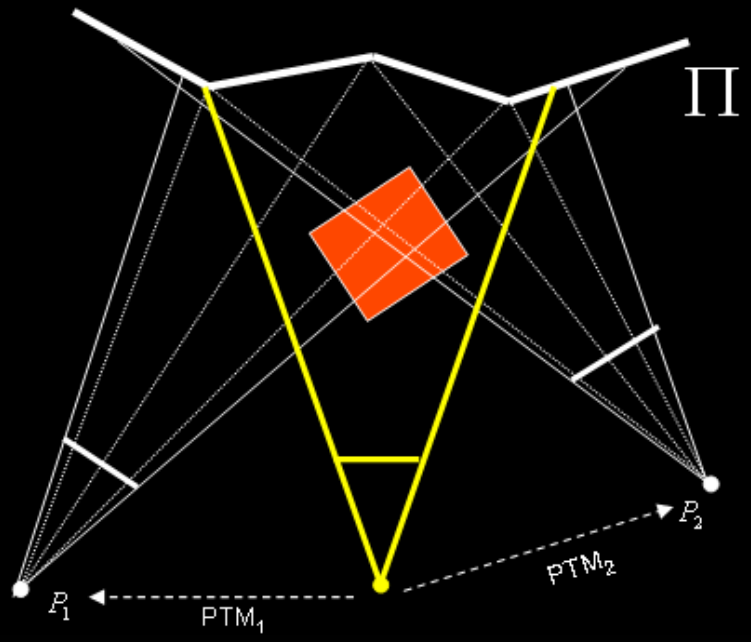
Non-Trivial Surfaces



parametric surfaces: warping can be computed using parametric description (projectors have to be registered)

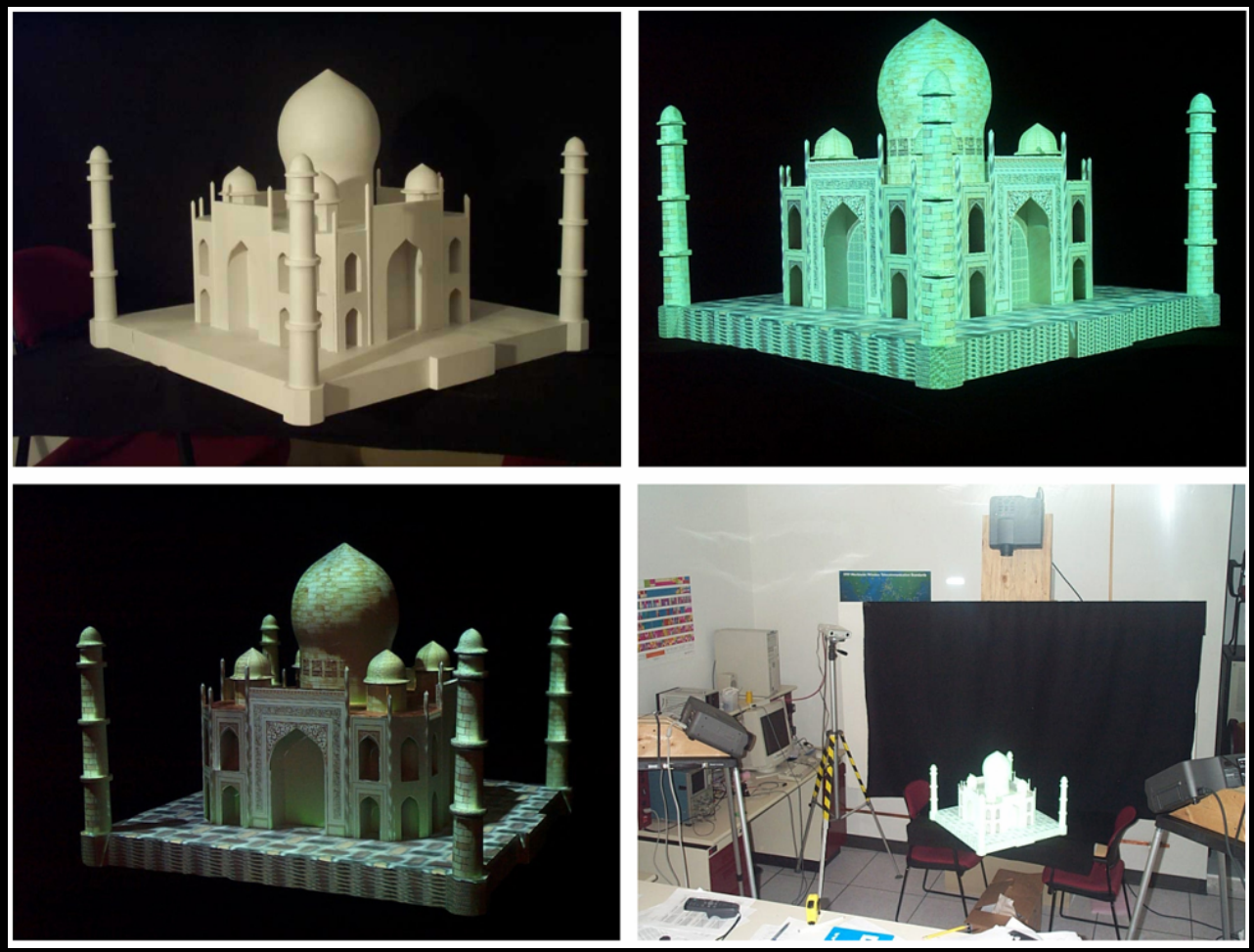
Projective Texture Mapping

- given a geometric definition of the surface
  - scan or model
- determine intrinsic and extrinsic of projectors with respect to surface
  - measure projections of known 3D surface points on image plane of projector and solve equation system to determine parameters of matrix
- define virtual camera with same parameter for each projector
- render 3D model of surface, textured with images, from perspective of projectors/virtual cameras
- texture coordinates can be automatically generated from target perspective via projective texture mapping



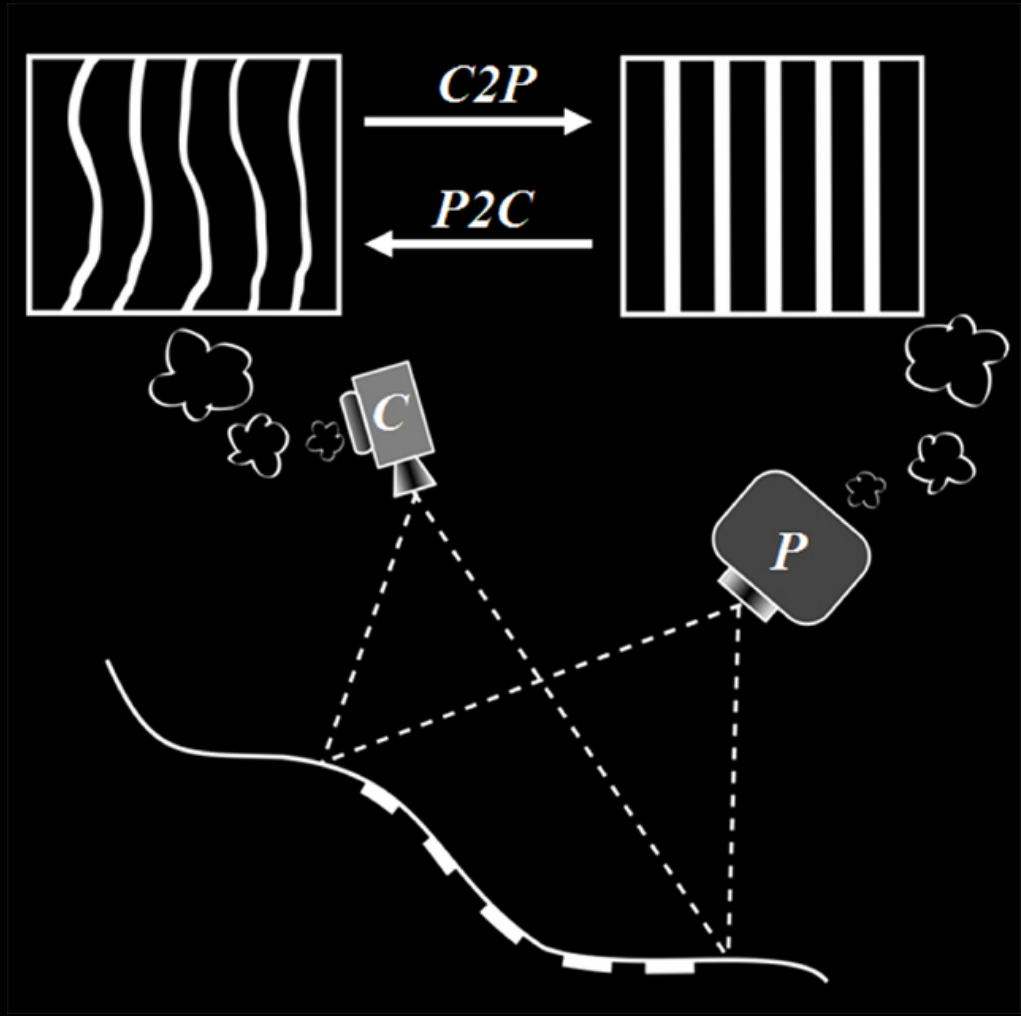
$$\begin{pmatrix} wx \\ wy \\ wz \\ w \end{pmatrix} = \begin{bmatrix} f & \cdot & \cdot & \cdot \\ \cdot & f & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 \\ \cdot & \cdot & 1 & \cdot \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Example: Shader Lamps



Courtesy: Raskar, et al., EGRW 2001

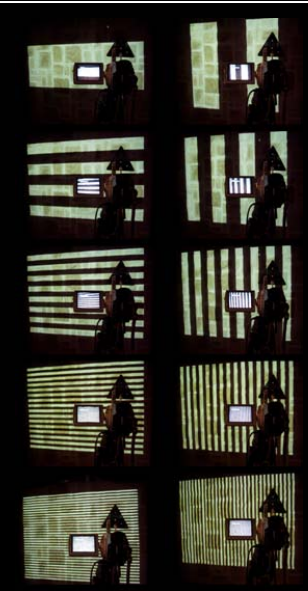
Complex Surfaces

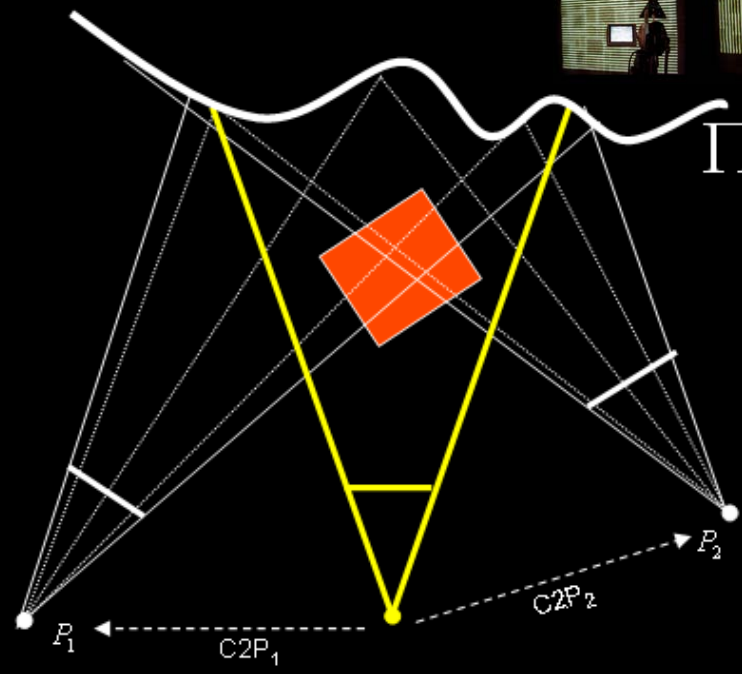


Pixel Displacement Mapping

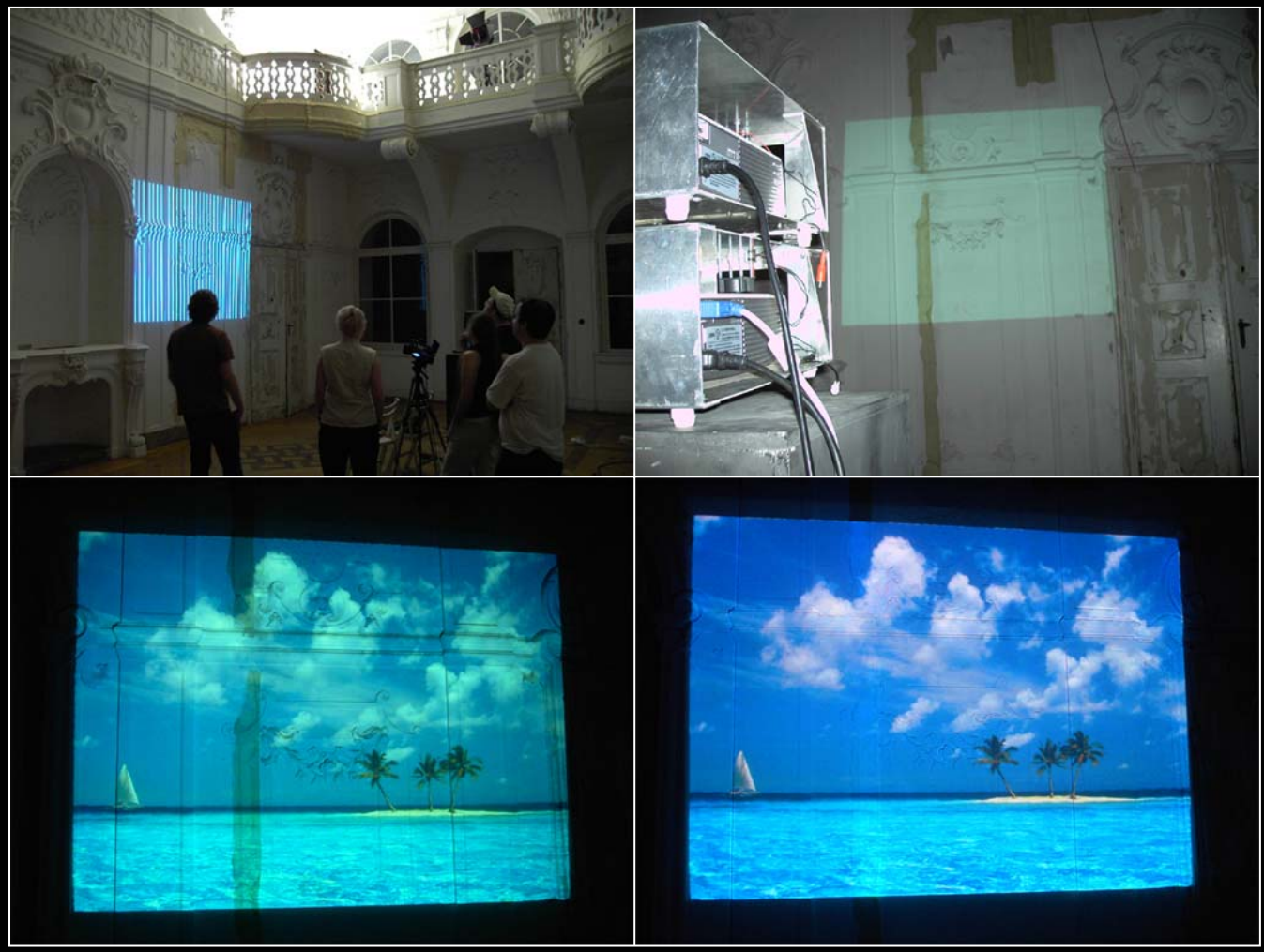
- registering projections to such a surface by determining their intrinsic and extrinsic is too imprecise
  - non-linear lens distortion
  - errors in measuring fiducials
- rendering of 3D surface representation from perspective of projector might be to slow
  - high geometric complexity of model
  - many triangles to render
    - project, raster, texture
- measure per-pixel mapping between projector perspectives and target perspective (e.g., camera)
- render image from target perspective and map it (look-up) into perspective of projectors (e.g., pixel-shading)

problem:
works only for static target perspective!
(but image-based rendering approaches exist)





Example: Stucco Wall

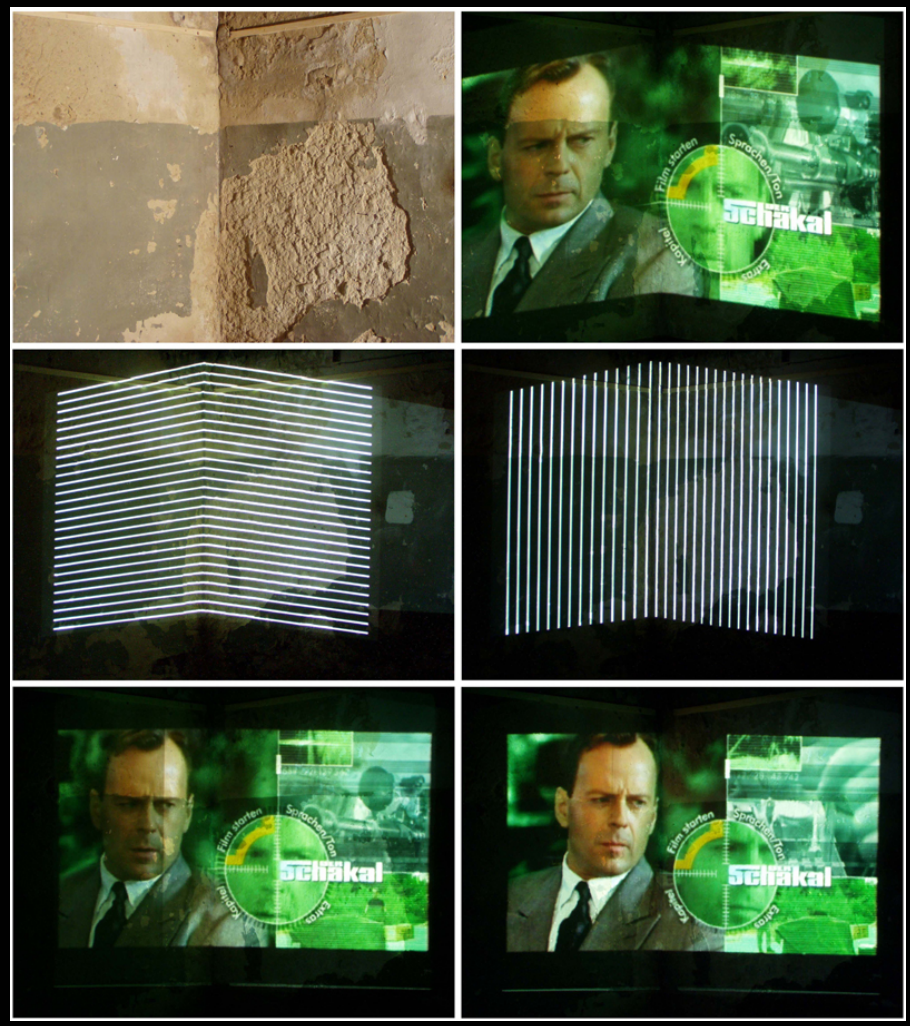


In coop. with castle Etters-burg

Example: Fossil Cast

In coop. with Senckenberg Museum

Example: Scruffy Room Corner

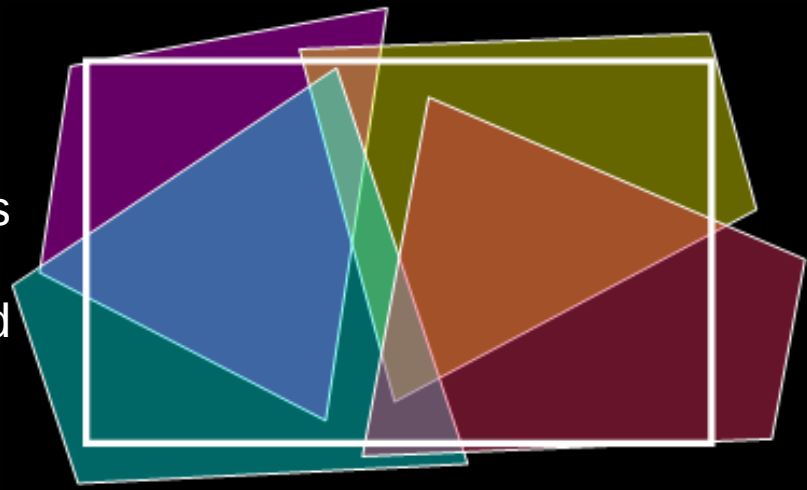


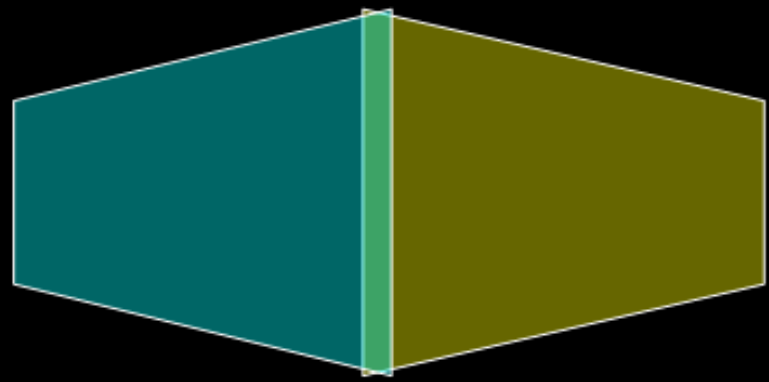
Bimber et al.,
IEEE Computer
2005

Radiometric Compensation

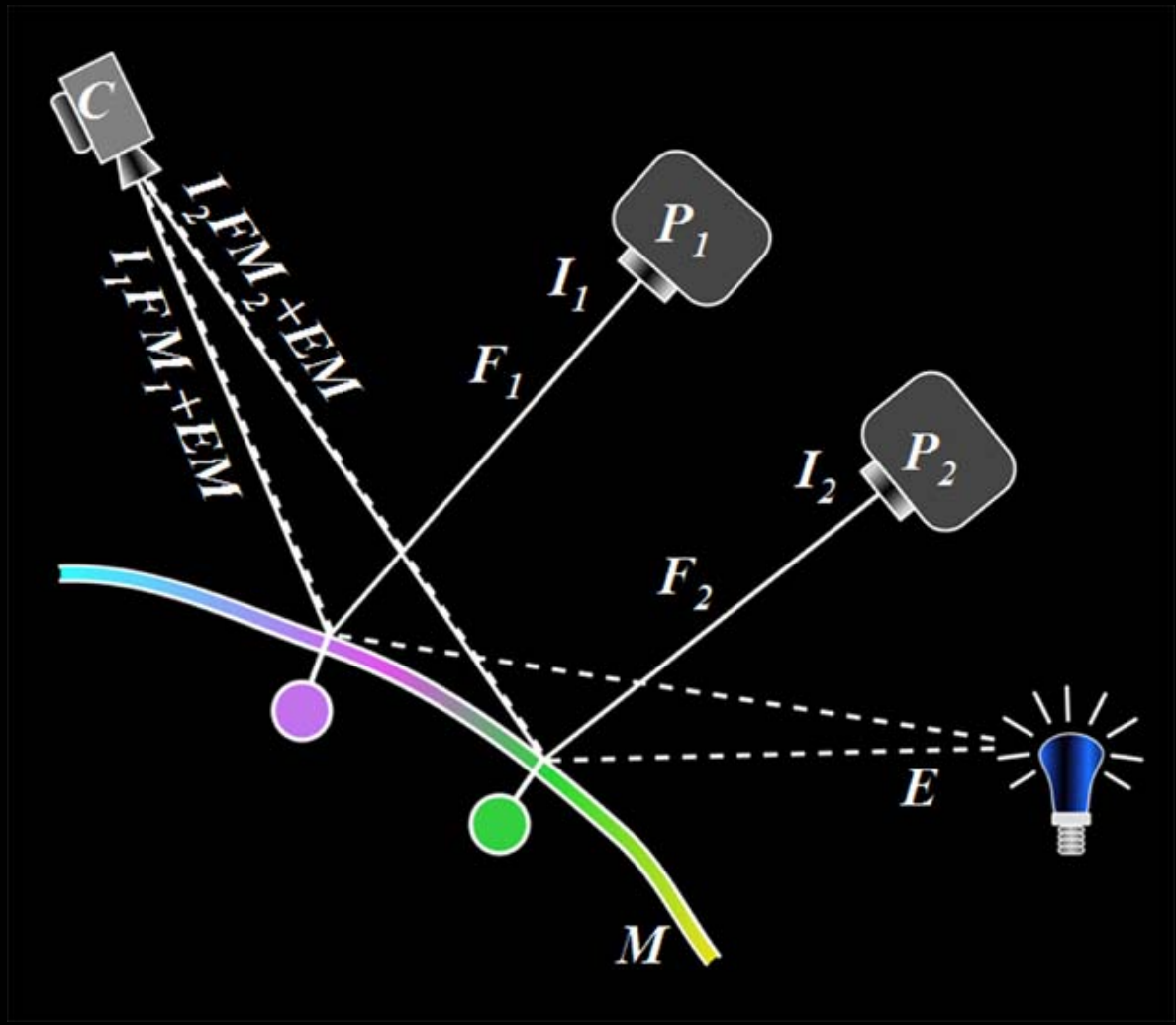
Photometric Calibration

- regions of display surfaces that are illuminated by multiple projectors simultaneously appear brighter
- projectors can have different brightness and can cover a different color space
- result: inconsistent image (intensity and color)
- humans can perceive 2% difference in brightness and a color variation of 2*nm*
- variations in brightness is more critical than variation in color
- solutions: intensity blending and color space mapping
- **these techniques are not explained here!**
- **we assume that projectors and cameras are linearized and color mapped**



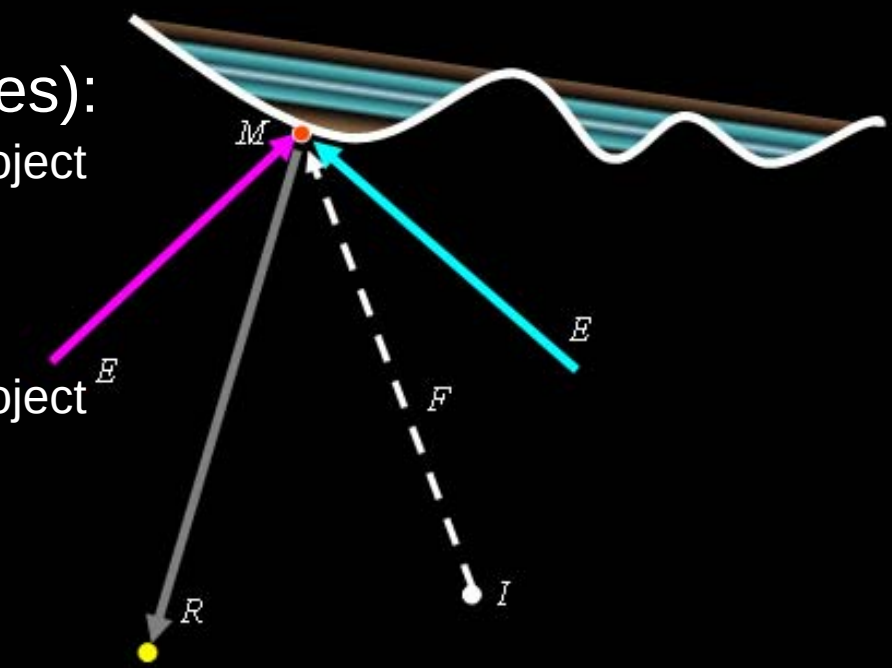


Compensating Local Light Effects



Single Projector

Bimber et al, IEEE Computer 2005



determining parameters (textures):

(1) turn off environment light and project black flood image

I=0,E=0 → BFM

(2) turn on environment light and project black flood image

I=0,E=1 → EM (incl. **BFM !**)

(3) turn off environment light and project white flood image

I=1,E=0 → FM (incl. **BFM !**)

→ FM=FM-BFM

compensation (per pixel):

I=(R-EM)/(FM)

R=IFM+EM

I → projected image
B → black-level
F → projector-2-surface form factor
E → environment light
M → surface reflectance (diffuse)

Color Mixing

Nayar et al, CVPR 2004

determining color mixing matrix V:

for un-normalized matrix: (camera and projector response must be known and linearized):

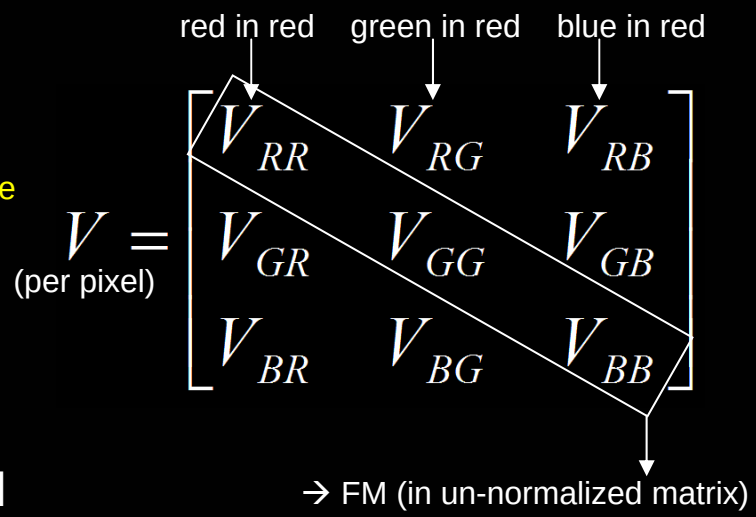
capture 9+ images → least squares

for normalized matrix (camera response must be known, projector response can be unknown):

diagonals are 1 (unknown scaling)

off-diagonals are $\mathbf{V_{ij}=\Delta C_j/\Delta I_i=\Delta C_j/\Delta R_i}$ (since $\mathbf{V_{ii}=1}$, $\mathbf{\Delta I_i=\Delta C_i}$)

capture 6 images **C** (2 per color channel to determine deltas)



compensation (per pixel):

$\mathbf{I=V^{-1}*R}$ (does not consider environment light!)

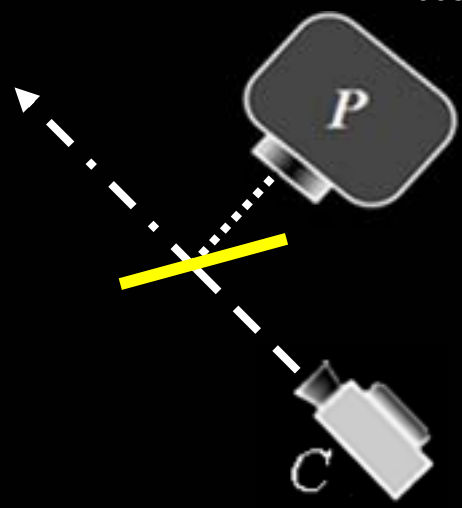
R=V*I

I → projected image

V → color mixing matrix (projector/camera/reflectance)

Dynamic Adaptation

Fujii et al, CVPR 2005



determining color mixing matrix V_0:

similar as before: $\mathbf{V_{ij}=\Delta C_j/\Delta I_i}$

(un-normalized!)

determine reflected environment light E_0*M_0 at t=0:

$\mathbf{E_0*M_0=C-V_0*I}$ (project arbitrary I and capture C)

compensation (per pixel at t):

$\mathbf{I_t=V_0^{-1}*(R*M_0/M_{t-1}-E_{t-1}*M_0)}$

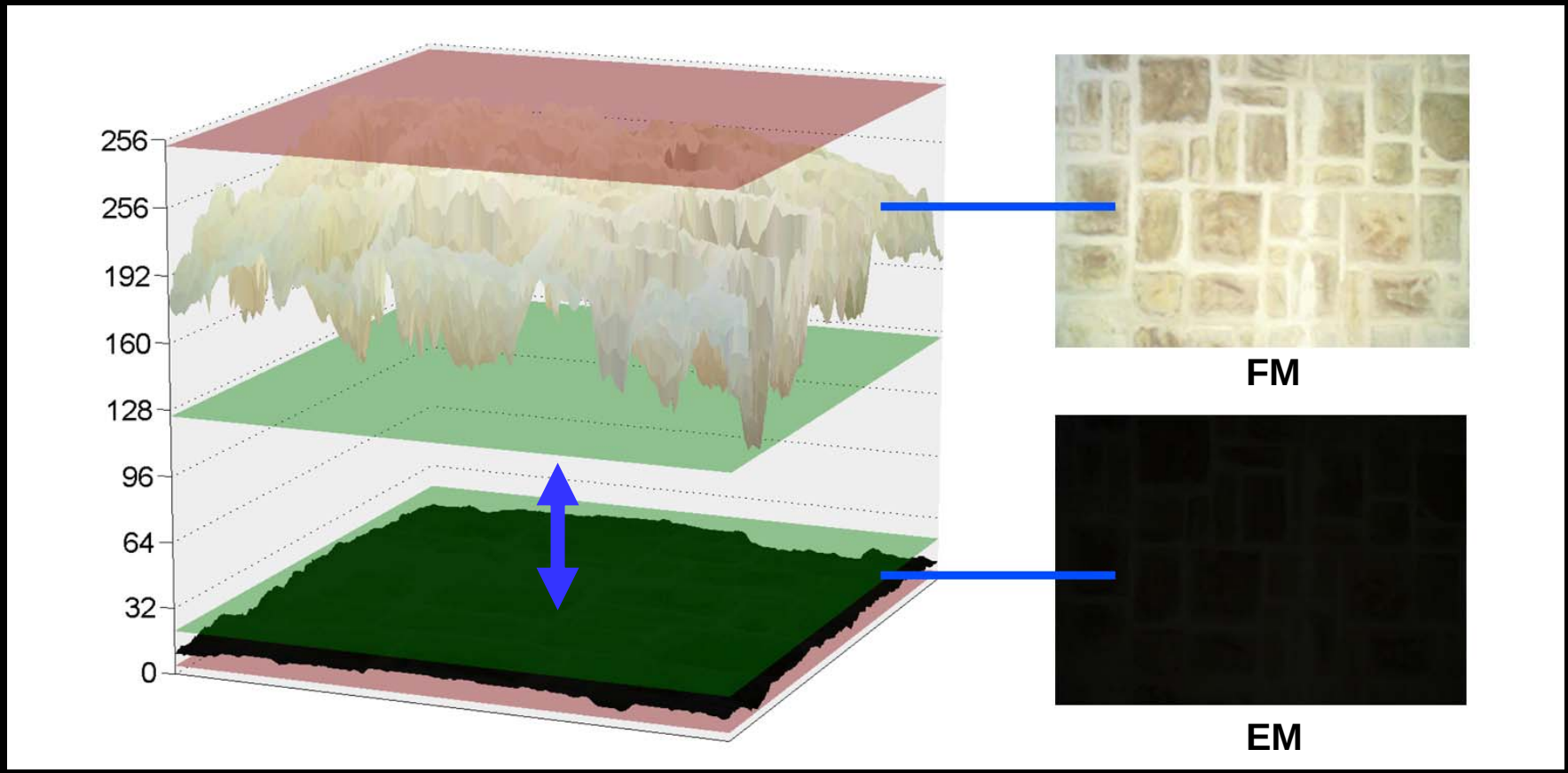
$\mathbf{\rightarrow E_{t-1}*M_0}$ **approx.** $\mathbf{E_0*M_0}$

$\mathbf{\rightarrow M_0/M_{t-1}= C_0/C_{t-1}}$

$\mathbf{R_t=M_t/M_0*(E_t*M_0+V_0*I_t)}$

- t → time index
- I_t → projected image at t
- V_0 → un-normalized color mixing matrix at t=0 (const.)
- M_t → material at t
- M_0 → material at t=0
- E_t → environment light at t=0

Limited Dynamic Range and Brightness



Multiple Projectors

strategy: balance intensity load

- assume: total intensity is equally balanced among multiple low-capacity units

$$I_i = I_1 = I_2 = \ldots = I_N$$

- this is equivalent to the assumption that a single high capacity projector produces the total intensity arriving on the surface virtually

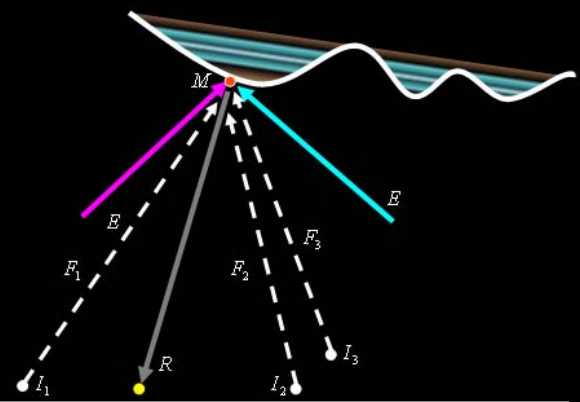
$$R = EM + I_i(F_1M + F_2M + \ldots + F_NM)$$
$$\rightarrow EM + I(F_1 + F_2 + \ldots F_N)M$$

compensation (per pixel):

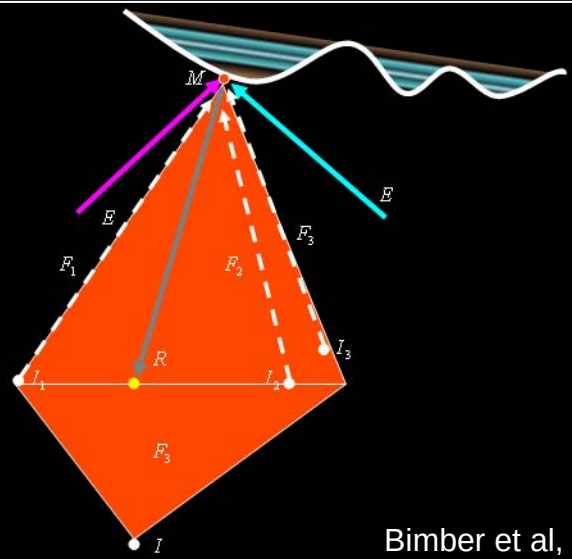
$$I_i = (R-EM)/(F_1M + F_2M + \ldots + F_NM)$$

remember: $F_iM = F_iM - B_iF_iM$!

or $BFM = B_1F_1M + \ldots + B_iF_iM$



$$R = EM + I_1F_1M + I_2F_2M + \ldots + I_NF_NM$$



Bimber et al, IEEE Computer 2005

Considering Human Visual Perception

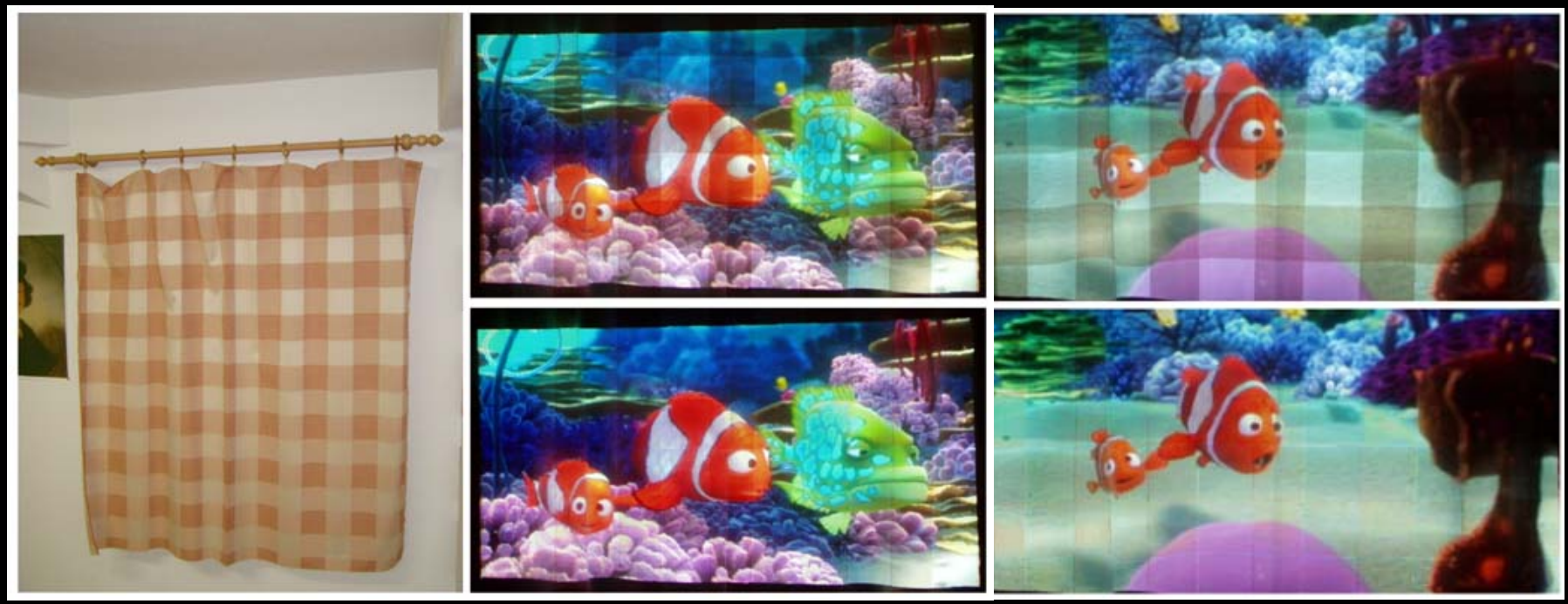
threshold map (Ramasubramanian et al. Siggraph'99)

- computes for every pixel of an image **R** the amount of luminance difference that is imperceptible
- considers ***contrast***, ***luminance*** and ***spatial frequency*** in local neighborhood

adaptation of un-compensated (original) image R:

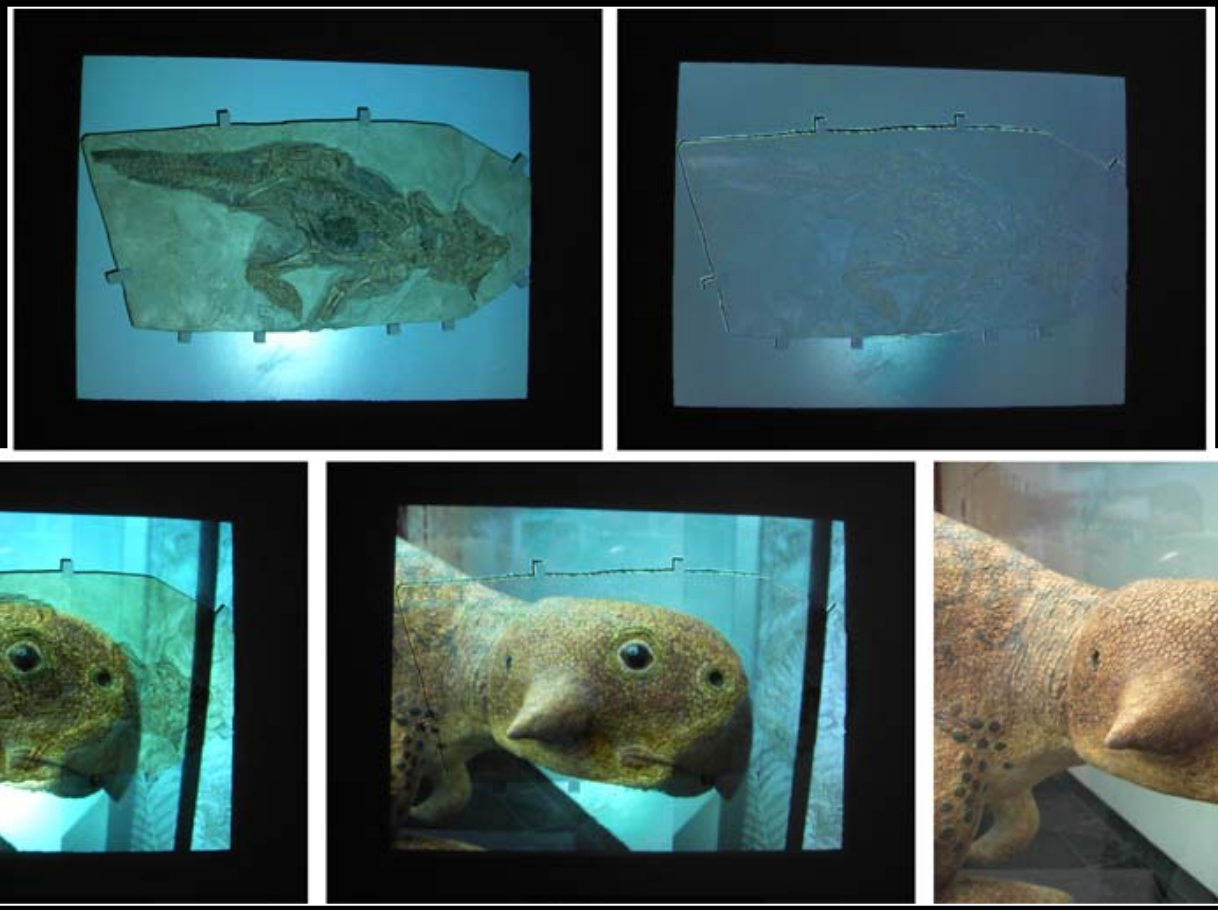
- compute and apply a single (global) scaling factor **R'=R*α** that minimizes the perceived error (Wang, et al. 2005, only monochrome, not real-time, single projector)
- coming soon: color, real-time, global and local adaptation, potentially multiple projectors

Example: Curtain



Bimber et al, IEEE Computer 2005

Example: Fossil



In coop. with Senckenberg Museum

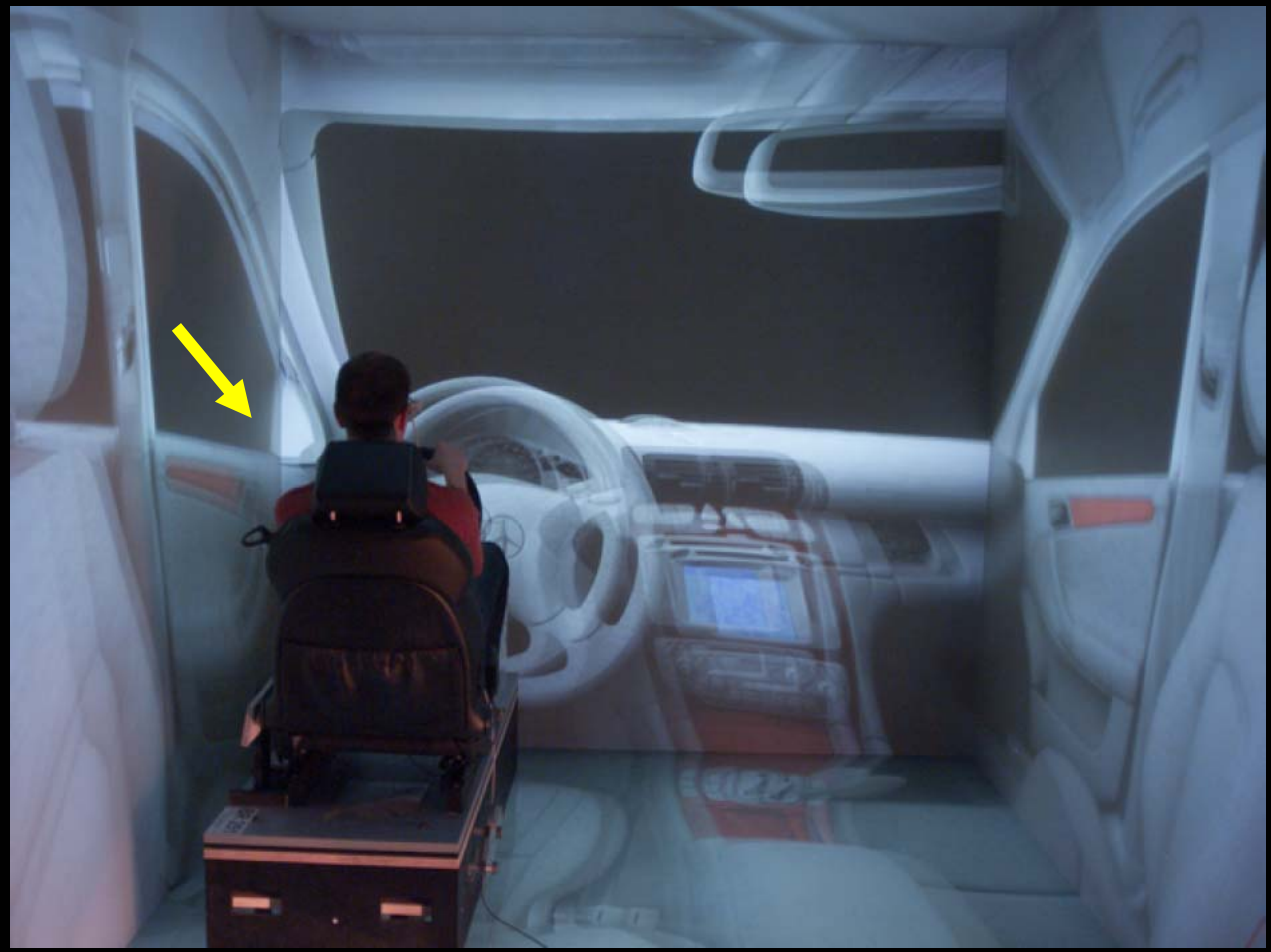
Example: Natural Stone Wall

Bimber et al, IEEE Computer 2005
In coop. with Bennert Group

Example: Wallpaper

Bimber et al, IEEE Computer 2005

Compensating Interreflections



Dmitriev, et al.
A CAVE system for interactive modeling of global illumination in car interior, VRST'04

physical illumination effects that occur on the ***display's surfaces*** have also to be taken into account!

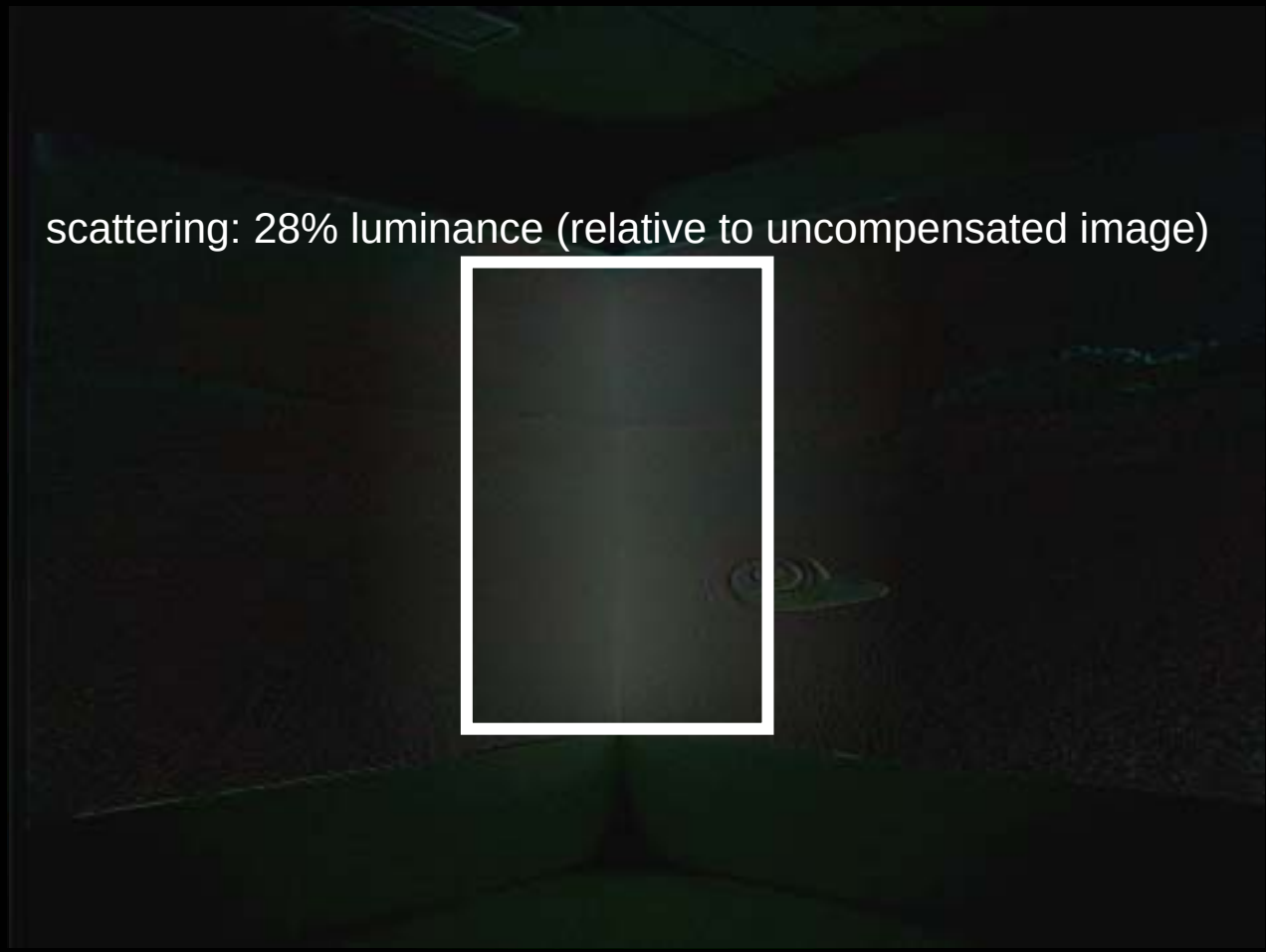
Compensating Interreflections

note: imaging and visualization limitations of LDR photographs and presentation projector

Compensating Interreflections

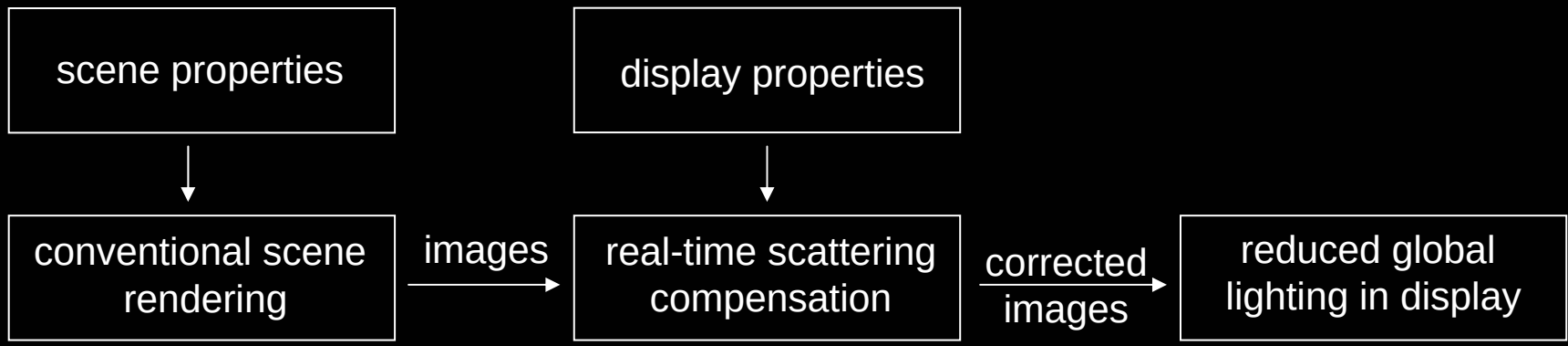
note: imaging and visualization limitations of LDR photographs and presentation projector

Compensating Interreflections



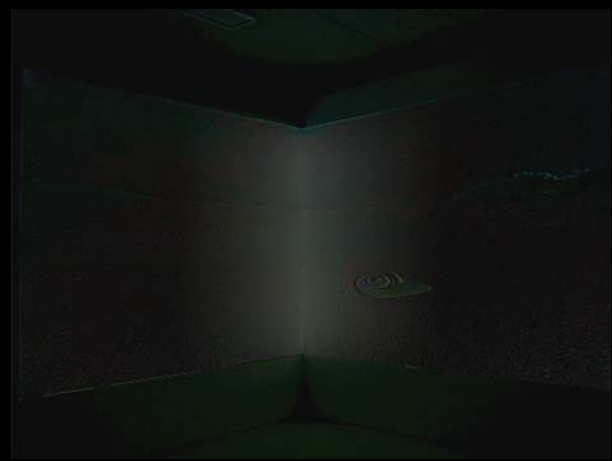
note: imaging and visualization limitations of LDR photographs and presentation projector

Compensating Interreflections



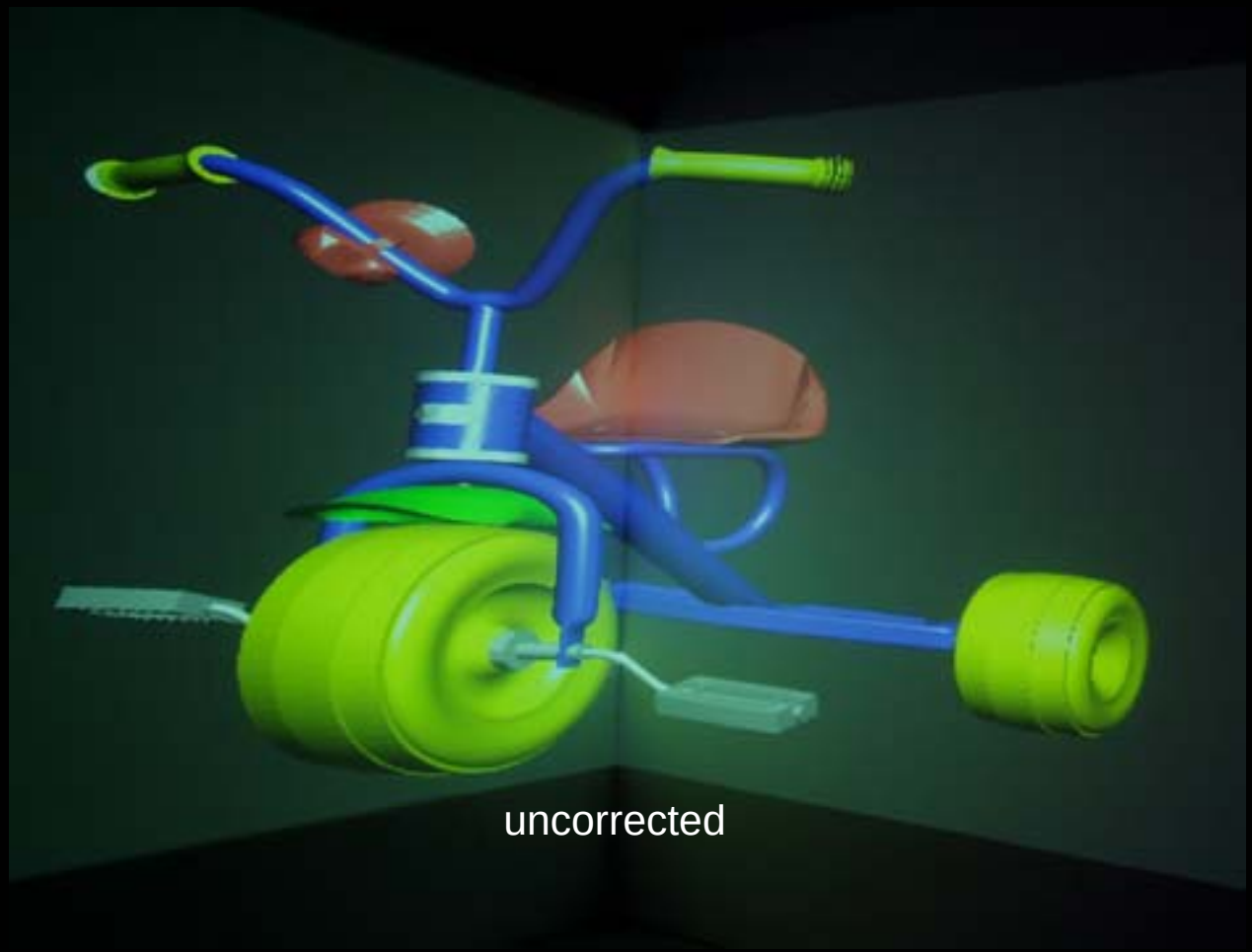
regular projection

corrected projection



compensated scattering

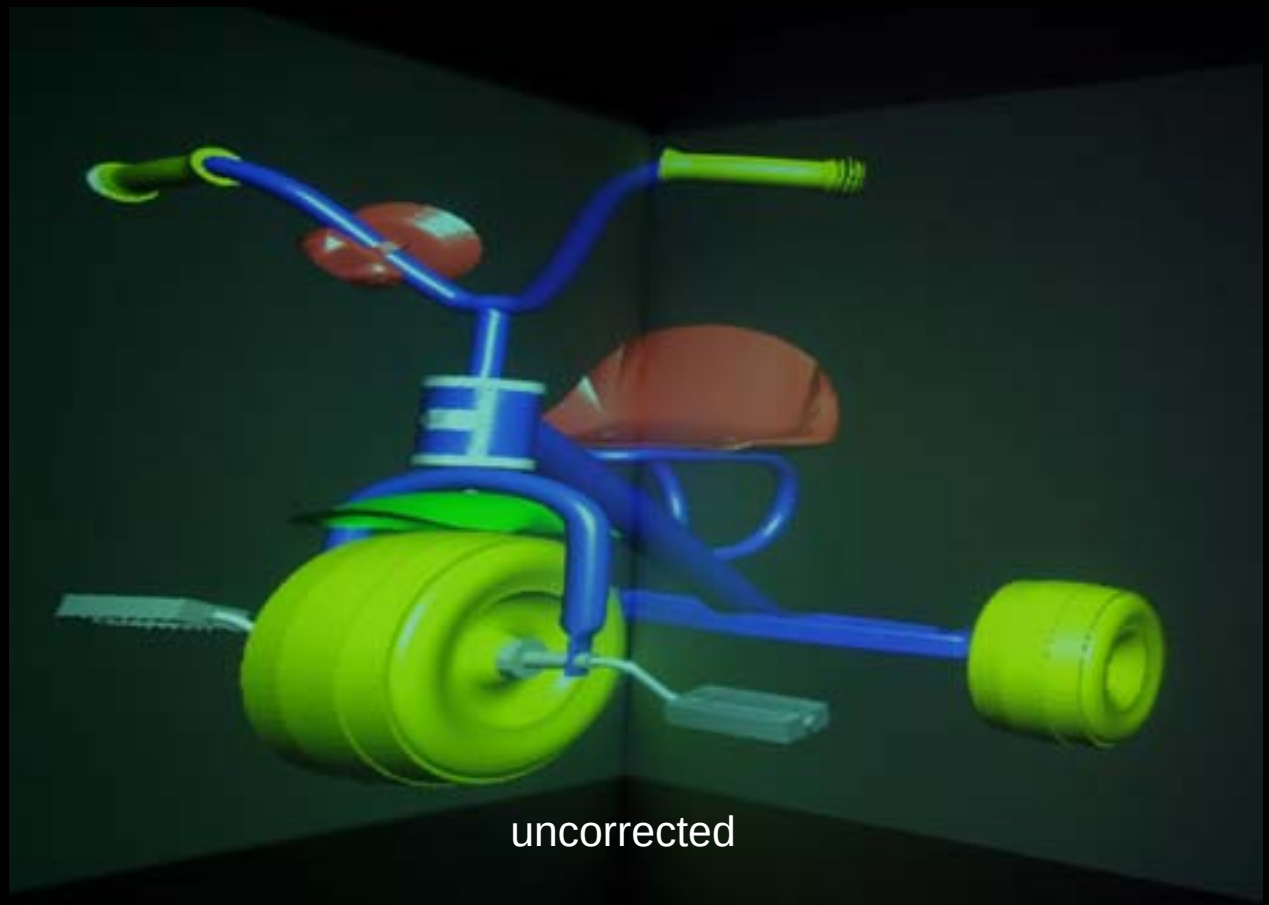
Compensating Interreflections



uncorrected

note: imaging and visualization limitations of LDR photographs and presentation projector

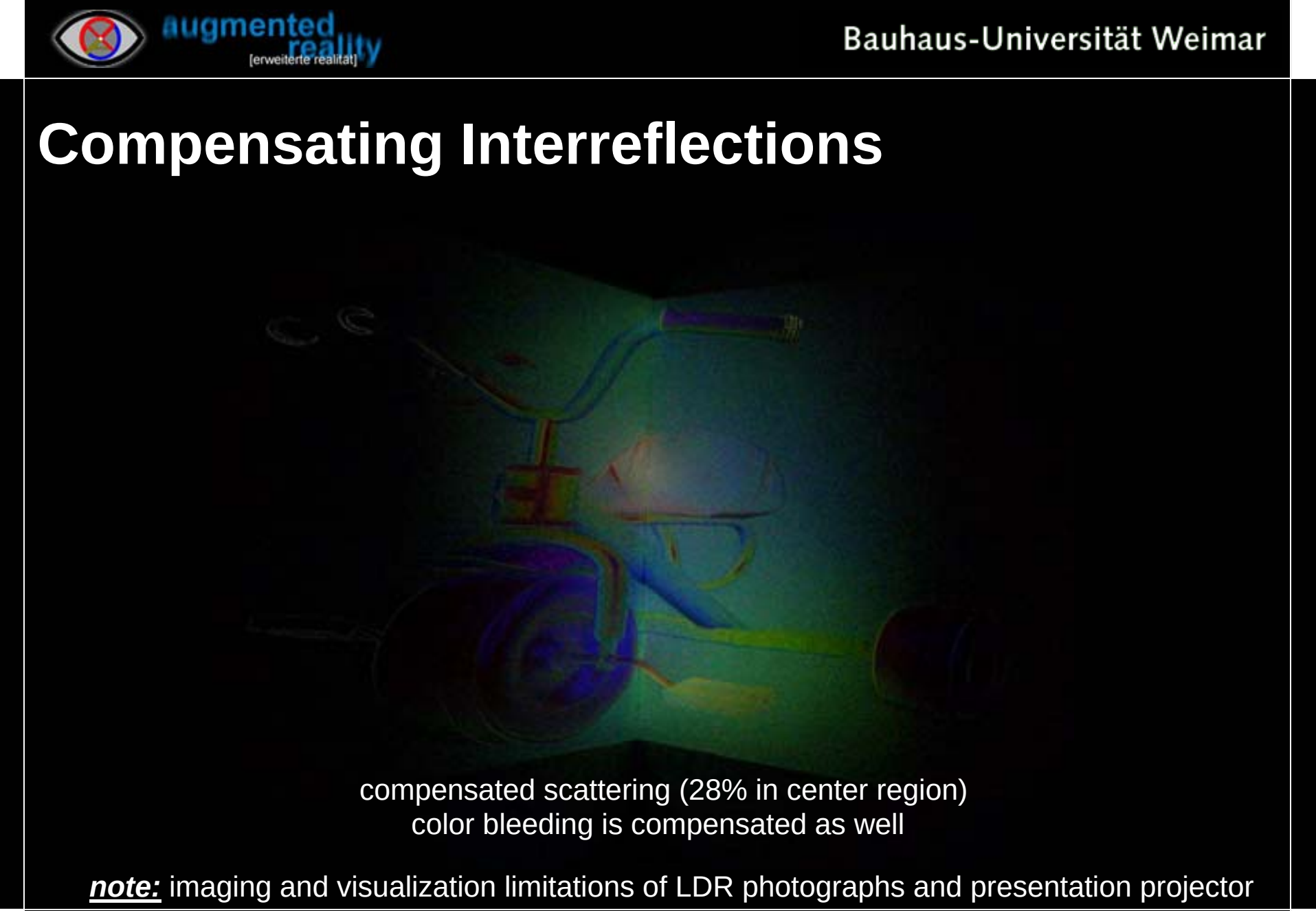
Compensating Interreflections



uncorrected

note: imaging and visualization limitations of LDR photographs and presentation projector

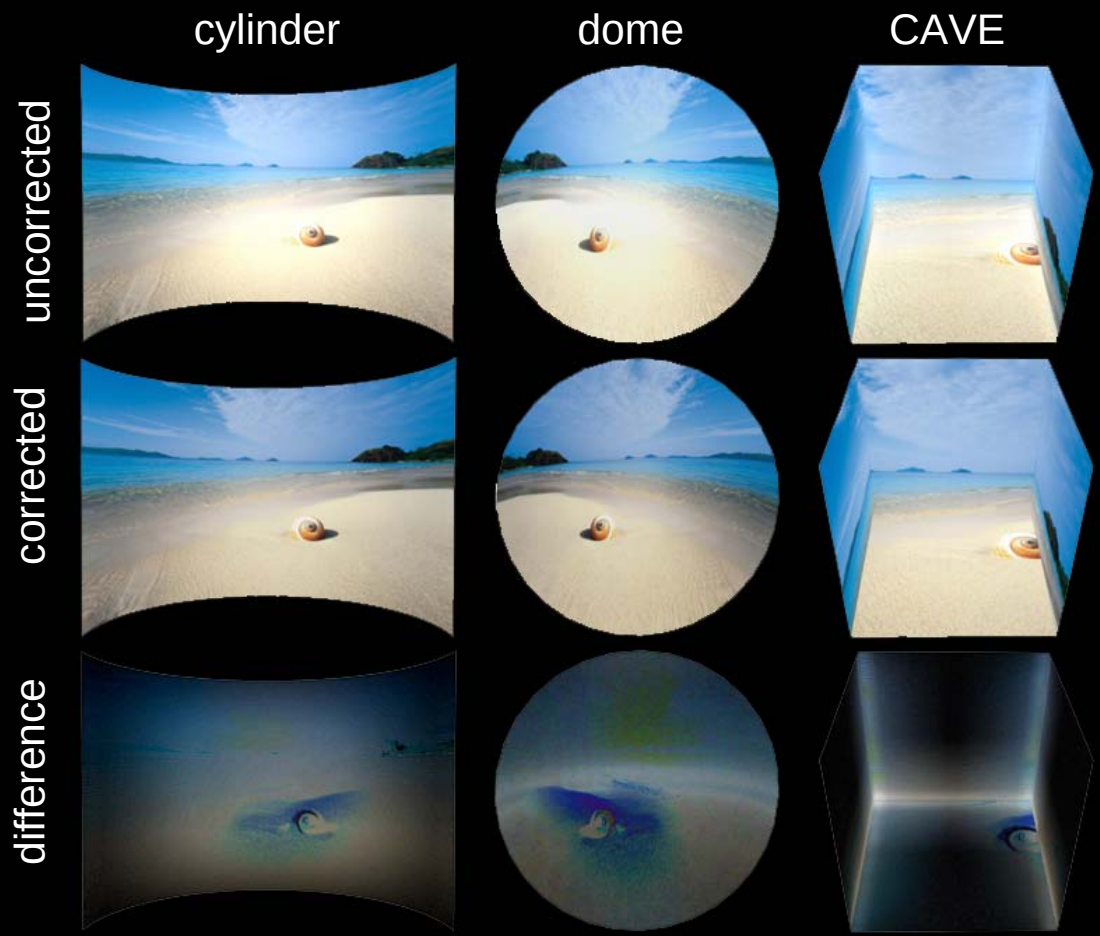
Compensating Interreflections



compensated scattering (28% in center region)
color bleeding is compensated as well

note: imaging and visualization limitations of LDR photographs and presentation projector

Compensating Interreflections



note: this is a simulation (difference is contrast enhanced)

Related Work

Image correction for (stereo) projection displays

- lots of work on ***multi-projector calibration*** (geometry, photometry, and blending): van Baar, Majumder, Yang, Raskar, and many more
- ***cross-talk reduction*** for stereo displays: Lipscomp et al. 95, Konrad et al. '00 (active), Klimenko '03 (passive)
- ***photometric mapping*** for Infitec driven CAVE and wall: Kresse et al. '03
- ***high-dynamic range visualizations*** with foveal adaptation in CAVEs: Dmitriv et al. '04
- ***geometric, radiometric and focus correction*** for stereoscopic projection onto arbitrary surfaces: Bimber et al. '05/'06

Radiosity on the GPU

- ***matrix radiosity via Jacobi iteration***: Carr et al. '03
- ***progressive refinement radiosity***: Coombe et al. '04

Inverse Global Illumination and Inverse Light Transport

- ***inverse global illumination*** to recover reflectance: Yu, Debevec et al. '99
- ***shape from interreflections***: Nayar, et al. '91
- ***removing interreflections in photographs***: Seitz et al. '05

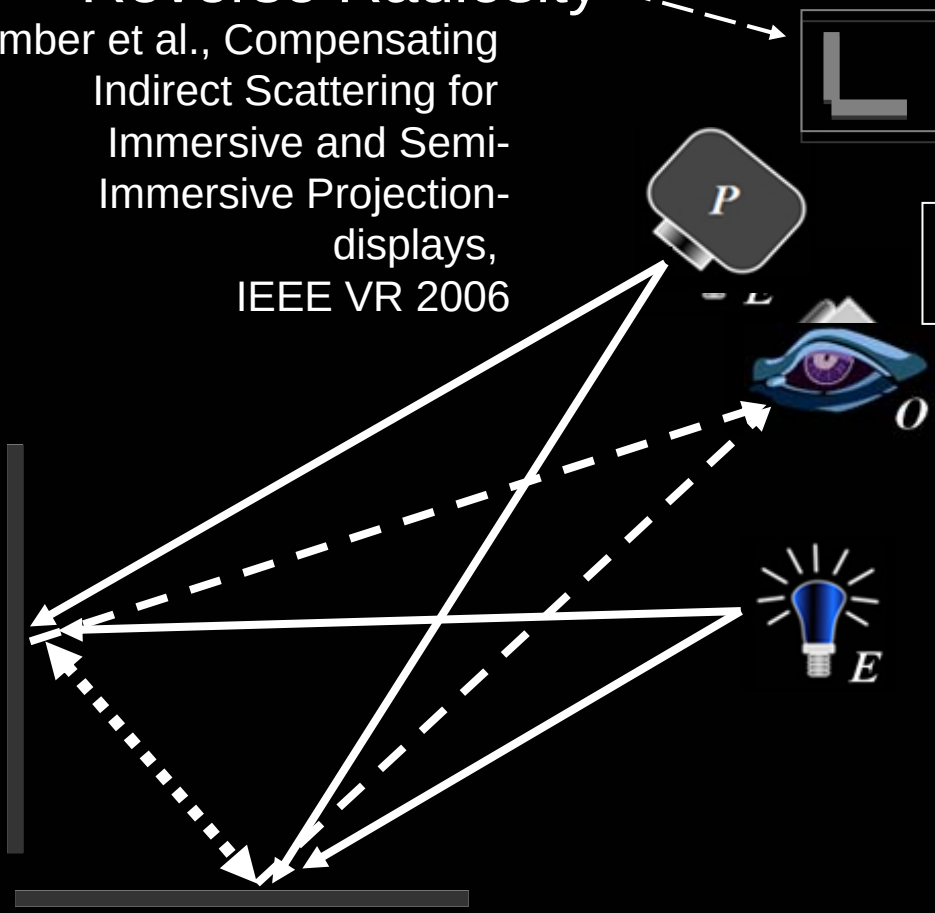
The Reverse Radiosity Problem

Reverse Radiosity

Bimber et al., Compensating Indirect Scattering for Immersive and Semi-Immersive Projection-displays, IEEE VR 2006

Radiosity

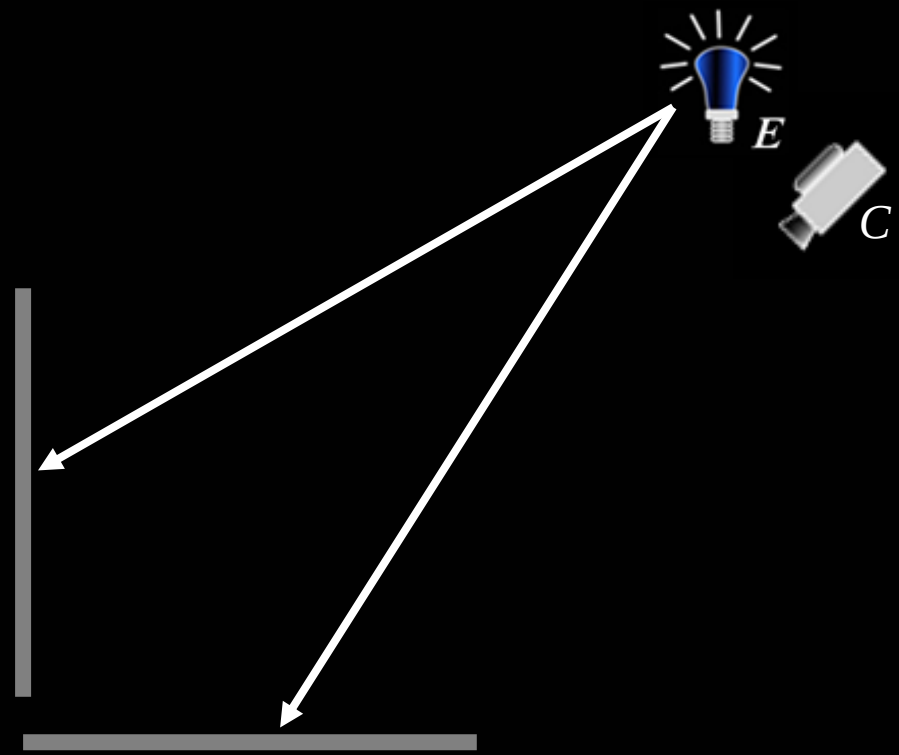
$$B_i = E_i + M_i \sum_{j, j \neq i}^{P} {}_i B_j F_{ji}$$



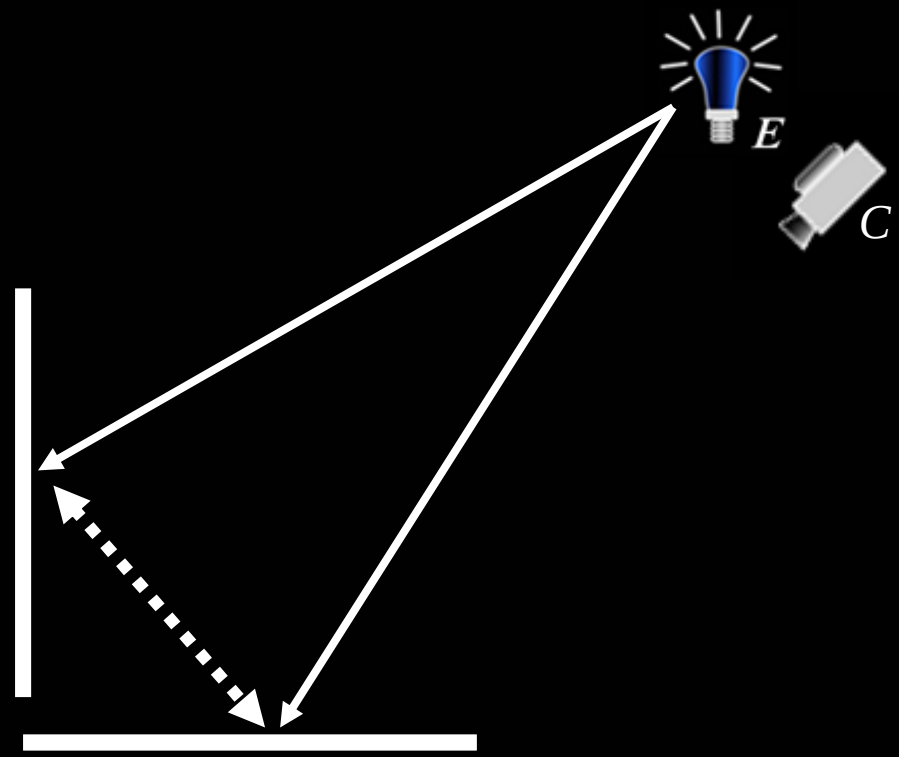
Inverse Light Transport

Seitz et al., A Theory of Inverse Light Transport (removing interreflections in photographs), ICCV, October 2005

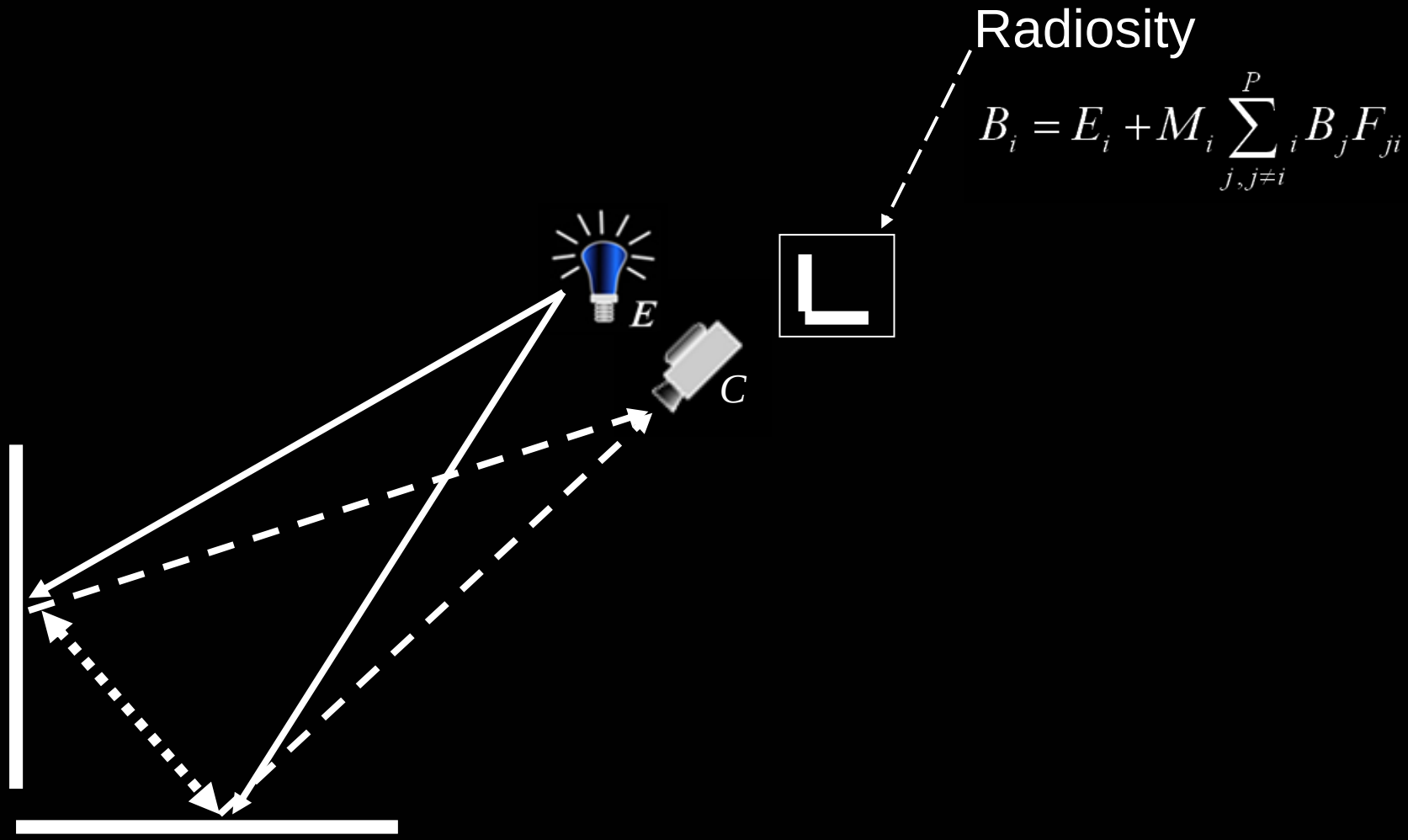
The Reverse Radiosity Problem



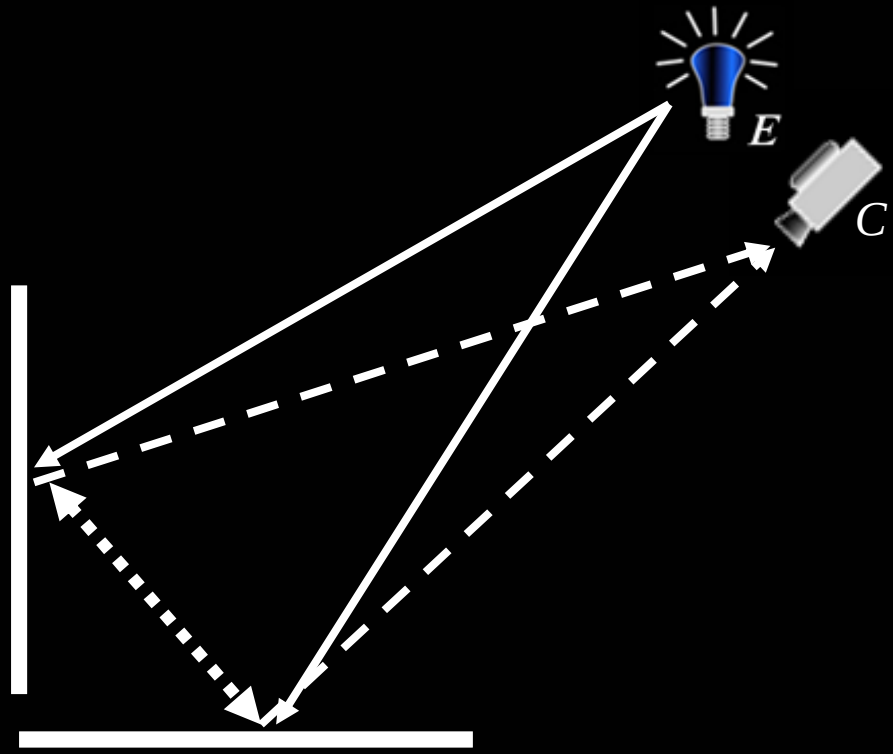
The Reverse Radiosity Problem



The Reverse Radiosity Problem



The Reverse Radiosity Problem



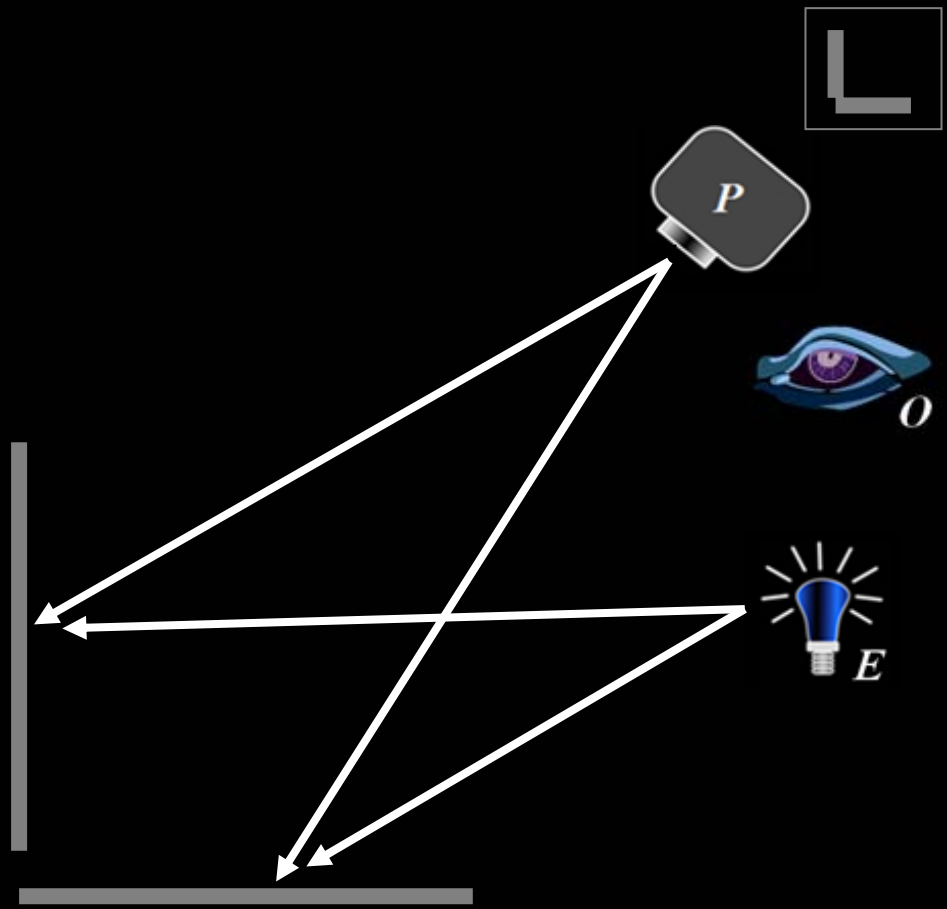
Radiosity

$$B_i = E_i + M_i \sum_{j, j \neq i}^{P} {}_i B_j F_{ji}$$

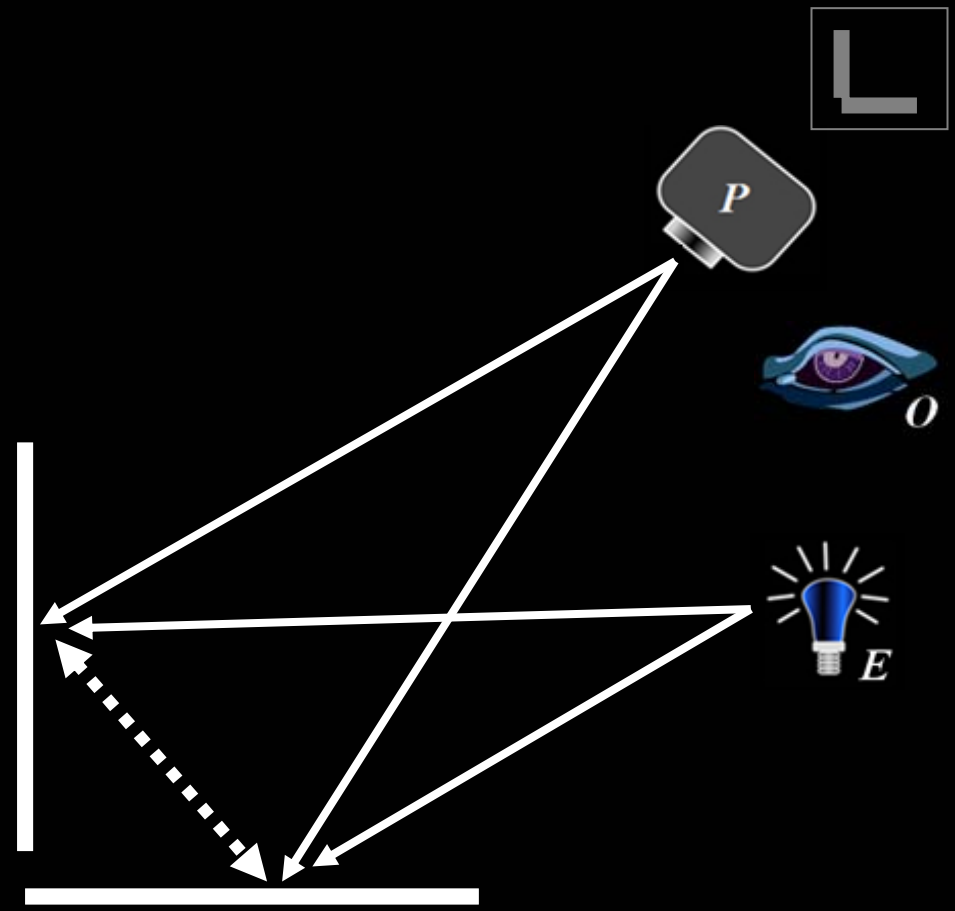
Inverse Light Transport

Seitz et al., A Theory of Inverse Light Transport (removing interreflections in photographs), ICCV, October 2005

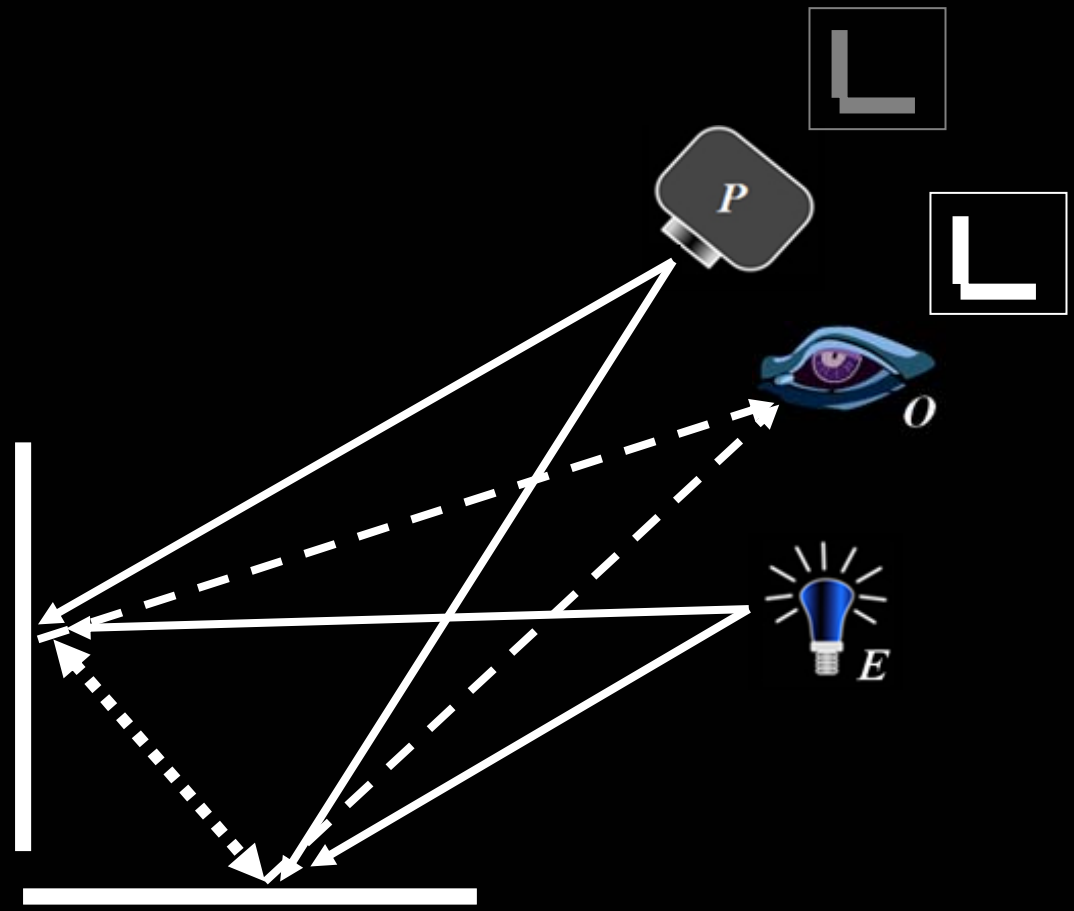
The Reverse Radiosity Problem



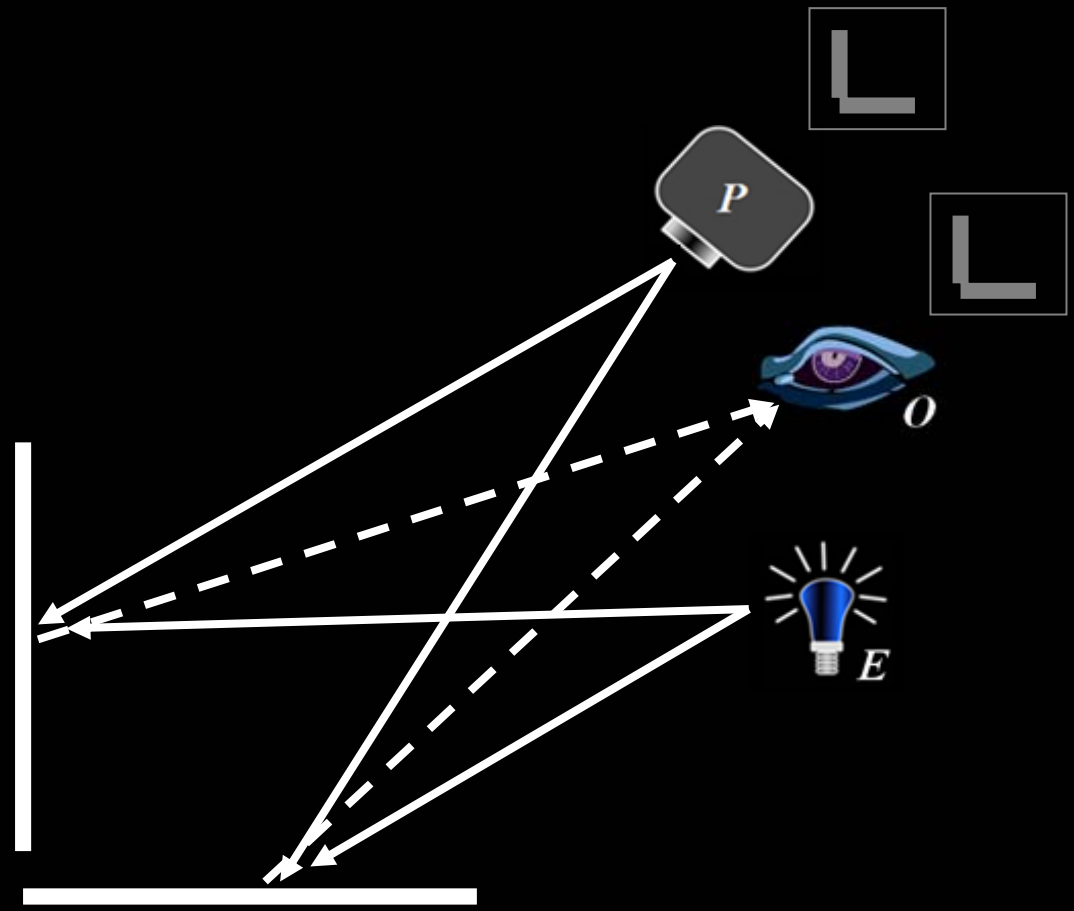
The Reverse Radiosity Problem



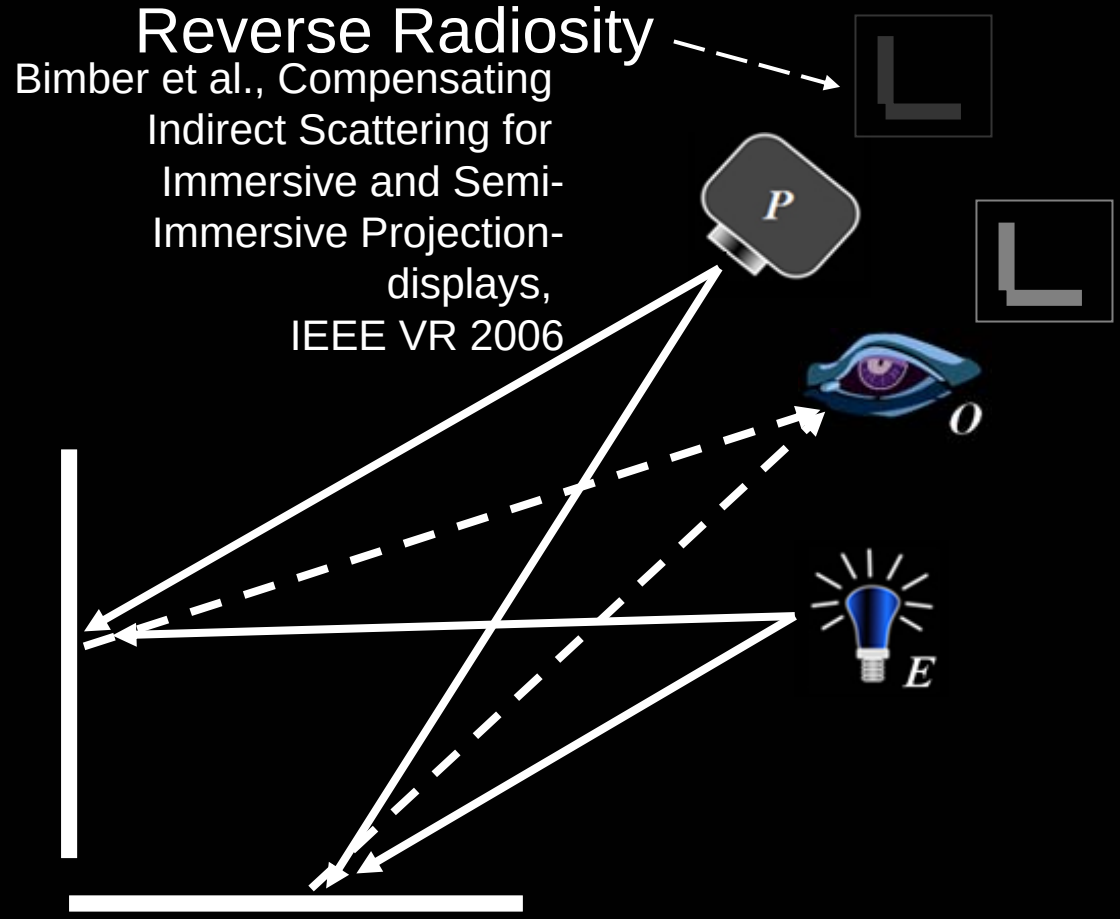
The Reverse Radiosity Problem



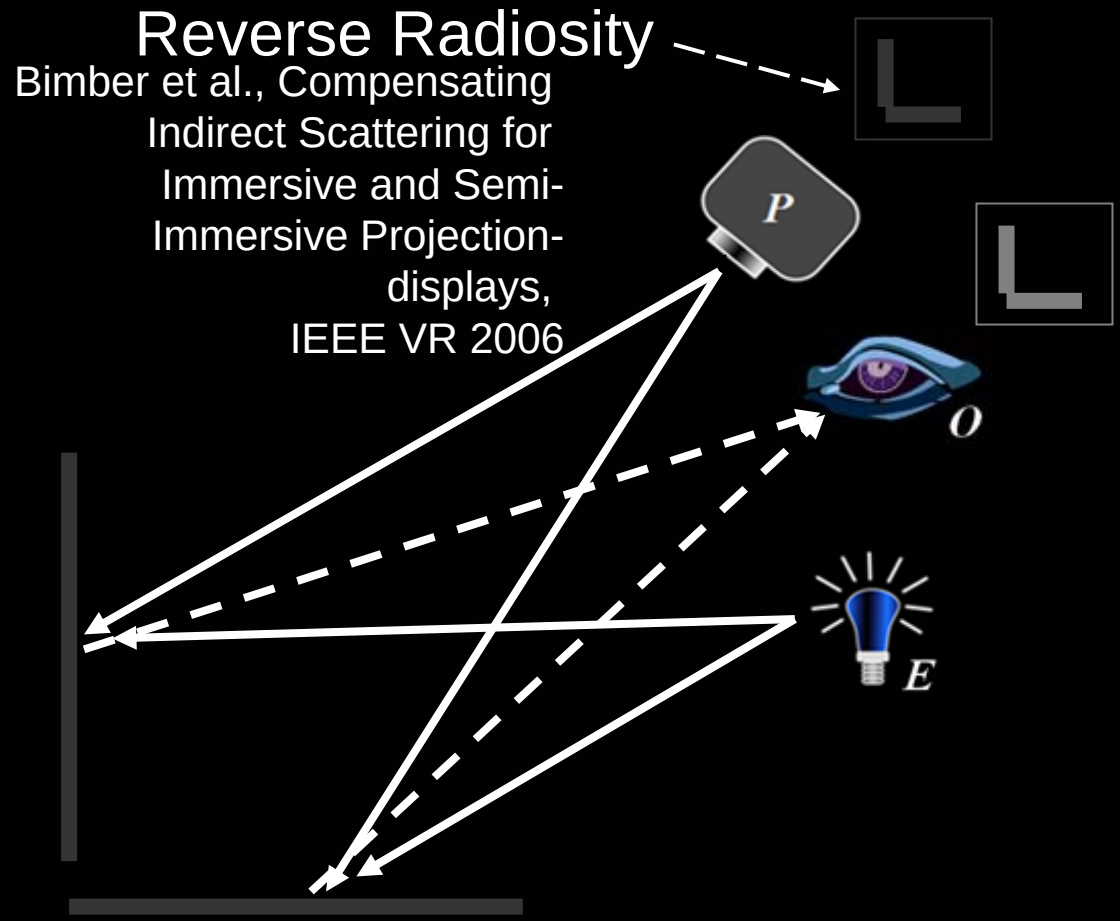
The Reverse Radiosity Problem



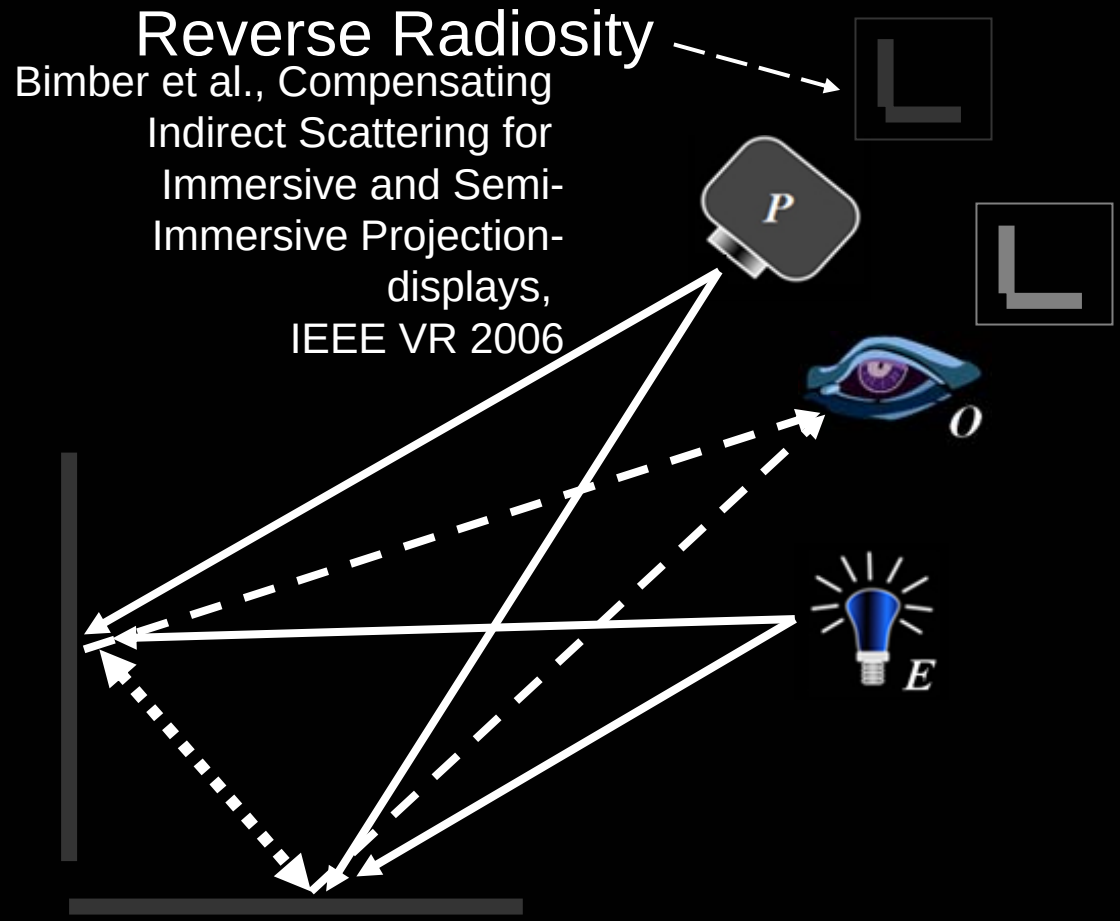
The Reverse Radiosity Problem



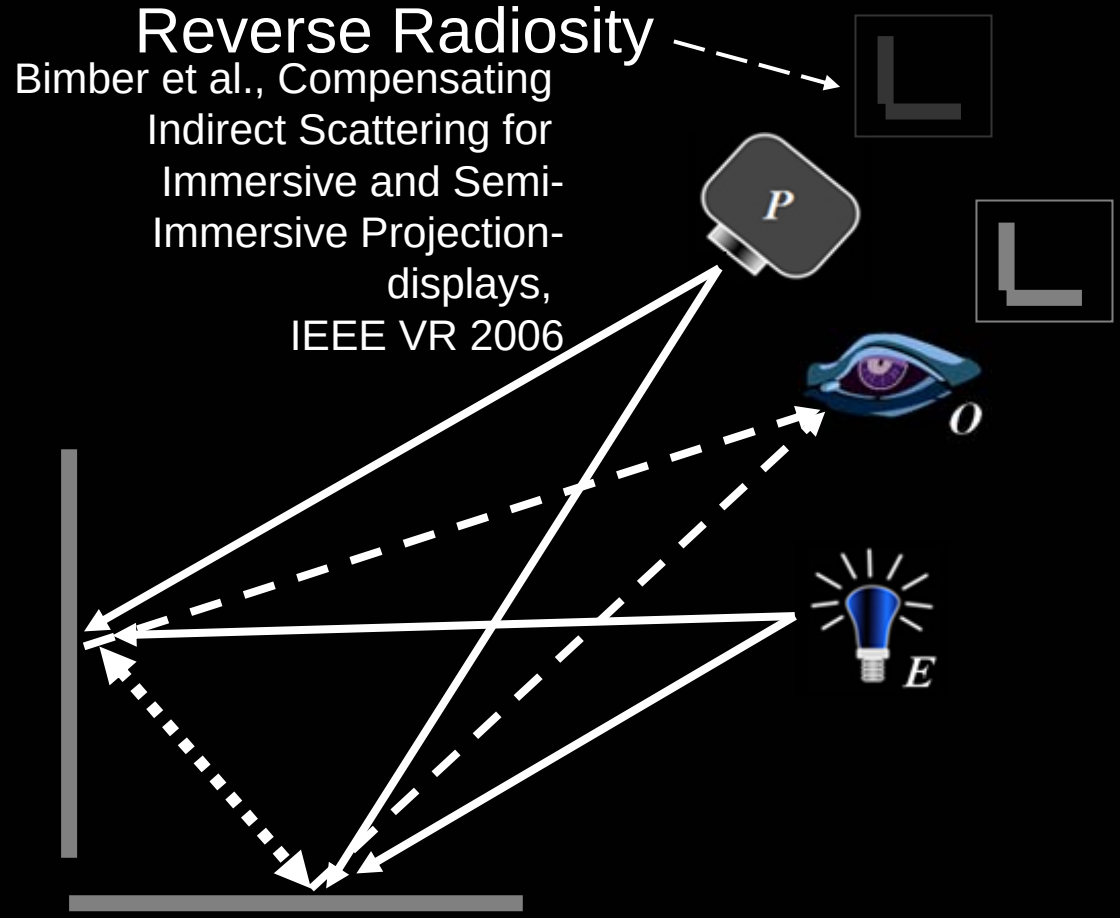
The Reverse Radiosity Problem



The Reverse Radiosity Problem



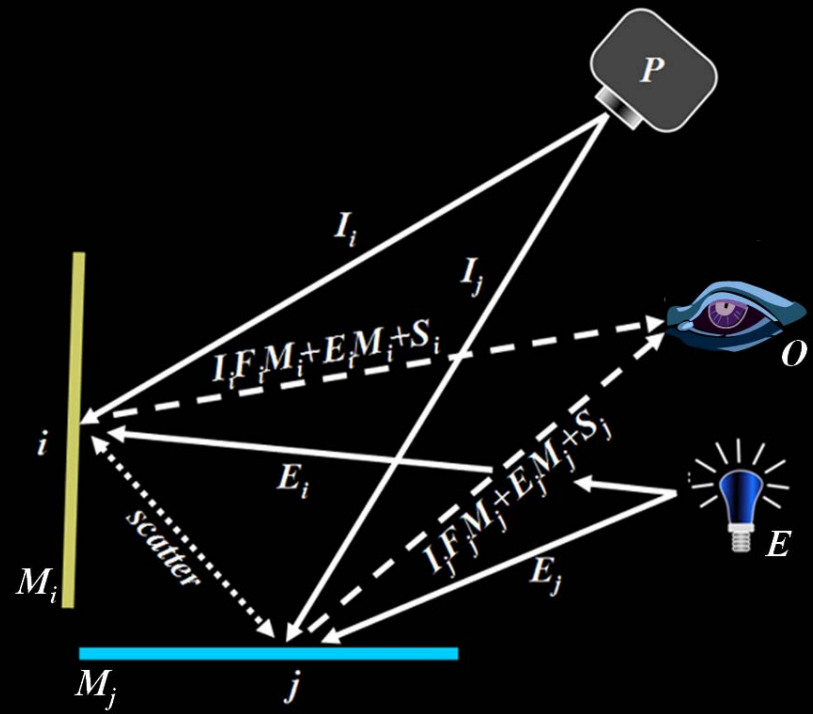
The Reverse Radiosity Problem



Numerical Solution

assumption: higher scatter levels do not contribute a lot

our numerical solution: compute scattering level for level and compensate in between (consider only the first *H* scatter levels)



Scattering (1st level):

$$S_i = \sum_{j,j\neq i}^{P} \left(I_j F_j M_j + E_j M_j\right) F_{ji} M_i$$

Compensated 1st level scattering:

$$I_i^{l+1} = \frac{1}{F_i M_i}\left(R - E_i M_i - S_i^1\right)$$

note: same applies for back projection

F_j: projector to patch (j) form factor
I: projected image
R: original image
E: environment light at patch
M: reflectance of patch
S: gathered scattering at patch
F_{ji}: patch to patch (j to i) form factor
i,j: patch indices
l: iteration
h: scatter level (H max)

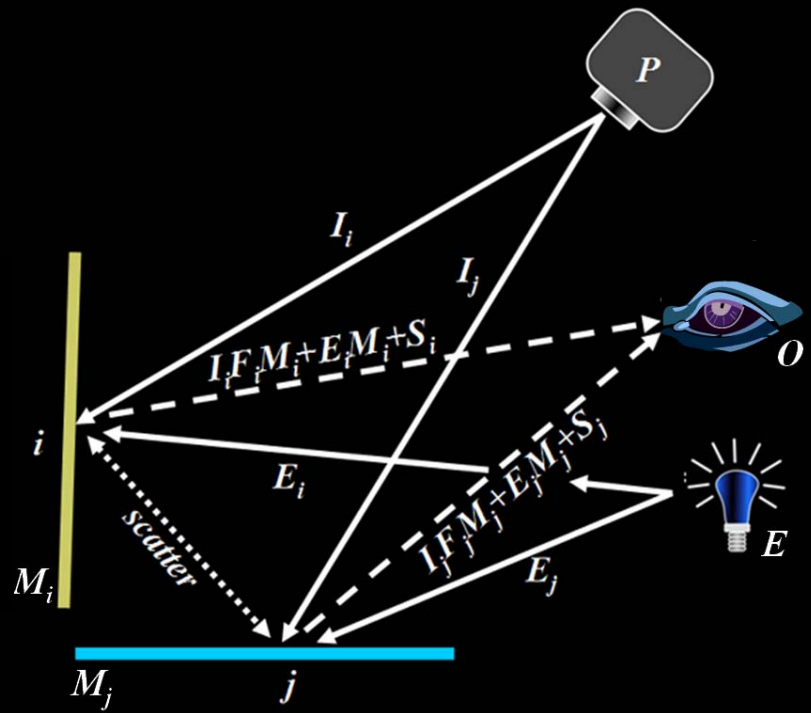
Numerical Solution

Scattering (h^{th} level):

$$S_i = \sum_{j,j\neq i}^{P} \left(I_j F_j M_j + E_j M_j\right) F_{ji} M_i +$$

$$\sum_{j,j\neq i}^{P} \sum_{k,k\neq j}^{P} \left[\left(I_k F_k M_k + E_k M_k\right) F_{kj} M_j\right] F_{ji} M_i + \ldots$$

Recursive: $$S_i^h = \sum_{j,j\neq i}^{P} S_j^{h-1} F_{ji} M_i$$



note: same applies for back projection

Compensated h^{th} level scattering:

$$I_i^{l+1} = \frac{1}{F_i M_i}\left(R - E_i M_i - \sum_{h=1}^{H} S_i^h\right)$$

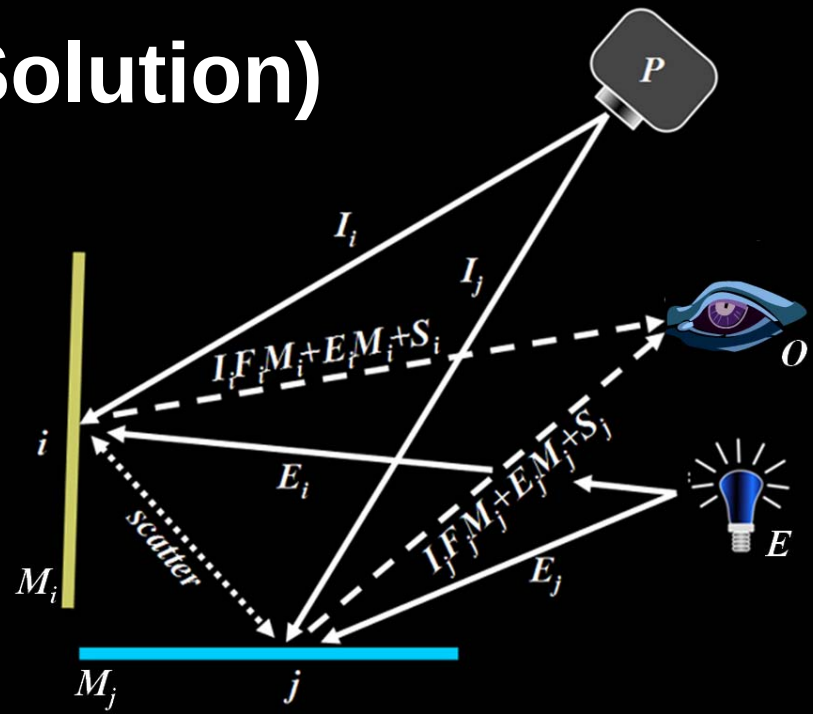
F_j: projector to patch (j) form factor
I: projected image
R: original image
E: environment light at patch
M: reflectance of patch
S: gathered scattering at patch
F_{ji}: patch to patch (j to i) form factor
i,j: patch indices
l: iteration
h: scatter level (H max)

Algorithms (Numerical Solution)

1st scatter level only:

```
compute I^0 //direct illumination
l=0 //number of passes
repeat
        compute S^l (for I^l)
        compute I^(l+1) with S^l
        l=l+1
until exit condition is fulfilled
display I^(l+1)
```

note: yellow parts are implemented as fragment shaders, equivalent to ***Jacobi iterations***



h^{th} scatter level:

```
compute I^0 //direct illumination
l=0 //number of passes
repeat
        compute S^(1,l) (for I^l)
        for h=2 to H //scatter level
                compute S^(h,l) from S^(h-1,l)
        endfor
        compute I^(l+1) with S^(1..H,l)
        l=l+1 //number of passes
until exit condition is fulfilled
display I^(l+1)
```

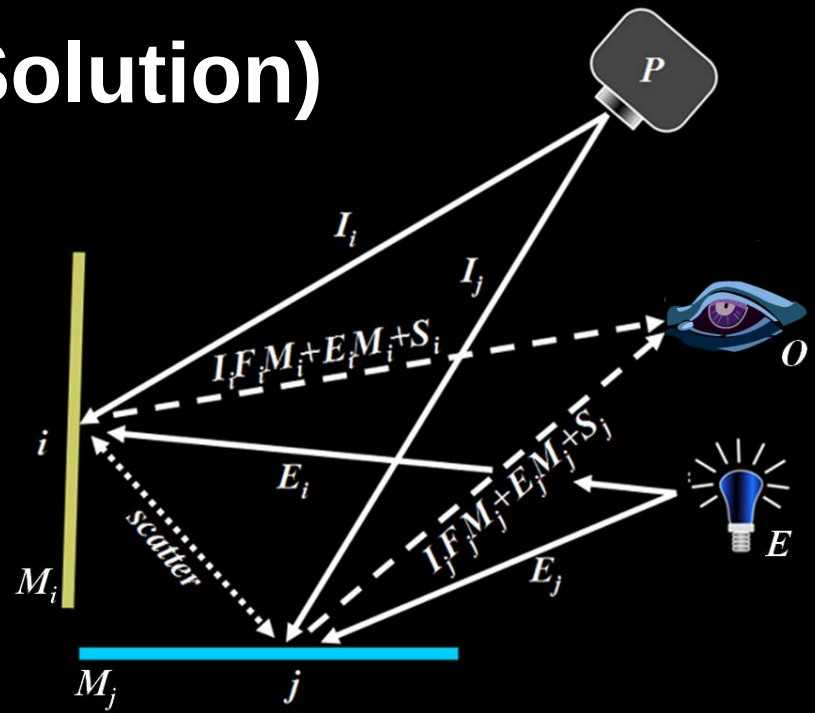
note: same applies for back projection

F_j: projector to patch (j) form factor
I: projected image
R: original image
E: environment light at patch
M: reflectance of patch
S: gathered scattering at patch
F_{ji}: patch to patch (j to i) form factor
i,j: patch indices
l: iteration
h: scatter level (H max)

Algorithms (Numerical Solution)

Exit condition 1:

- ***constant quality***
  - results from two consecutive iterations do not differ much
  - GPU implementation: ***occlusion queries***
    - fragment shader discards corresponding pixels whose r,g,b difference is below a ***pixel discrepancy threshold***.
    - occlusion query delivers number of pixel which are written (pixels with differences above threshold)
    - if this pixel count is below ***image discrepancy threshold***



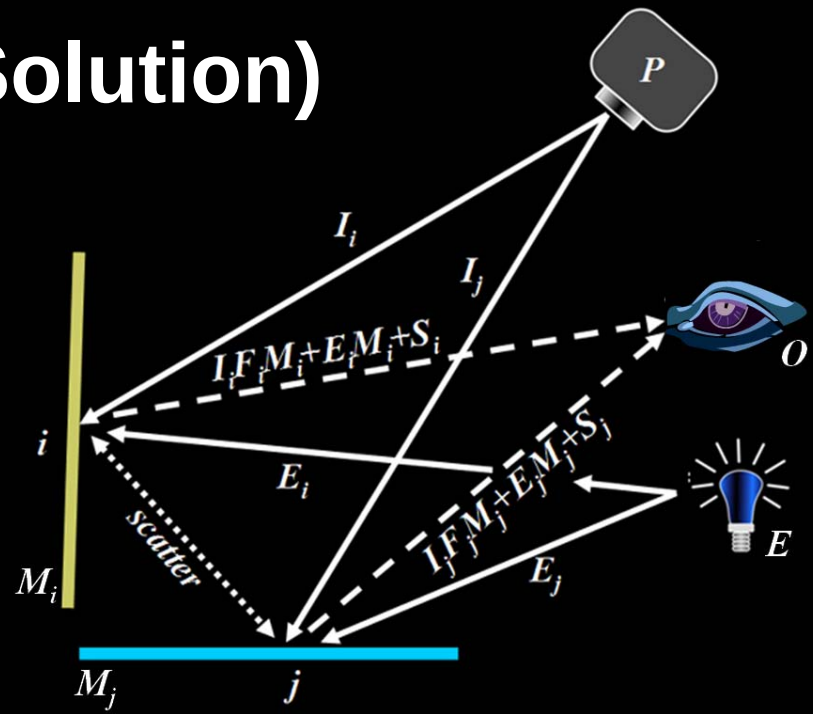
note: same applies for back projection

F_j: projector to patch (j) form factor
I: projected image
R: original image
E: environment light at patch
M: reflectance of patch
S: gathered scattering at patch
F_{ji}: patch to patch (j to i) form factor
i,j: patch indices
l: iteration
h: scatter level (H max)

Algorithms (Numerical Solution)

Exit condition 2:

- ***constant solving rate***
  - measure the time for each render pass individually and subtract it from the time slot until it is exhausted



note: same applies for back projection

stop solving if either one of the two exit conditions is true

F_j: projector to patch (j) form factor
I: projected image
R: original image
E: environment light at patch
M: reflectance of patch
S: gathered scattering at patch
F_{ji}: patch to patch (j to i) form factor
i,j: patch indices
l: iteration
h: scatter level (H max)

Analytical Solution

technical report, February 2006:
http://e-pub.uni-weimar.de/frontdoor.php?source_opus=773

$$S_i^h = \sum_{j,\, j\neq i}^{P} S_j^{h-1} F_{ji} M_i \qquad S_i^1 = \sum_{j,\, j\neq i}^{P} \left(I_j F_j M_j + E_j M_j \right) F_{ji} M_i \rightarrow S_i^1 = \sum_{j,\, j\neq i}^{P} C_j F_{ji} M_i$$

$$R_i = C_i + \overbrace{\sum_{j,\, j\neq i}^{P} C_j F_{ji} M_i}^{h=1} + \overbrace{\sum_{j,\, j\neq i}^{P} \sum_{k,\, k\neq j}^{P} \left[C_k F_{kj} M_j \right] F_{ji} M_i}^{h=2} + \overbrace{\sum_{j,\, j\neq i}^{P} \sum_{k,\, k\neq j}^{P} \sum_{l,\, l\neq k}^{P} \left(\left[C_l F_{lk} M_k \right] F_{kj} M_j \right) F_{ji} M_i}^{h=3} + \ldots$$

$$R_i = C_i \left(1 + \underbrace{\sum_{j,\, j\neq i}^{P} F_{ji} M_i}_{A} + \sum_{j,\, j\neq i}^{P} \sum_{k,\, k\neq j}^{P} F_{kj} M_j F_{ji} M_i + \sum_{j,\, j\neq i}^{P} \sum_{k,\, k\neq j}^{P} \sum_{l,\, l\neq k}^{P} F_{lk} M_k F_{kj} M_j F_{ji} M_i + \ldots \right)$$

Seitz's interreflection cancellation operator **(but applied to image without interreflections!)**

$$C = MF \cdot R$$

$$R_i = C_i \left(1 + A + A^2 + A^3 + \ldots \right) \xrightarrow{\lim(h \rightarrow \infty)} R_i = C_i \left(\frac{1}{1 - A} \right) \rightarrow C_i = R_i - R_i A \rightarrow C_i = R_i - \sum_{j,\, j\neq i}^{P} R_j F_{ji} M_i$$

$$I_i = \frac{\left(R_i - \sum_{j,\, j\neq i}^{P} R_j F_{ji} M_i \right) - E_i M_i}{F_i M_i}$$

the amount of indirect ***scattering*** that is produced by the ***compensated image throughout all levels*** **equals** the amount of indirect ***scattering*** that is produced by the ***original image in the first level***!

Interactive Rendering

- ***down-sample*** image into discrete patch resolution *P*
- compensate in patch resolution
- ***up-sample*** result into original image resolution *Q* and ***avoid resolution artefacts***:

$$I_Q = I_Q^0 - \uparrow\left(\downarrow I_Q^0 - I_P\right)$$

- filter out entries of form-factor matrix that are below predefined threshold (thus form factor matrix F_{ji} is a ***sparse matrix***!)
- compensate combined stereo pairs in a single step (little difference in down sampled versions)

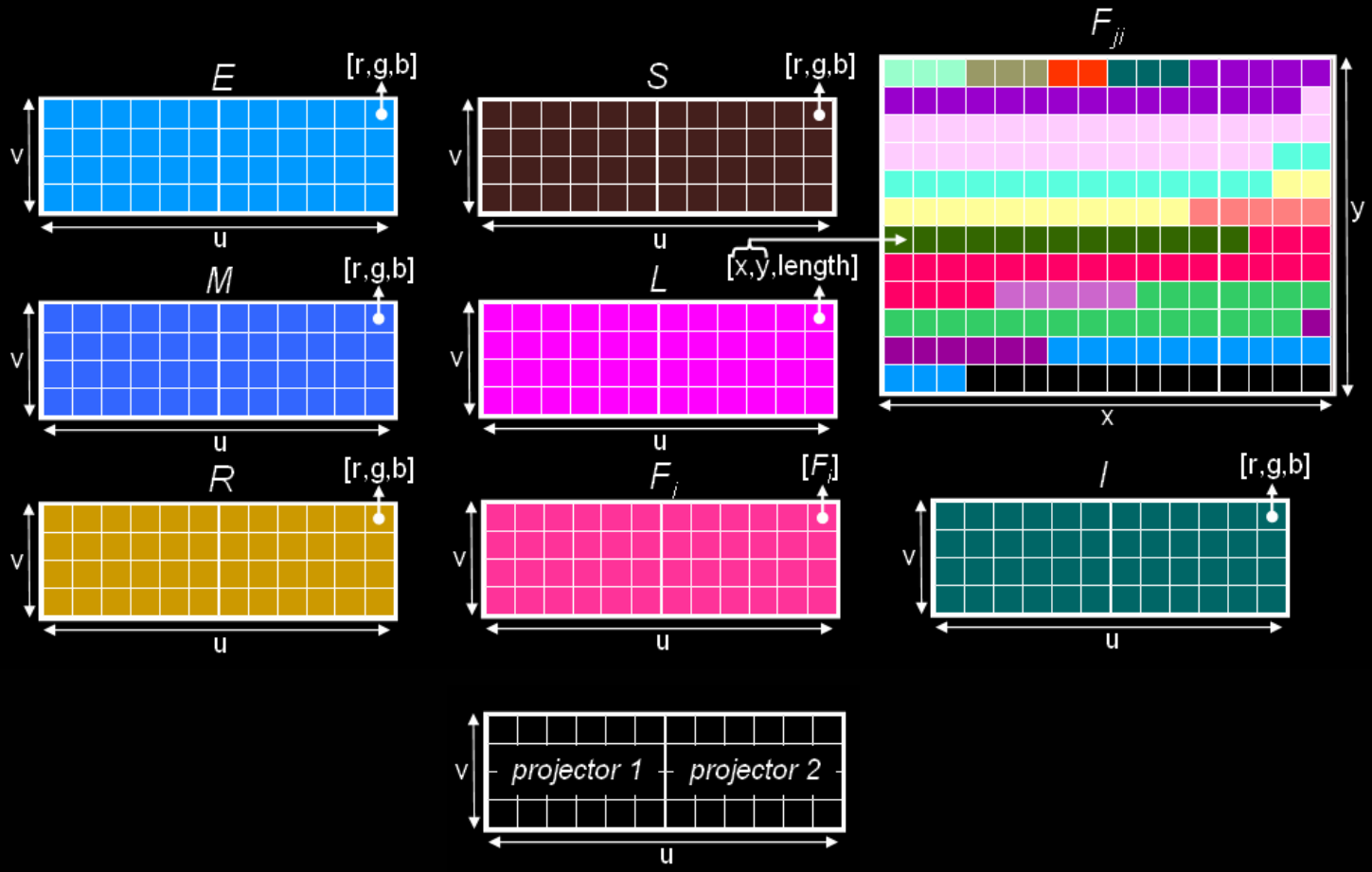
I_Q^0

$\downarrow I_Q^0$

I_P

I_Q

Optimized Data Structures



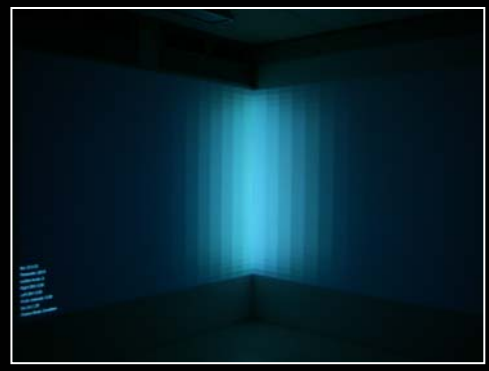
example parameterization for two-sided stereo wall

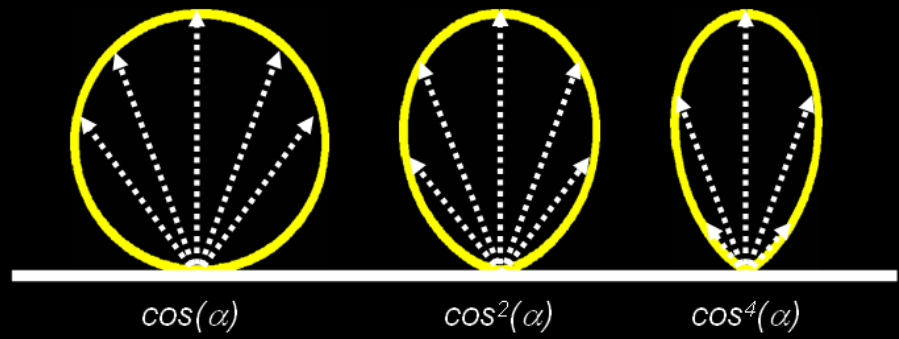
Light Directing Projection Screens

Example approximations for projector-to-patch and patch-to-patch form factors:

$$F_i = \frac{\cos(\alpha_i)}{\pi r_i^2} dA_i$$

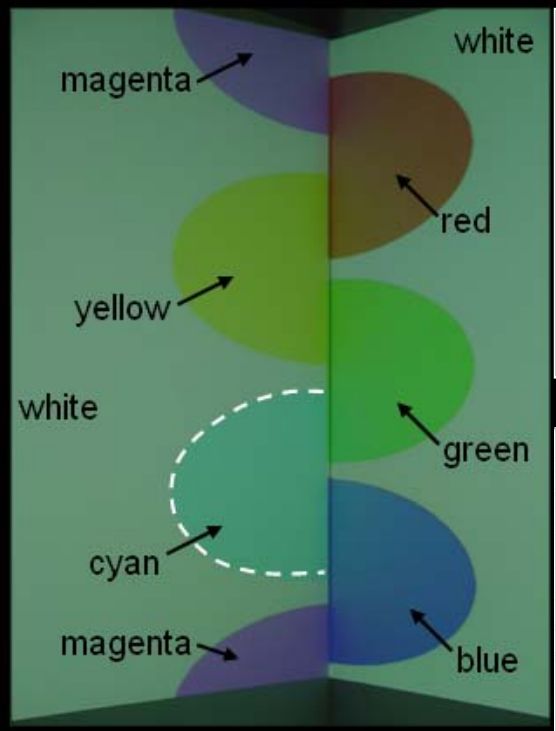
$$F_{ji} = \frac{\cos(\alpha_i)\cos(\alpha_j)}{\pi r_{ji}^2} dA_i$$





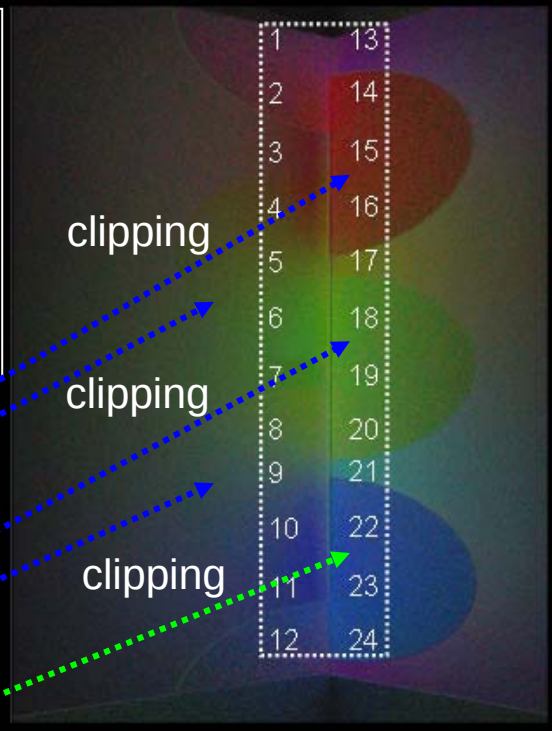
angular dependency of diffuse light directing projection screens (e.g. HPOS, DAB, or SPS) can be approximated by exponentiating the cosine functions

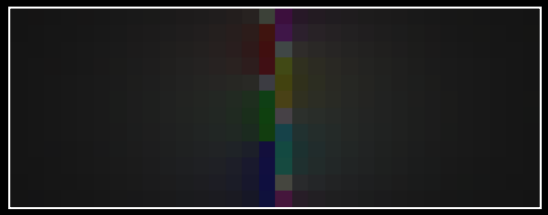
Validation and Limitations



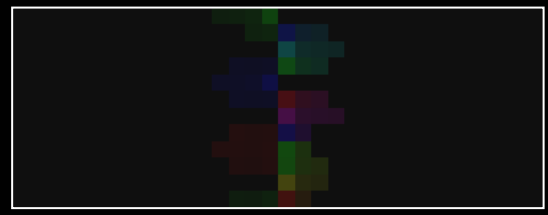
capture uncorrected image *U*
capture corrected image *C*
investigate:
direct radiance *D*
gathered scattering *S*
difference *U-C*

Π	S	D	U-C	U-C	D	S	Π
1	1,1,1	1,0,1	**1,0,1**	1,0,1	1,1,1	1,0,1	13
2	1,0,0	1,0,1	1,0,0	**1,0,0**	1,0,0	1,0,1	14
3	1,0,0	1,1,1	1,0,0	**1,0,0**	1,0,0	1,1,1	15
4	1,0,0	1,1,0	1,0,0	**1,0,0**	1,0,0	1,1,0	16
5	1,1,1	1,1,0	**1,1,0**	1,1,0	1,1,1	1,1,0	17
6	0,1,0	1,1,0	0,1,0	**0,1,0**	0,1,0	1,1,0	18
7	0,1,0	1,1,1	0,1,0	**0,1,0**	0,1,0	1,1,1	19
8	0,1,0	0,1,1	0,1,0	**0,1,0**	0,1,0	0,1,1	20
9	1,1,1	0,1,1	**0,1,1**	0,1,1	1,1,1	0,1,1	21
10	0,0,1	0,1,1	0,0,1	0,0,1	0,0,1	0,1,1	22
11	0,0,1	1,1,1	0,0,1	0,0,1	0,0,1	1,1,1	23
12	0,0,1	1,0,1	0,0,1	0,0,1	0,0,1	1,0,1	24



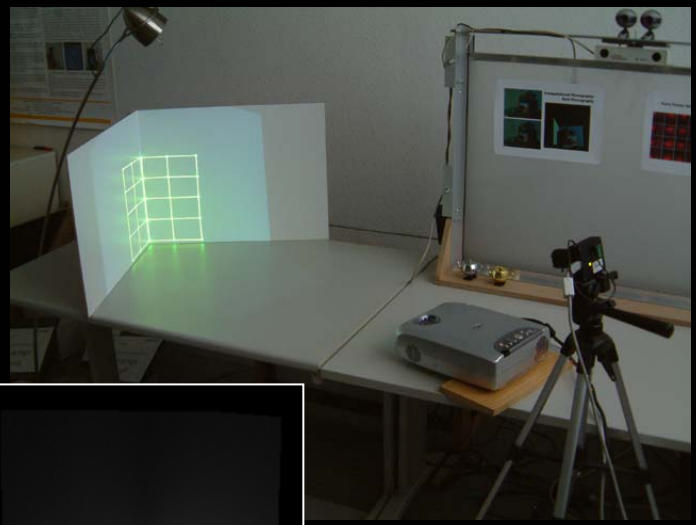


gathered scattering (computed)



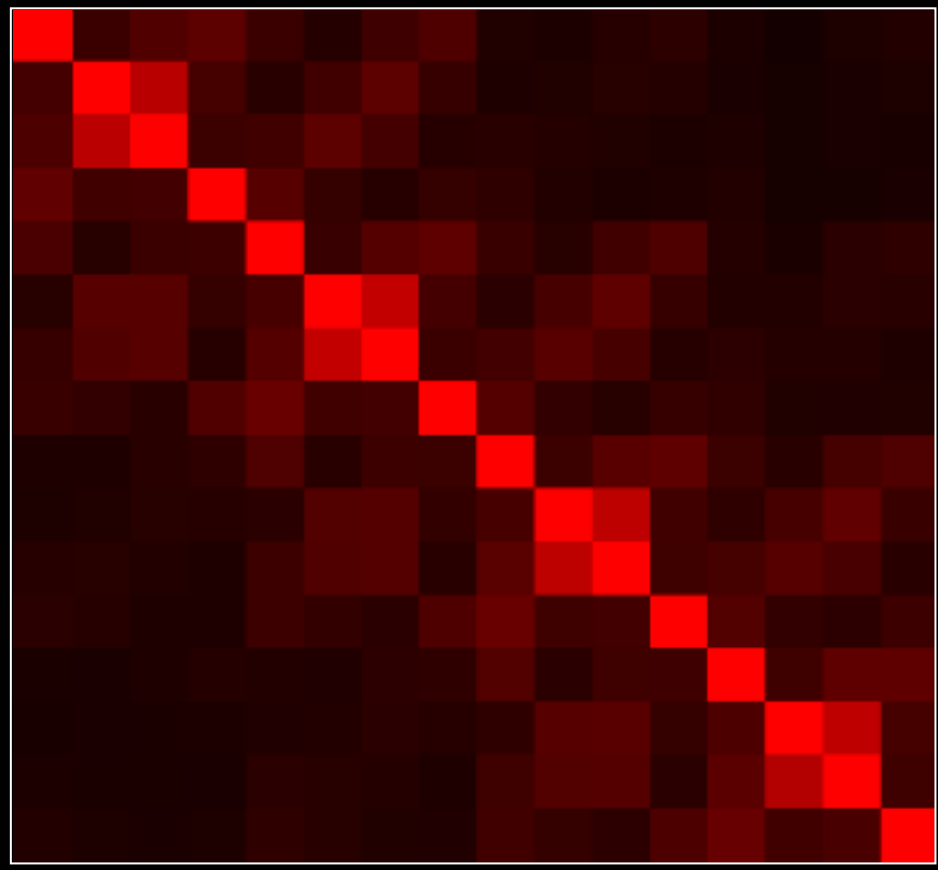
clipping (computed)

Form-factors from Light Transport



experimental setup

more information on
light transport acquisition:
Sen,et al., dual photography,
Siggraph'05

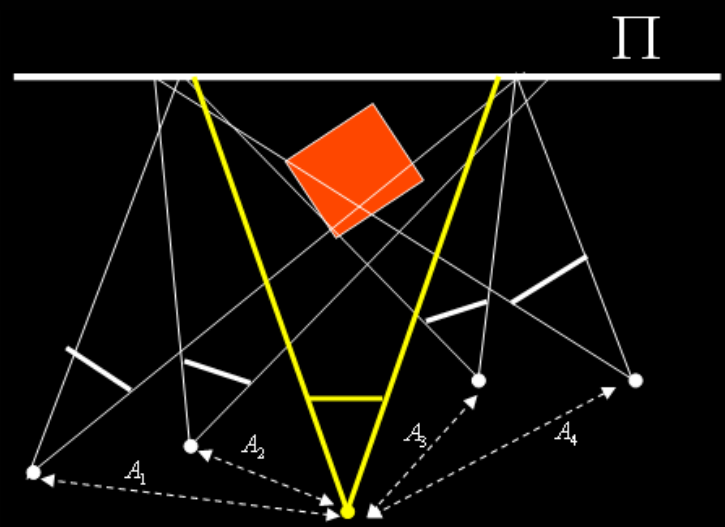


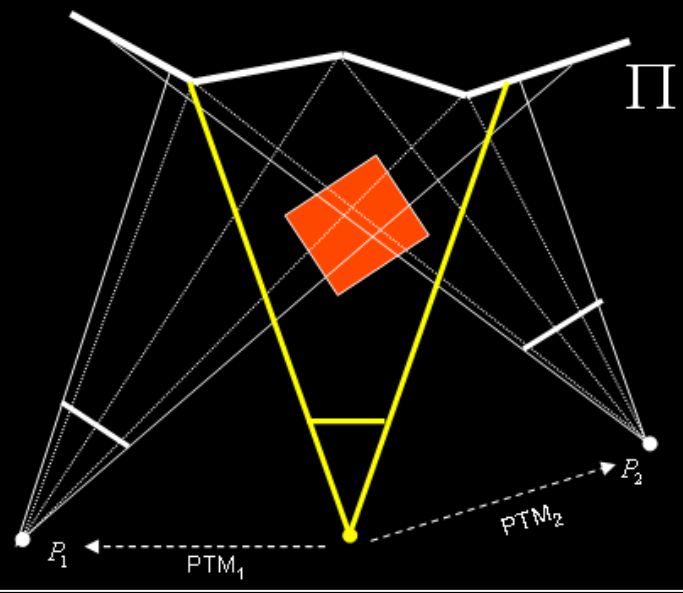
measured 16x16 form-factor matrix
(computed from light transport matrix)

Advanced Techniques
View-Dependence

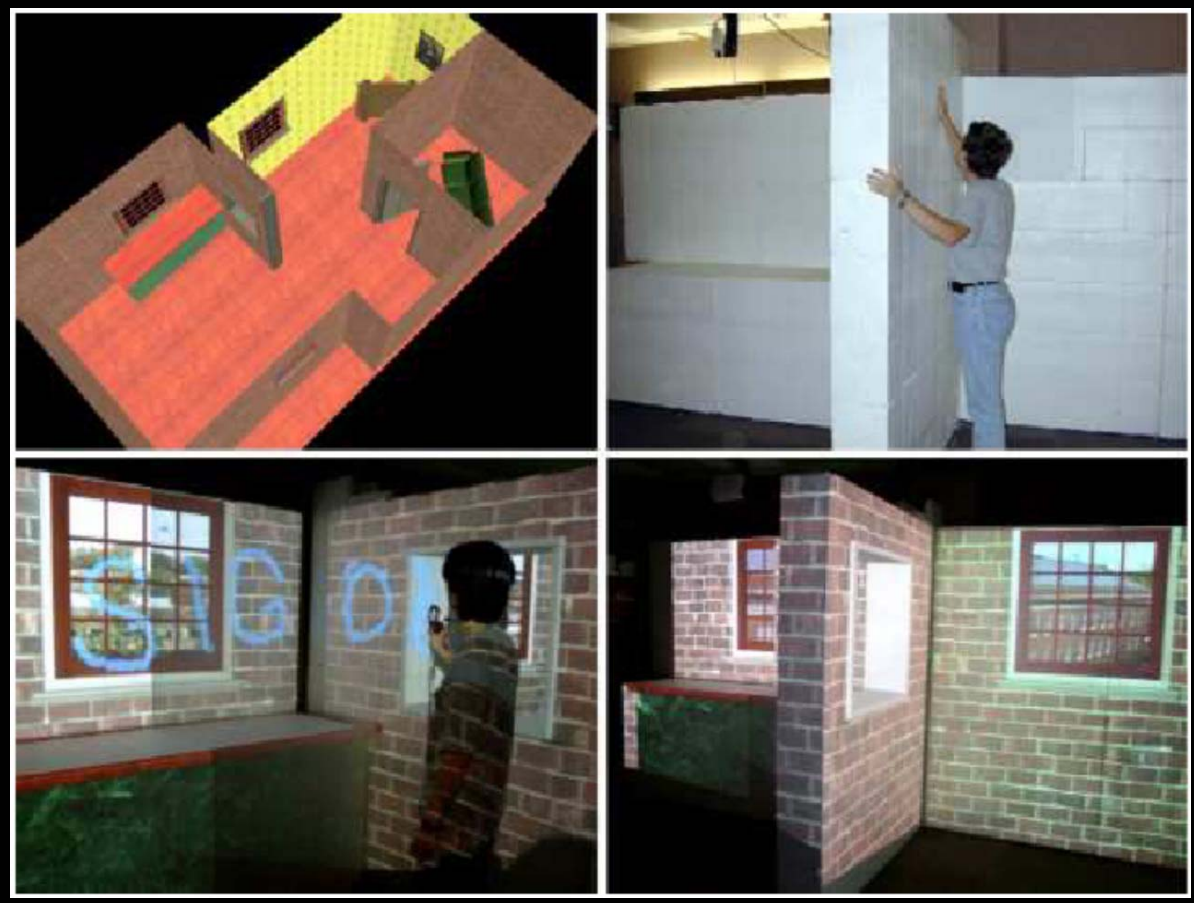
Non-Complex Surfaces

- view-dependent geometry correction can be compute if geometry is known
- for example:
  - **planar/multi-plane:** off-axis projection
  - **parametric:** warping via parametric description
  - **scanned/modelled:** projective texture mapping





Example: Life-Sized Projector-Based Dioramas



Courtesy: Low, et al., 2001

Complex Surfaces

- if geometry is unknown image-based rendering helps
- sample geometric and radiometric parameters from multiple (source) camera (perspective)
- for novel (destination) camera
  - compute weighted penalties:

$$p_j = \alpha a_j + (1-\alpha) b_j$$

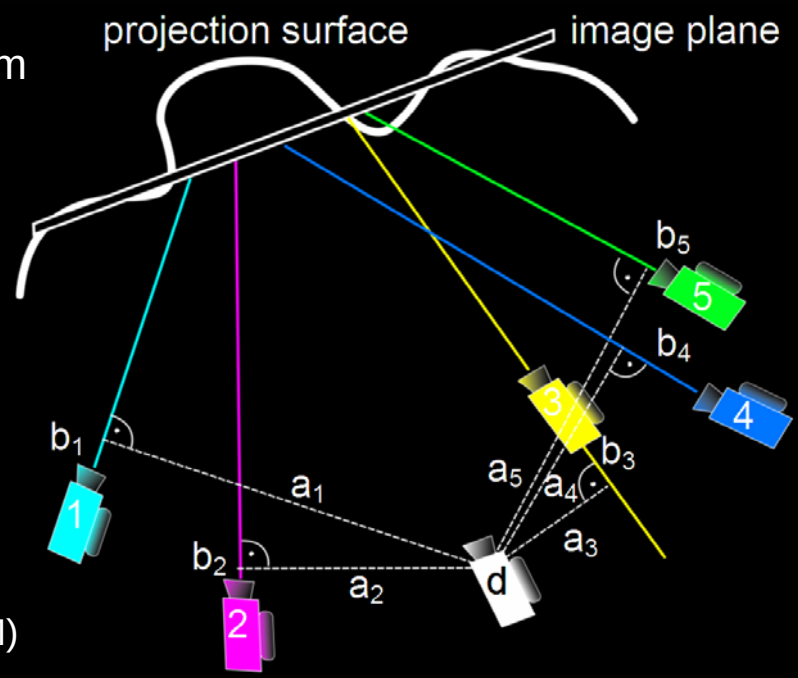
  - select k best perspectives (lowest penalties) and normalize them:

$$w_j = \left(1 - \frac{p_j}{\max_{pk}}\right) \frac{1}{p_j}$$

  - interpolate new parameter textures (P_i2C_j, $F_{ij}M$, $E_{ij}M$) and direction vector for destination perspective to render new IP:

$$t_d = \sum_j^k w_j t_j$$

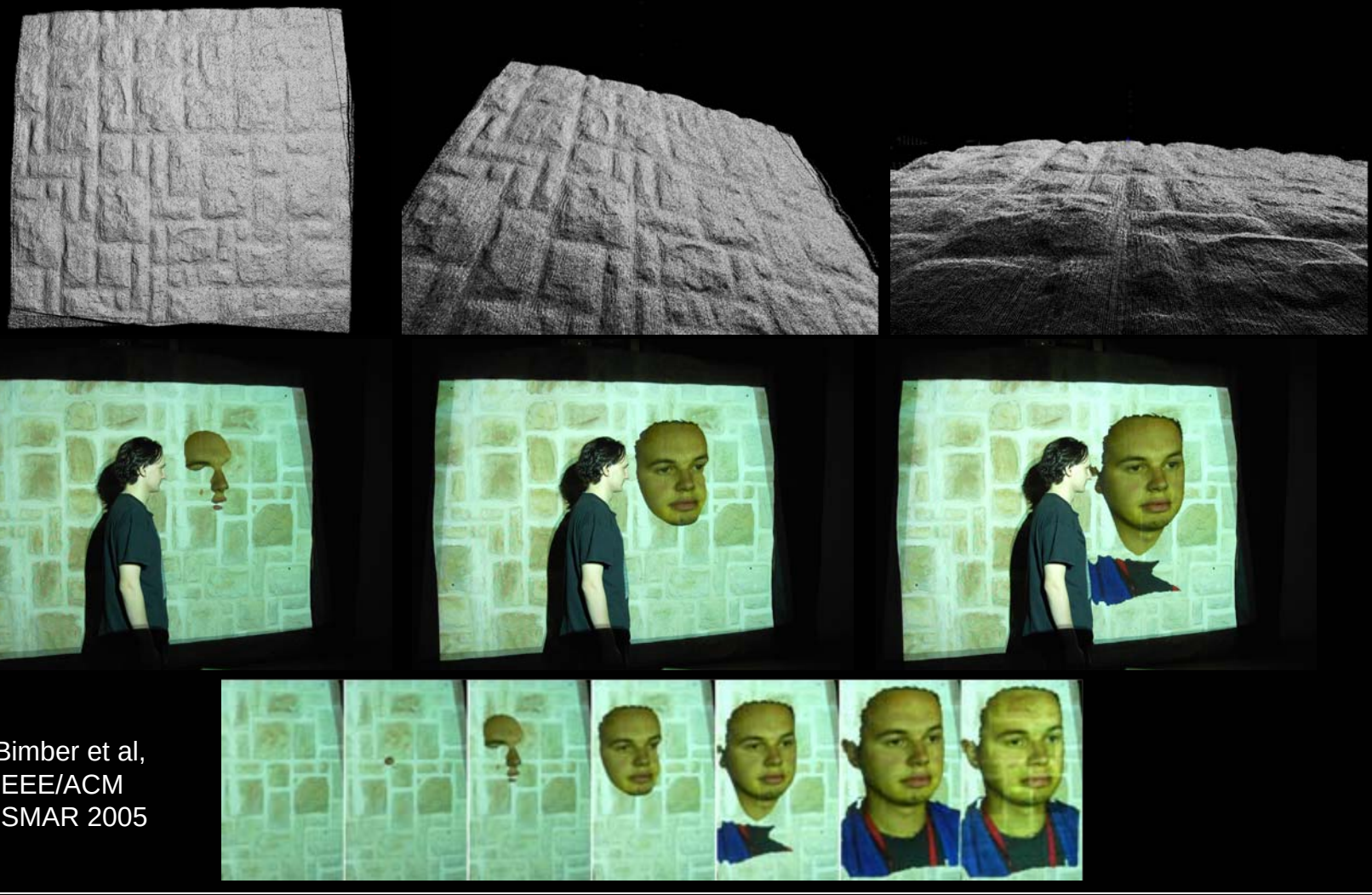
  - lookups in $F_{ij}M$, $E_{ij}M$ with non-interpolated P_i2C_j
  - lookups in IP with interpolated P_i2C_j



Example: Tracking and Stereo

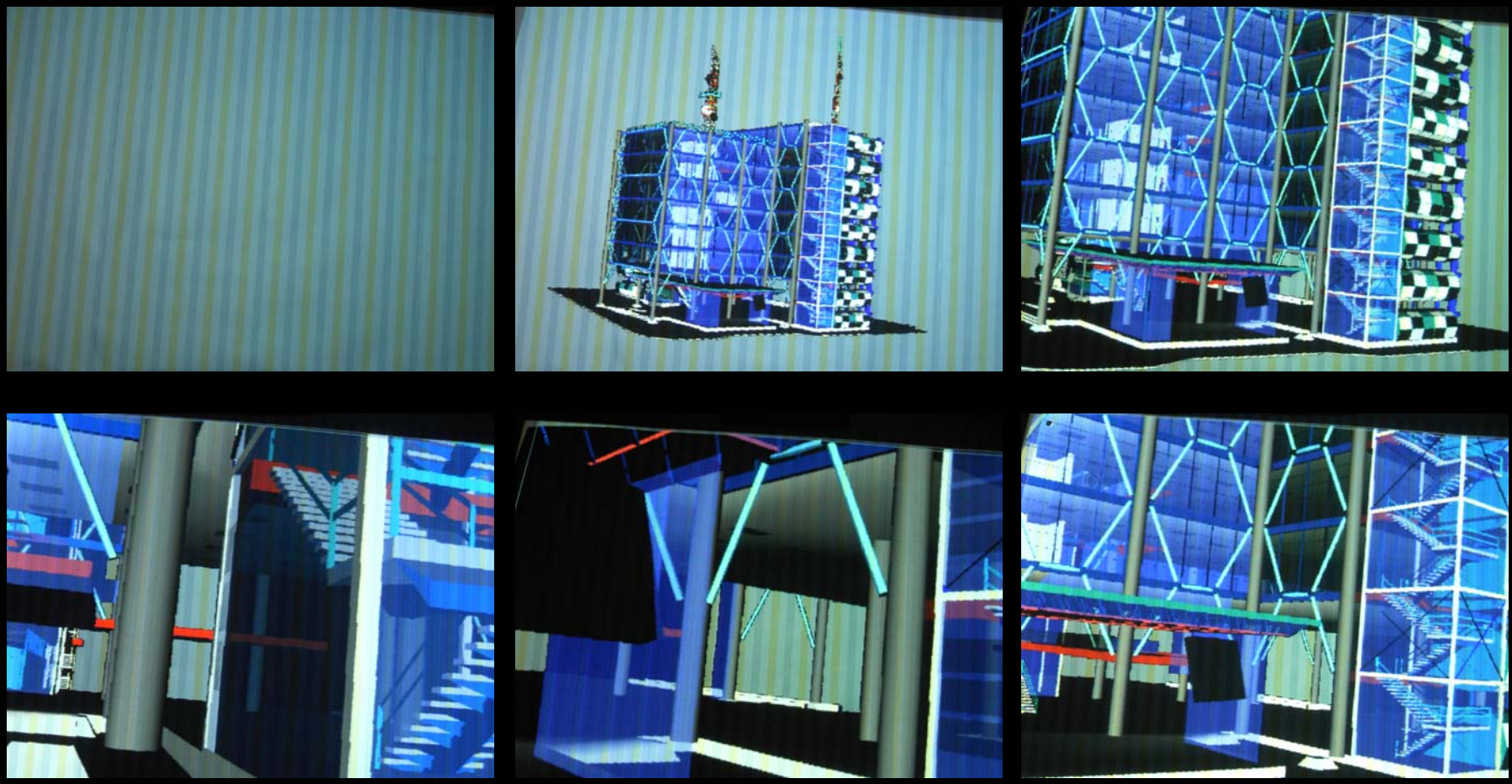
Bimber et al, IEEE/ACM ISMAR 2005

Depth and Occlusion



Bimber et al,
IEEE/ACM
ISMAR 2005

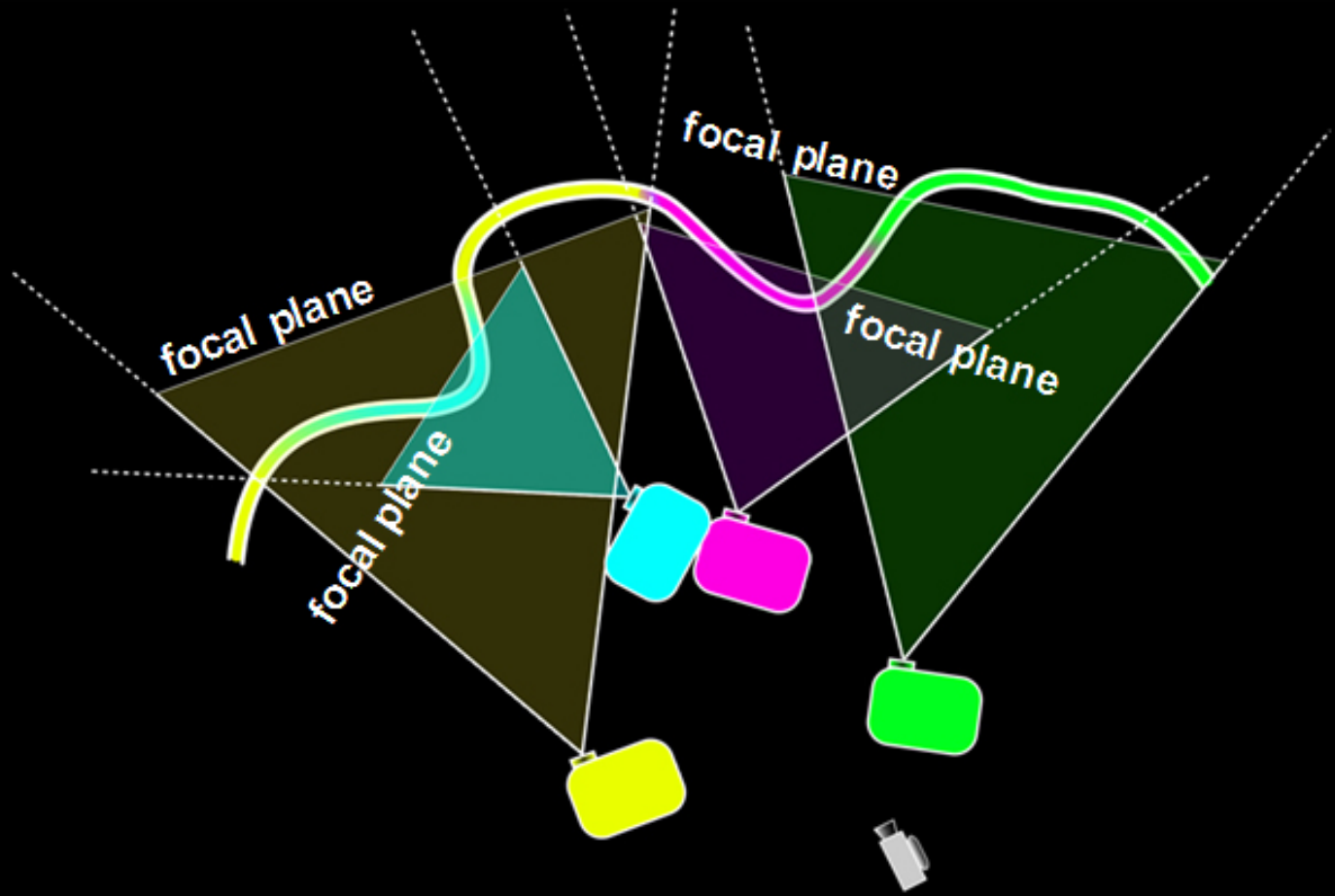
Example: Stereo on Wallpaper



Bimber et al, IEEE/ACM ISMAR 2005

Advanced Techniques
Multi-Focal Projection

Multi-Projector-Camera Technique that Increases Focal Depth



Determining Defocus

- structured light projection of grid point samples (2- dimensional phase shift)
  - **pre-correction:** geometric and radiometric correction (corrected grid points must be observed in camera)

$$I_{x,y} = (R_{x,y} - EM_{x,y}) / FM_{x,y}$$

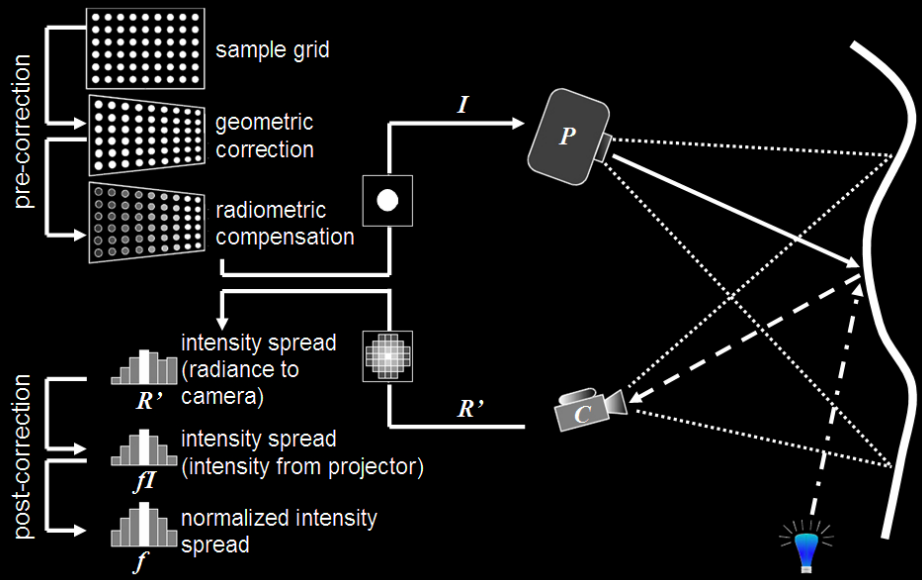
  - **post-correction:**

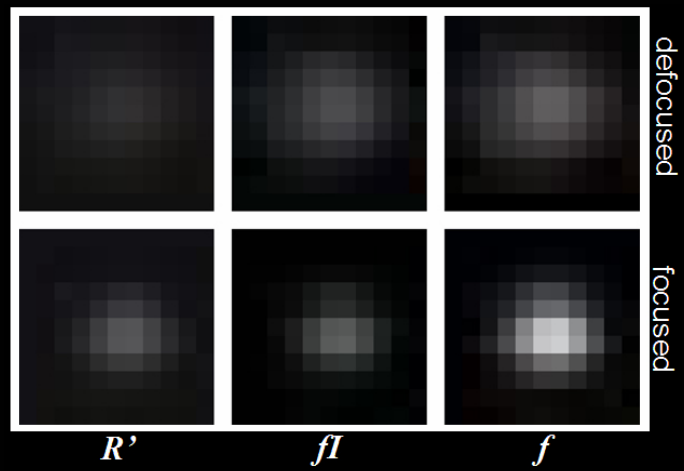
$$R'_{x',y'} = fI_{x,y} FM_{x',y'} + EM_{x',y'}$$

$$fI_{x,y} = \frac{(R'_{x',y'} - EM_{x',y'})}{FM_{x',y'}}$$

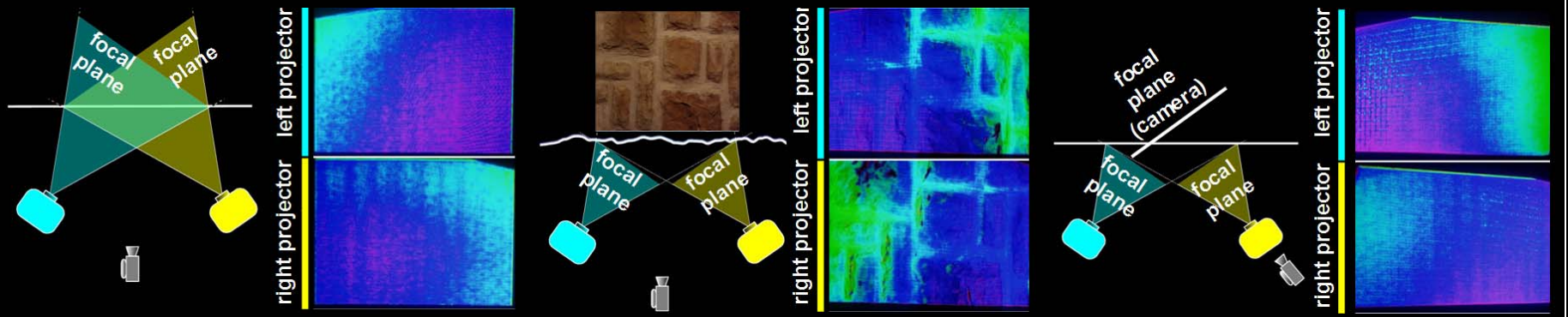
$$f = \frac{fI_{x,y}}{I_{x,y}}$$

  - the normalized intensity spread texture f serves as basis to estimate focus measures (e.g., via FFT/DCT, intensity loss, point spread, etc.)



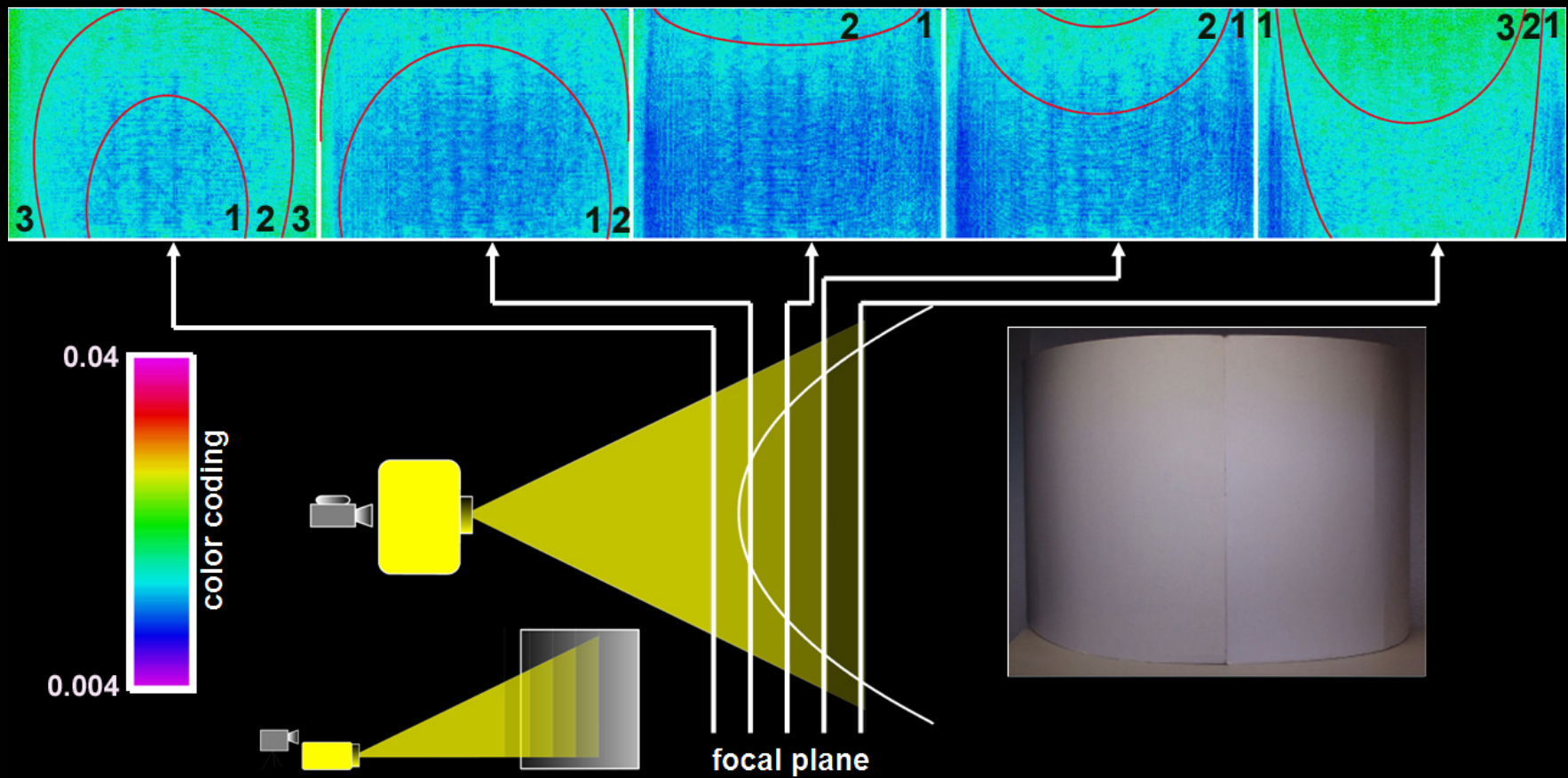


Example: Different Configurations



Bimber et al, IEEE TVCG 2006

Example: Shifting Focal Plane



Bimber et al, IEEE TVCG 2006

Image Composition

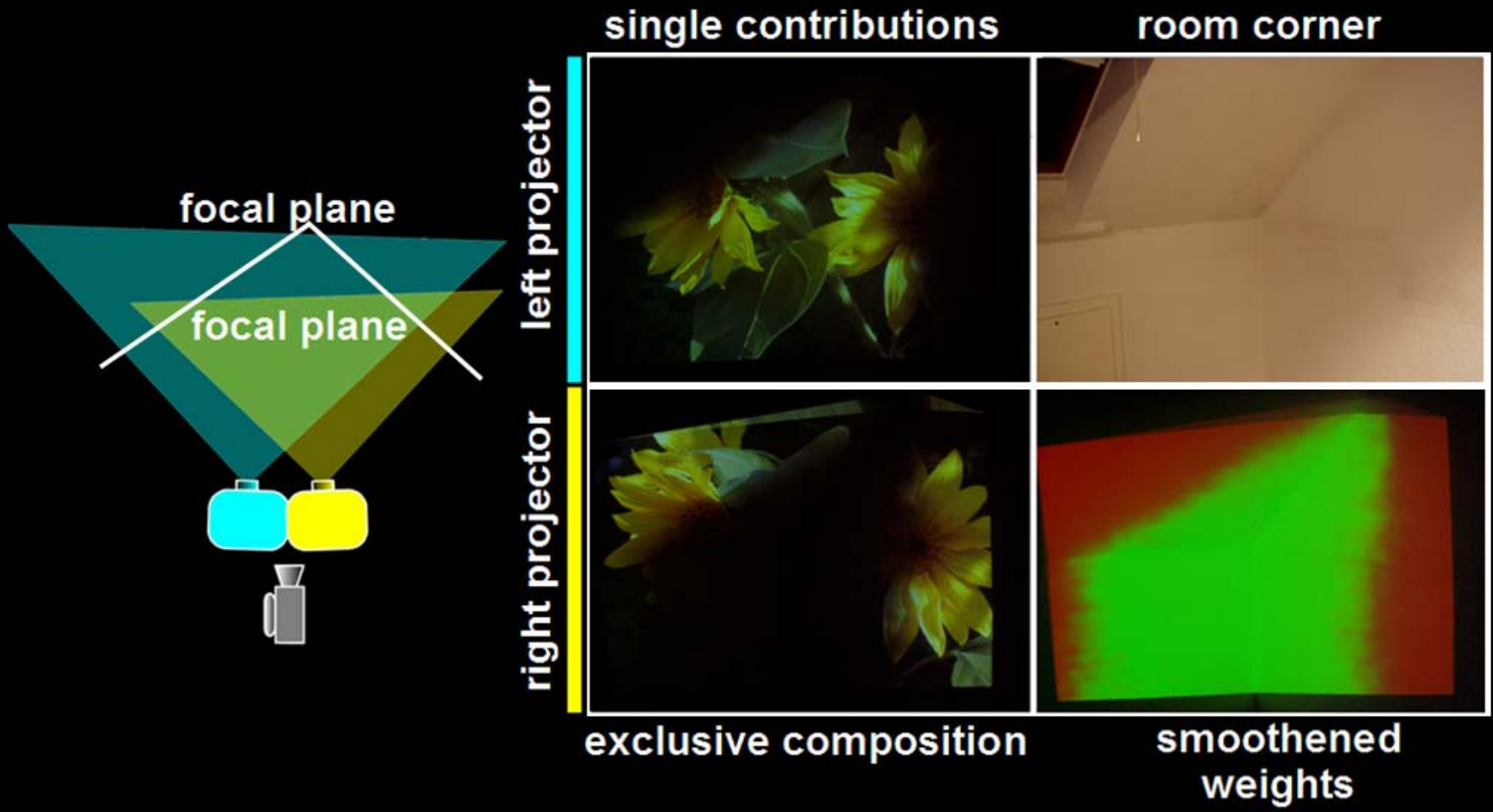
- using the focus values of each projector's pixels ($\Phi_{i,x,y}$), compose an image with minimal total defocus
  - **excusive composition:** surface point is covered by a single projector pixel (the one with highest $\Phi_{i,x,y}$)

$$I_i = \frac{w_i(R - EM)}{\sum_j^N w_j FM_j} \qquad w_{i,x,y} = \frac{\Phi_{i,x,y}}{\sum_j^N \Phi_{j,x,y}}$$

  - **weighted composition:** compute normalized weight and multiply it with *FM* and *I*

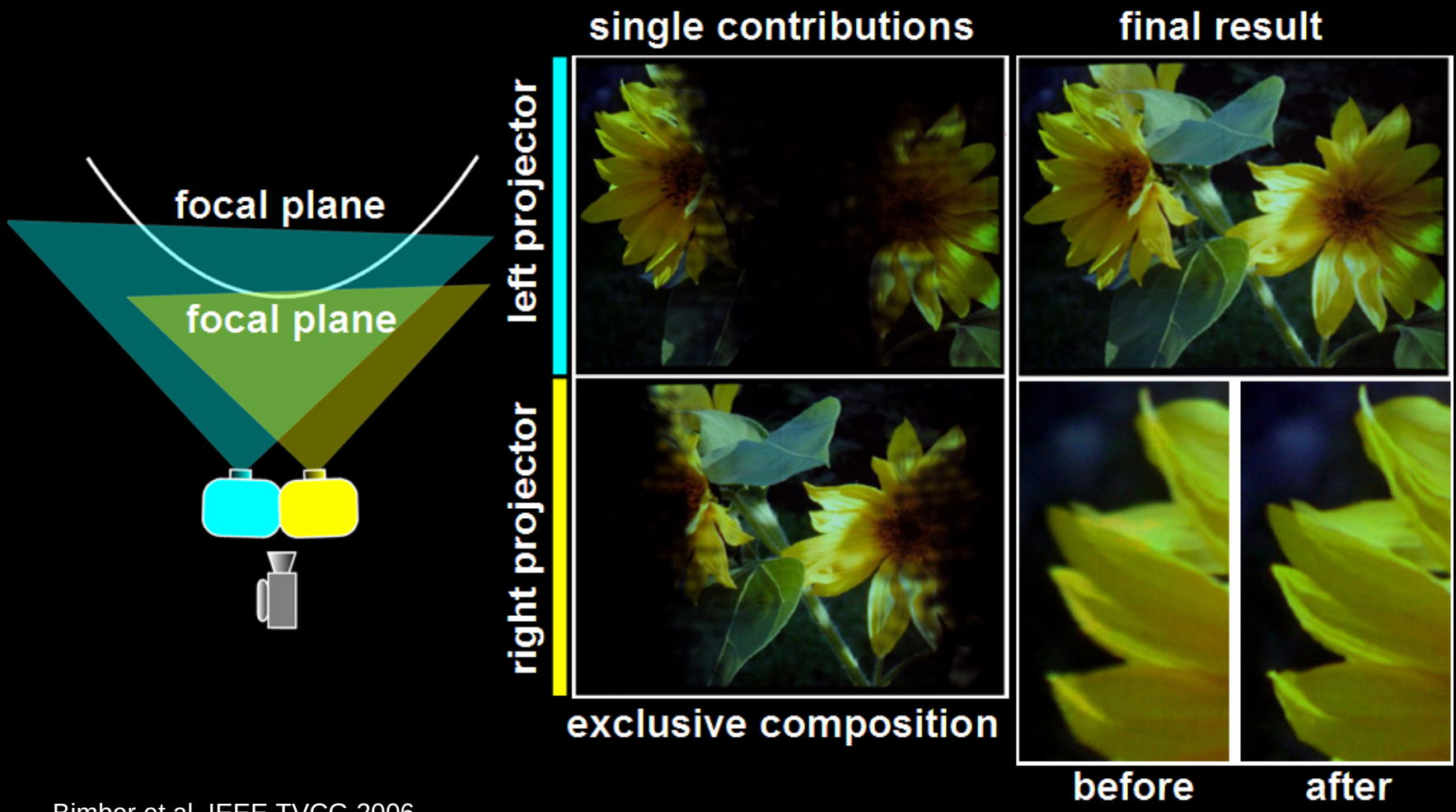
$$I_i = w_i(R\text{-}EM)/FM_i, \quad w_i = \begin{cases} 1 & \Phi_{i,x,y} \geq \Phi_{j,x,y} \\ 0 & else \end{cases}$$

Example: Room Corner



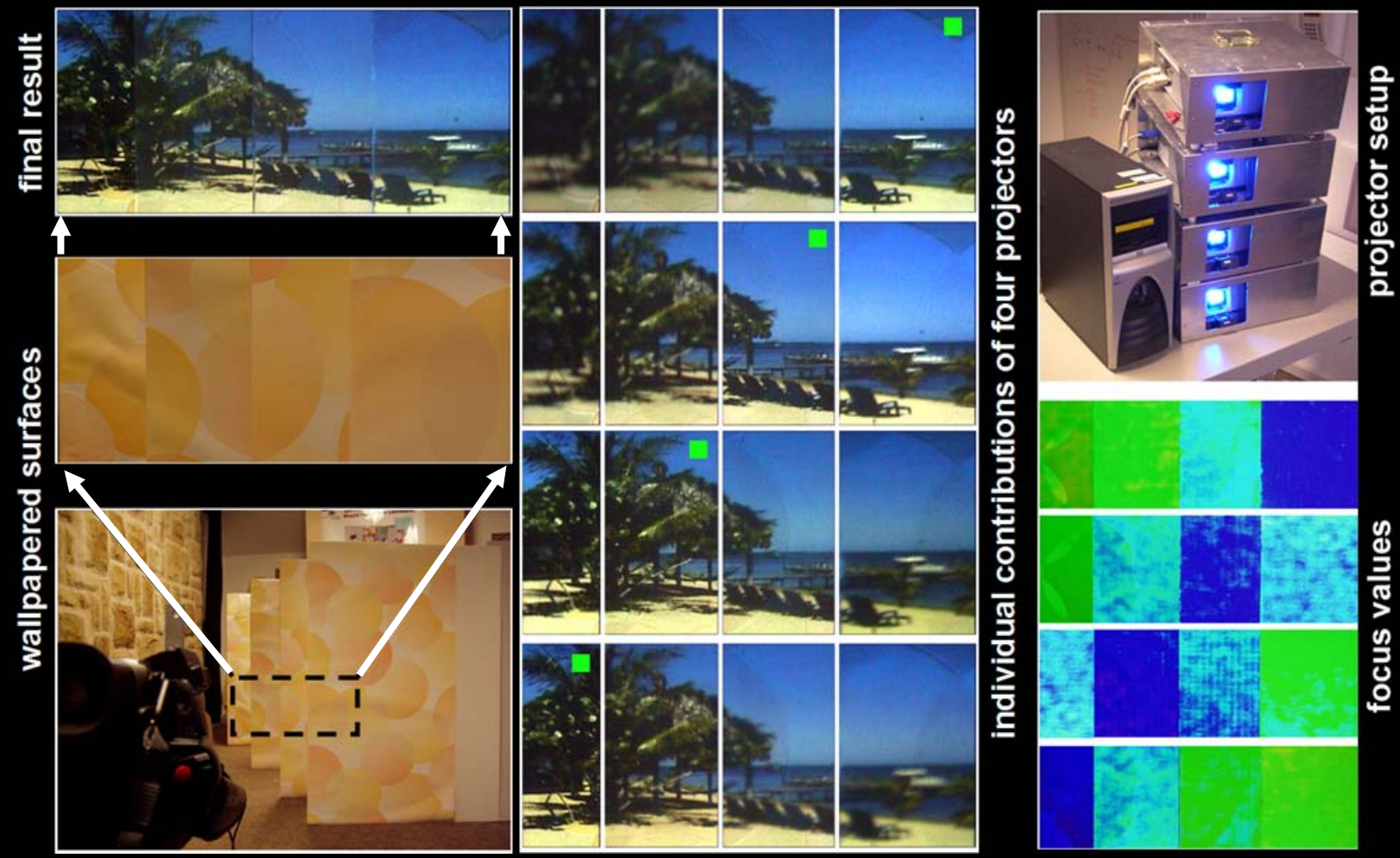
Bimber et al, IEEE TVCG 2006

Example: Cylindrical Surface



Bimber et al, IEEE TVCG 2006

Example: Large Focal Depth

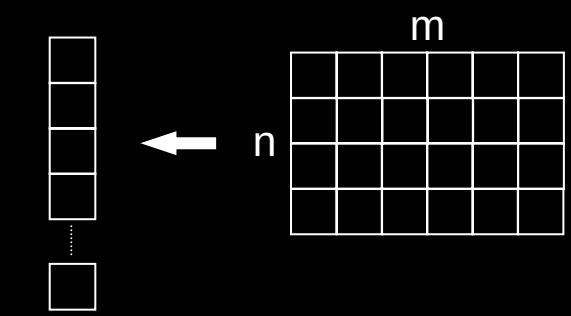


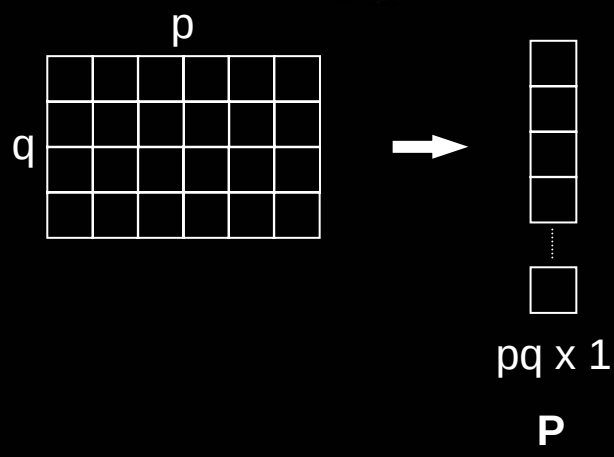
Bimber et al, IEEE TVCG 2006

Advanced Techniques
Light Transport

Acquisition

single camera & projector capture 4D slice of 8D reflectance field





$$c = Tp$$

$$T = \begin{bmatrix} & pq & \\ & & \\ & & \end{bmatrix} mn$$

Acquisition

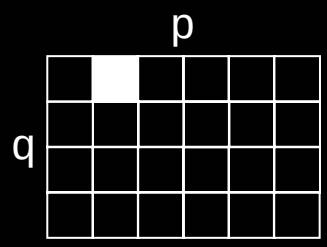
single camera & projector capture 4D slice of 8D reflectance field

$$c = Tp$$

$$T = \begin{bmatrix} & \\ & \\ & \end{bmatrix}$$

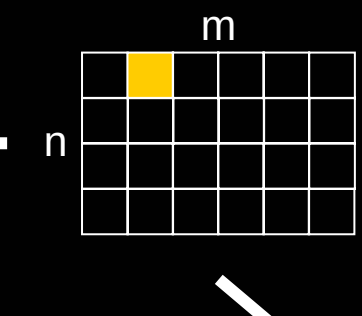
Acquisition

single camera & projector capture 4D slice of 8D reflectance field



pq x 1

P



mn x 1

c

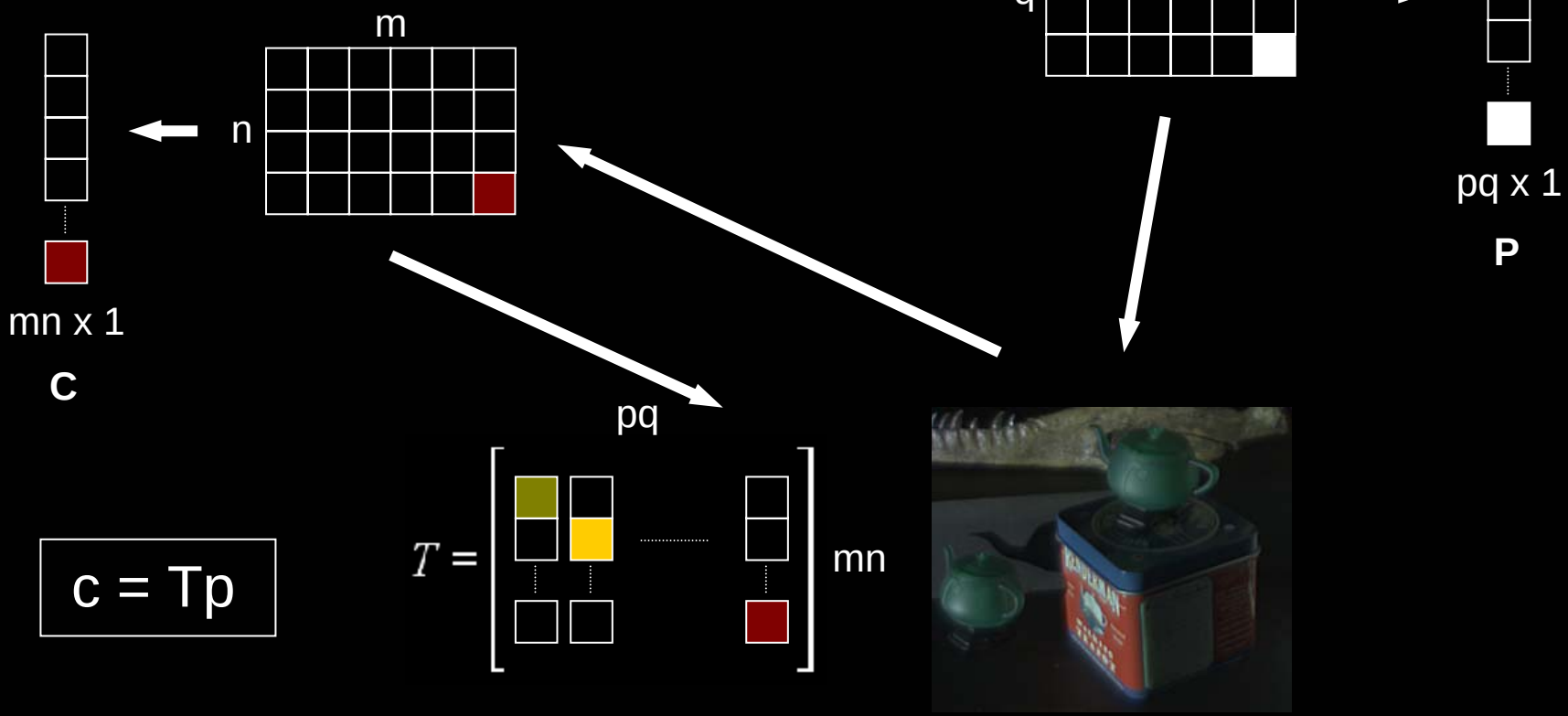
pq

$$T = \begin{bmatrix} \\ \\ \\ \end{bmatrix}_{mn}$$

$$c = Tp$$

Acquisition

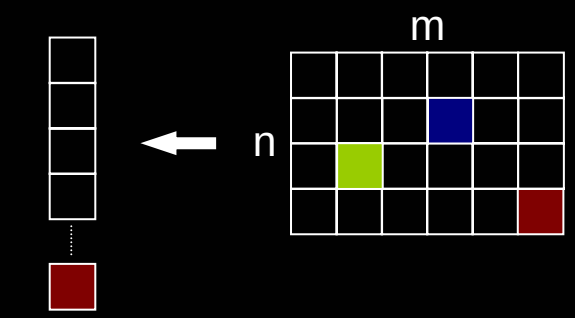
single camera & projector capture 4D slice of 8D reflectance field

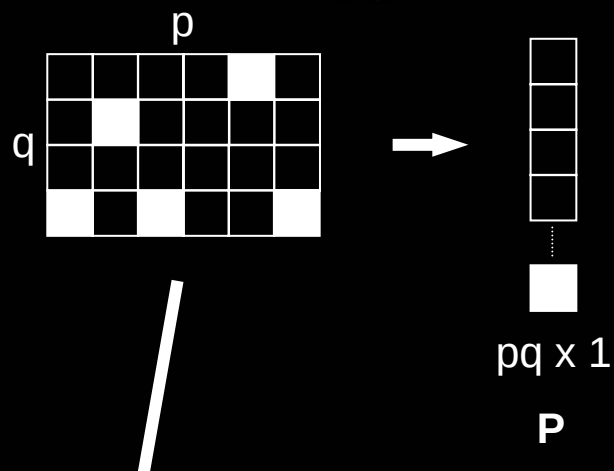


c = Tp

Acquisition

single camera & projector capture 4D slice of 8D reflectance field

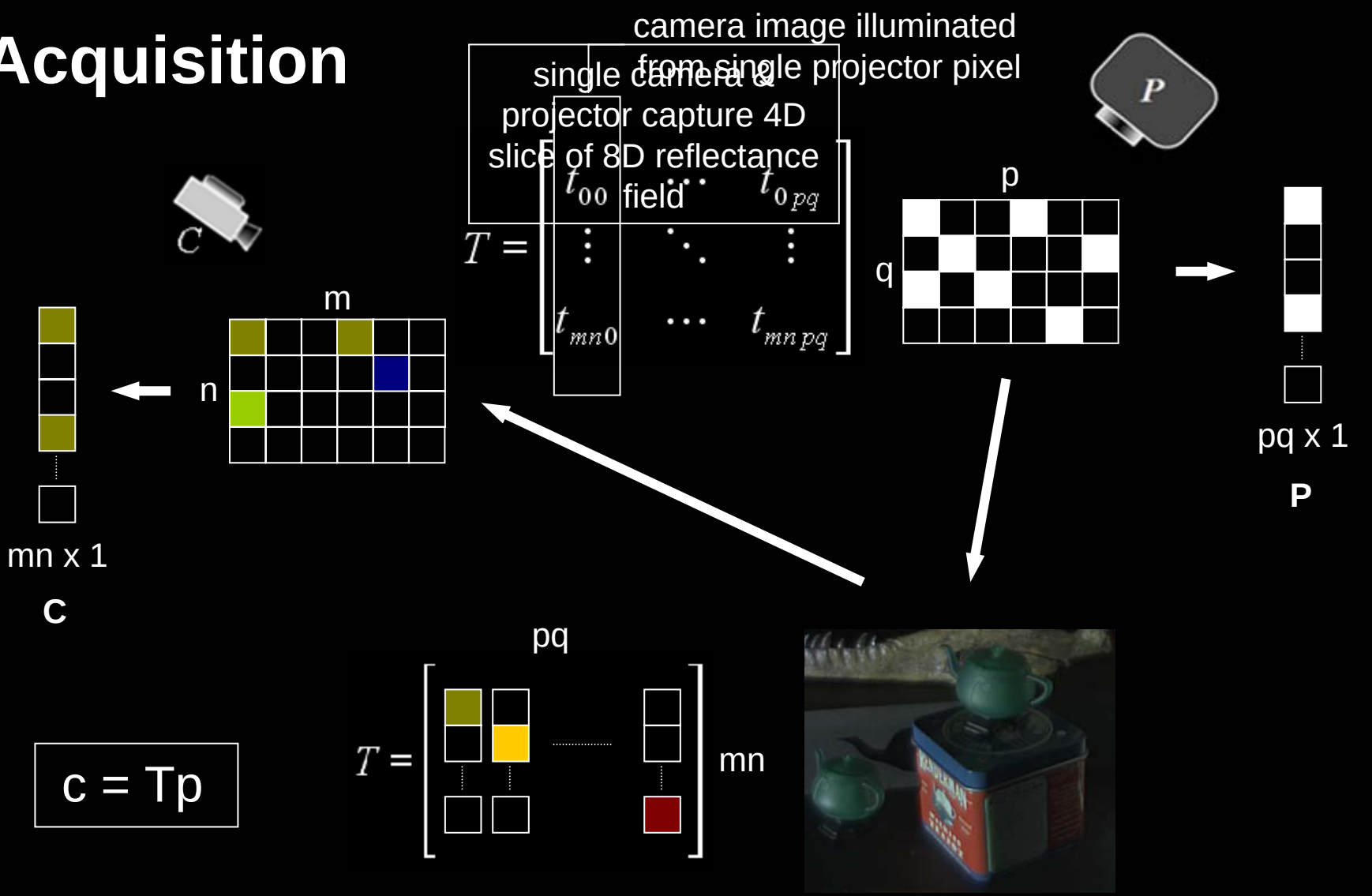




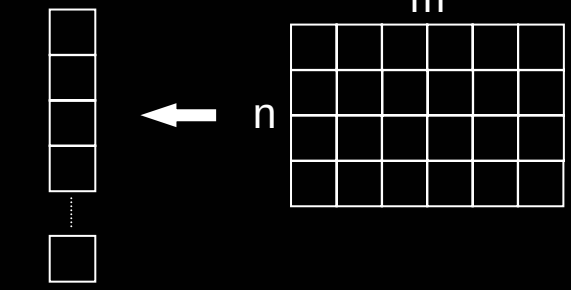
$c = Tp$

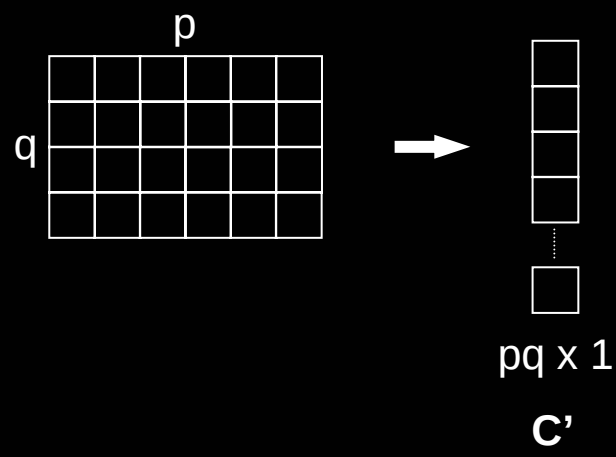
$$T = \begin{bmatrix} & \cdots & \\ & \cdots & \\ & \cdots & \end{bmatrix}_{mn}^{pq}$$

Acquisition



Dual Photography





$c = Tp$

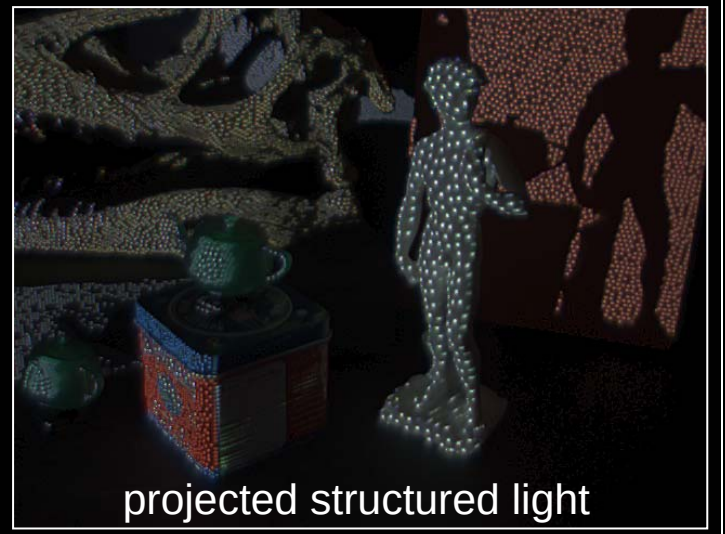
$c' = T^T p'$

$$T = \begin{bmatrix} \cdots \\ \cdots \end{bmatrix}_{mn}^{pq}$$

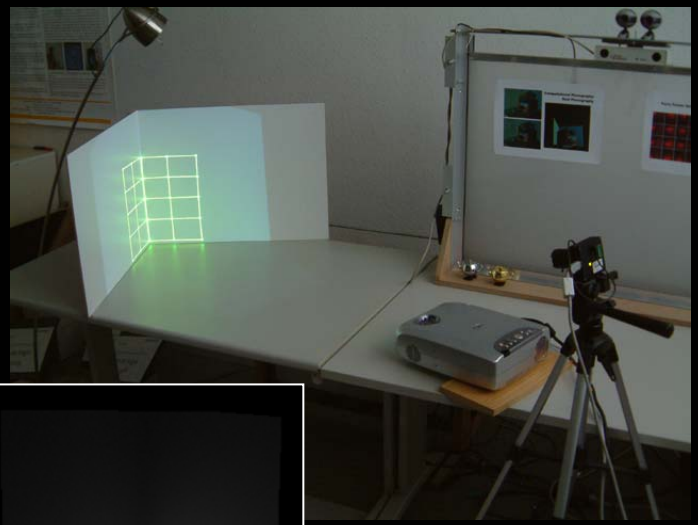
$$T^T = \begin{bmatrix} \cdots \\ \cdots \end{bmatrix}_{pq}^{mn}$$

Dual Photography

more information on dual photography: Sen,et al., Siggraph'05

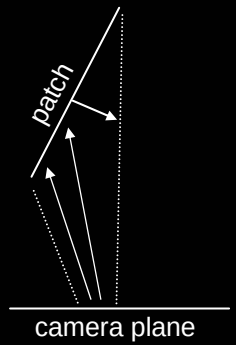


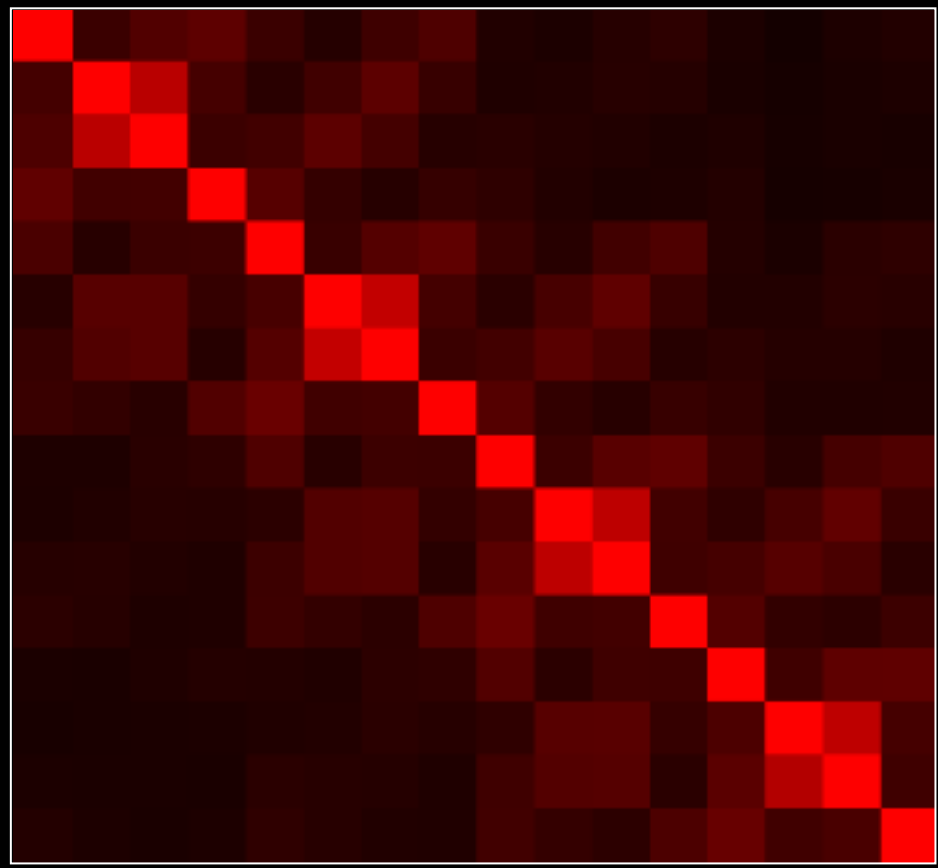
Form-Factors from Light Transport Matrix



experimental setup

$$M_e = \frac{d^2}{\cos\alpha} L_e$$

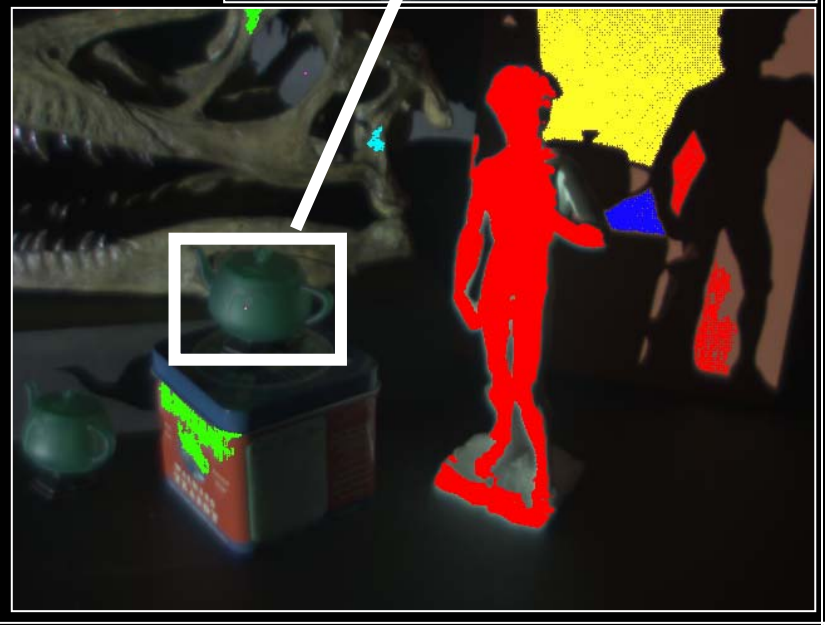




measured 16x16 form-factor matrix
(computed from light transport matrix)

Global Radiometric Compensation

- traditional radiometric compensation requires direct projector-camera pixel correspondence
- include arbitrary global illumination effects using T
- apply inverse light transport $T^{-1}C=P$
- since T is huge, decompose it into clusters and solve in real-time on GPU

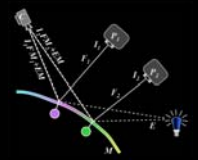


$$C = T\ P$$

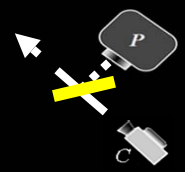
$$\begin{bmatrix} c_{10} \\ c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \end{bmatrix} = \begin{bmatrix} t^4_{10} & t^3_{10} \\ t^4_{11} & t^3_{11} \\ t^4_{12} & t^3_{12} \\ t^4_{13} & t^3_{13} \\ t^4_{14} & t^3_{14} \end{bmatrix} \begin{bmatrix} p_4 \\ p_3 \end{bmatrix} \rightarrow \begin{bmatrix} t^4_{10} & t^4_{11} & t^4_{12} & t^4_{13} & t^4_{14} \\ t^3_{10} & t^3_{11} & t^3_{12} & t^3_{13} & t^3_{14} \end{bmatrix}^{-1} \begin{bmatrix} c_{10} \\ c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \end{bmatrix} = \begin{bmatrix} p_4 \\ p_3 \end{bmatrix}$$

$$T^{-1} \qquad C = P$$

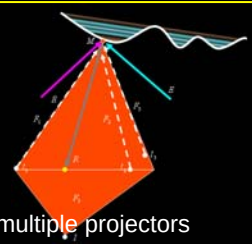
Outlook


single projector

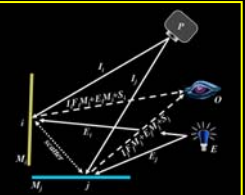
$$V = \begin{bmatrix} V_{RR} & V_{RG} & V_{RB} \\ V_{GR} & V_{GG} & V_{GB} \\ V_{BR} & V_{BG} & V_{BB} \end{bmatrix}$$
color mixing



dynamic adaptation

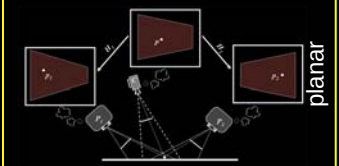

multiple projectors

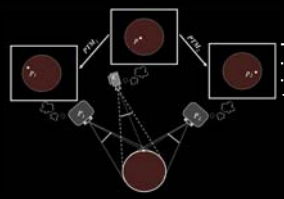
perception

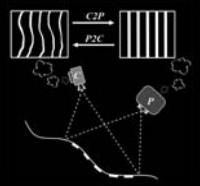

global effects

radiometric compensation

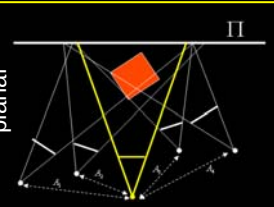
geometric warping

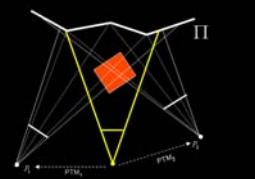

planar

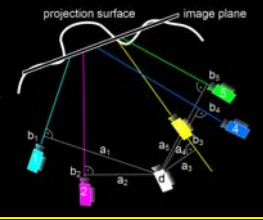

non-trivial


complex

view-dependence


planar

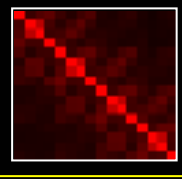

non-trivial



complex

light transport

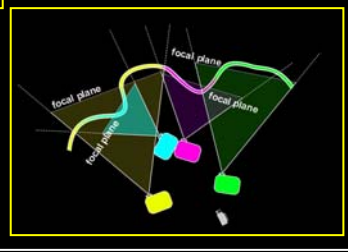
$$T = \begin{bmatrix} t_{00} & \cdots & t_{0pq} \\ \vdots & \ddots & \vdots \\ t_{mn0} & \cdots & t_{mnpq} \end{bmatrix}$$


global parameters

dual image

global rad. comp.

multi-focal projection

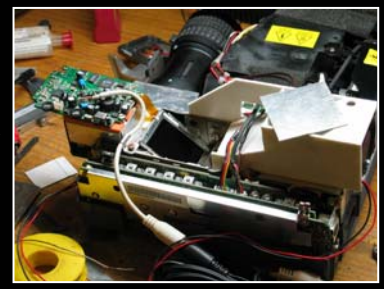


Limitations

- technological limitations of projectors:
  - brightness, resolution, focal depth
  - black-level and dynamic range
  - size, cost, portability
- technological limitations of cameras:
  - dual to limitations of projectors
- physical limitations of surfaces:
  - non-Lambertian: refraction, specular reflection, inter-reflections, sub-surface scattering, dispersion, diffraction, etc.

Future Work

- new techniques:
  - consider human visual perception
    - spent computational power only on overcoming limitations that can actually be perceived
  - consider global effects
    - inter-reflections, scattering, etc.
- new applications:
  - example: immersive visualizations of light simulations and materials
- new hardware:
  - example: HDR projector



Selected Papers on Geometric Correction

Bimber, O., Wetzstein, G., Emmerling, A., & Nitschke, C. (2005). Enabling View-Dependent Stereoscopic Projection in Real Environments. *Proc. of IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'05)*, 14-23.

Low, K-L., Welch, G., Lastra, A., & Fuchs, H. (2001). Life-Sized Projector-Based Dioramas, *Proc. Symp. Virtual Reality Software and Technology (VRST'01)*, 93-101.

Raskar, R. (1999). Oblique Projector Rendering on Planar Surfaces for a Tracked User. *Proc. of ACM Siggraph'99*, sketch.

Raskar, R., Brown, M.S., Yang, R., Chen, W., Welch, G., Towles, H., Seales, B., & Fuchs, H. (1999b). Multi-projector displays using camera-based registration, *Proc. of IEEE Visualization* (IEEE Viz'99), 161-168.

Raskar, R., Welch, G., Low, K.L. & Bandyopadhyay, D. (2001). Shader Lamps: Animating real objects with image-based illumination. *Proc. of Eurographics Rendering Workshop*, 89-102.

Selected Papers on Radiometric Compensation

Bell, I.E. (2003). Neutralizing Paintings with a Projector. *Proc. of SPIE/IS&T*, 5008, 560-568.

Bimber, O., Coriand, F., Kleppe, A., Bruns, E., Zollmann, S., & Langlotz, T. (2005). Superimposing Pictorial Artwork with Projected Imagery. *IEEE MultiMedia*. 12(1), 16-26.

Bimber, O., Grundhöfer, A., Zeidler, T., Danch, D., & Kapakos, P. (2006). Compensating Indirect Scattering for Immersive and Semi-Immersive Projection Displays. *Proc. of IEEE Virtual Reality (IEEE VR'06).*

Fujii, K., Grossberg, M.D., & Nayar, S.K. (2005). A projector-camera system with real-time photometric adaptation for dynamic environments. *Proc. of Computer Vision and Pattern Recognition (CVPR'05)*, 2, 20-25.

Grossberg, M.D., Peri, H., Nayar, S.K., & Bulhumeur, P. (2004). Making One Object Look Like Another: Controlling Appearance Using a Projector-Camera System. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04)*, 1, 452-459.

Nayar, S.K., Peri, H., Grossberg, M.D., & Belhumeur, P.N. (2003). A Projection System with Radiometric Compensation for Screen Imperfections. *Proc. of International Workshop on Projector-Camera Systems (ProCams'03).*

Wang, D., Sato, I., Okabe, T., & Sato, Y. (2005). Radiometric Compensation in a Projector-Camera System Based on the Properties of Human Vision System. *In Proc. of IEEE International Workshop on Projector-Camera Systems (ProCams'05).*

Yoshida, T., Horii, C. & Sato, K. (2003). A Virtual Color reconstruction System for Real Heritage with Light Projection. *Proc. of Virtual Systems and Multimedia*, 158-164.

Selected Papers Other and Related Techniques

Bimber, O. & Emmerling, A. (2006). Multi-Focal Projection: A Multi-Projector Technique for Increasing Focal Depth. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Brown, M., Majumder, A., and Yang, R. (2005). Camera-Based Calibration Techniques for Seamless Multi-Projector Displays. I*EEE Transactions on Visualization and Computer Graphics (TVCG)*, 11(2), 193-206.

Cotting, D., Naef, M., Gross, M., & Fuchs, H. (2004). Embedding Imperceptible Patterns into Projected Images for Simultaneous Acquisition and Display. *Proc. of IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*, 100-109.

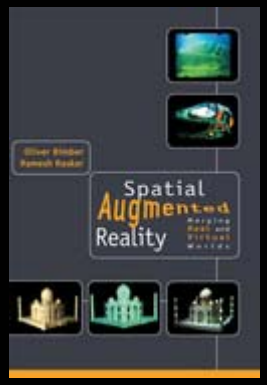
Ehnes, J., Hirota, K., & Hirose, M. (2004). Projected Augmentation – Augmented Reality using Rotatable Video Projectors. *Proc. of IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*, 26-35.

Underkoffler, J., Ullmer, B. & Ishii, H. (1999). Emancipated pixels: real-world graphics in the luminous room. *Proc. of ACM Siggraph*, 385-392.

Levoy, M., Chen, B., Vaish, V., Horowitz, M., McDowall, I., and Bolas, M. (2004) Synthetic Aperture Confocal Imagining, Proc. of ACM Siggraph'04, pp. 825-834.

Sen, P., Chen, B., Garg, G., Marschner, S.R., Horowitz, M., Levoy, M., and Lensch, H.P.A (2005)., Dual Photography, *Proc. of ACM Siggraph*, pp. 745-755

Ramasubramanian, M., Pattanaik, S.N., and Greenberg, D.P. (1999), A perceptually based physical error metric for realistic image synthesis, *Proc. of ACM Siggraph '99*, pp. 73–82.



Bimber, O. & Raskar, R.
Spatial Augmented Reality: Merging Real and Virtual Worlds. A K Peters LTD (publisher), ISBN: 1-56881-230-2.

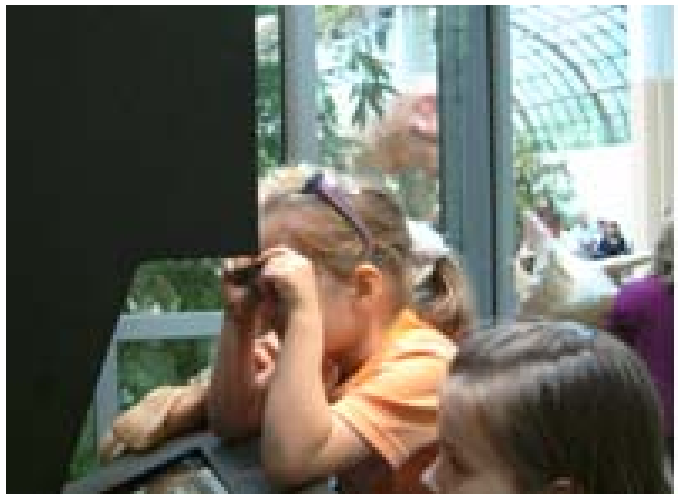
Thank you!

www.uni.weimar.de/medien/AR

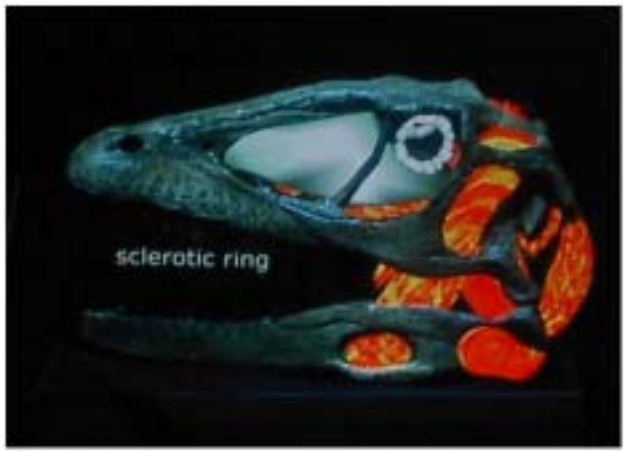
This project is supported by the Deutsche Forschungsgemeinschaft (DFG) under contract number PE 1183/1-1.

Part D:
Prototypes and Experiences

Virtual Showcase



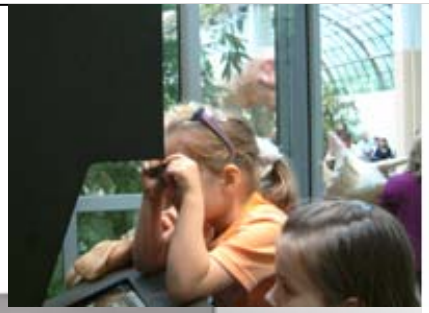
Special thanks to
Senckenberg Museum
Frankfut



Special thanks to
Fraunhofer IGD, Fraunhofer CRCG, and Lawrence Witmer, Ohio University

Bimber, et al. Merging Fossil Specimens with Computer-Generated Information." IEEE Computer 35:9, pp. 45-50, 2002.

Virtual Showcase



Since 2001:

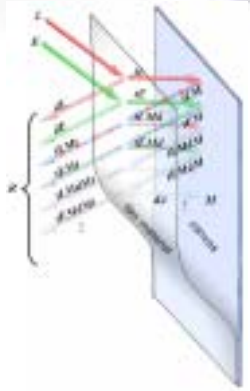
more than 11 exhibitions
more than 120,000 users

4 months unattended running time

Bimber, et al. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55,2001.

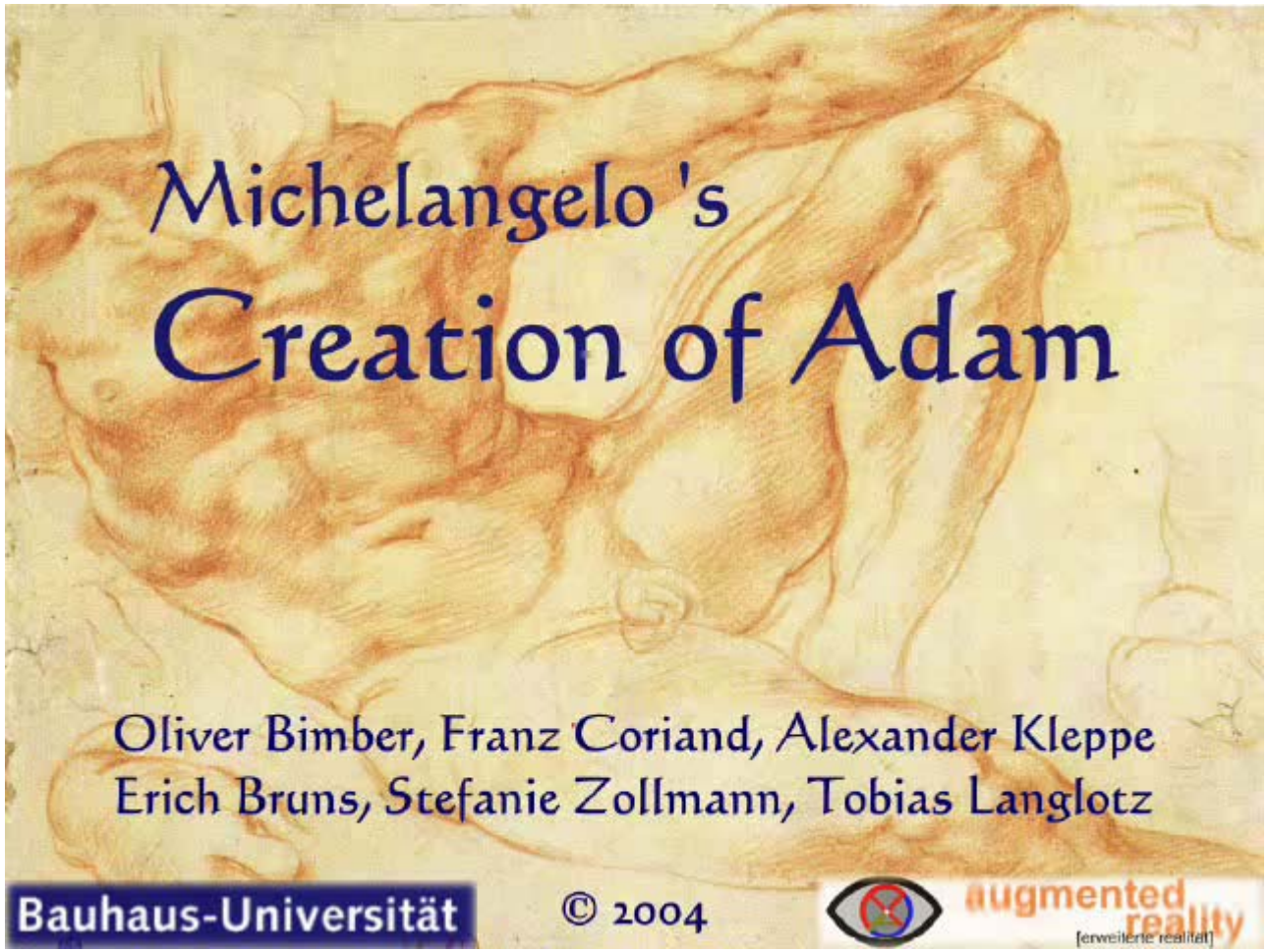
Special thanks to
Fraunhofer IGD, Fraunhofer CRCG, and
Lawrence Witmer, Ohio University

Augmenting Paintings



Bimber, e al. Superimposing Pictorial Artwork with Projected Imagery, IEEE MultiMedia, pp. 16-26, 2005

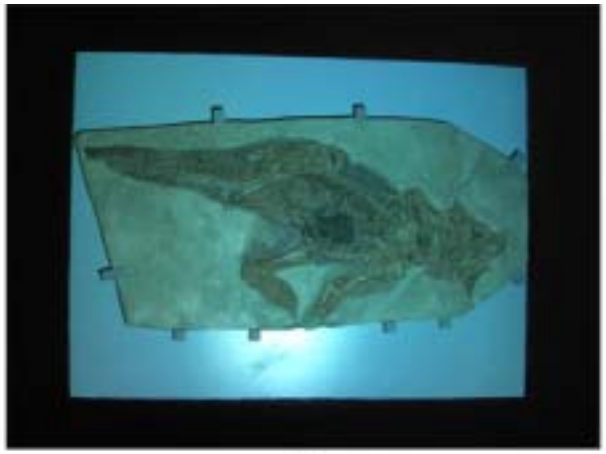
Augmenting Paintings

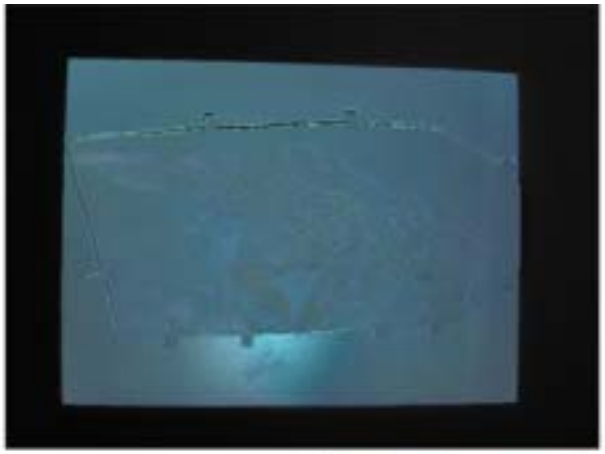


Bimber, e al. Superimposing Pictorial Artwork with Projected Imagery, IEEE MultiMedia, pp. 16-26, 2005

Special thanks to The Vatican Museum and the London National Gallery

Augmenting Fossils

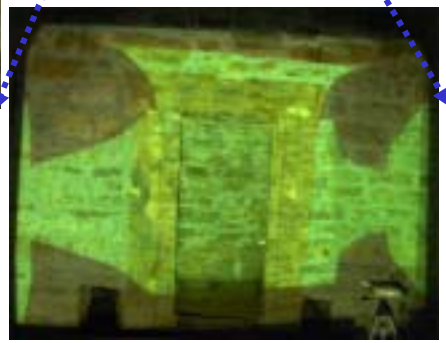


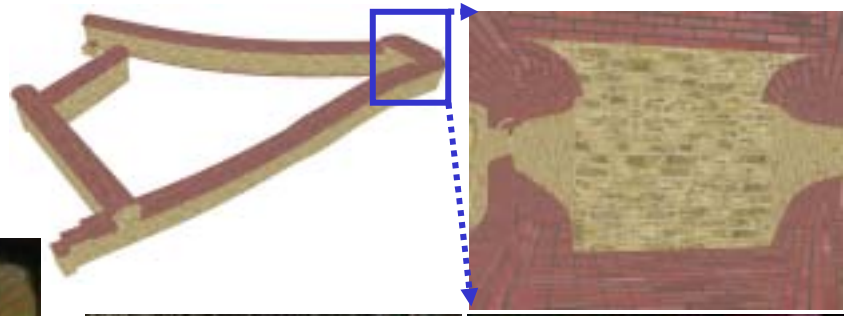


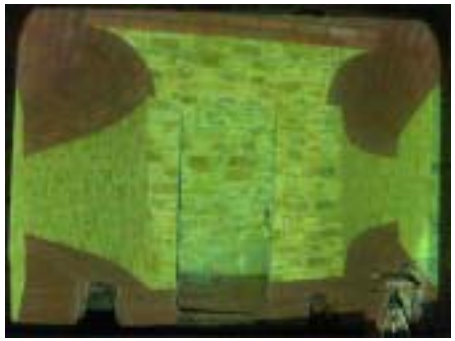
Special thanks to the Senckenberg Museum, Frankfurt

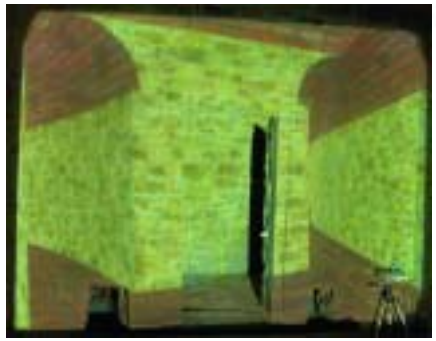
Augmentation of Ancient Water Reservoir

In cooperation with the Bennert Group



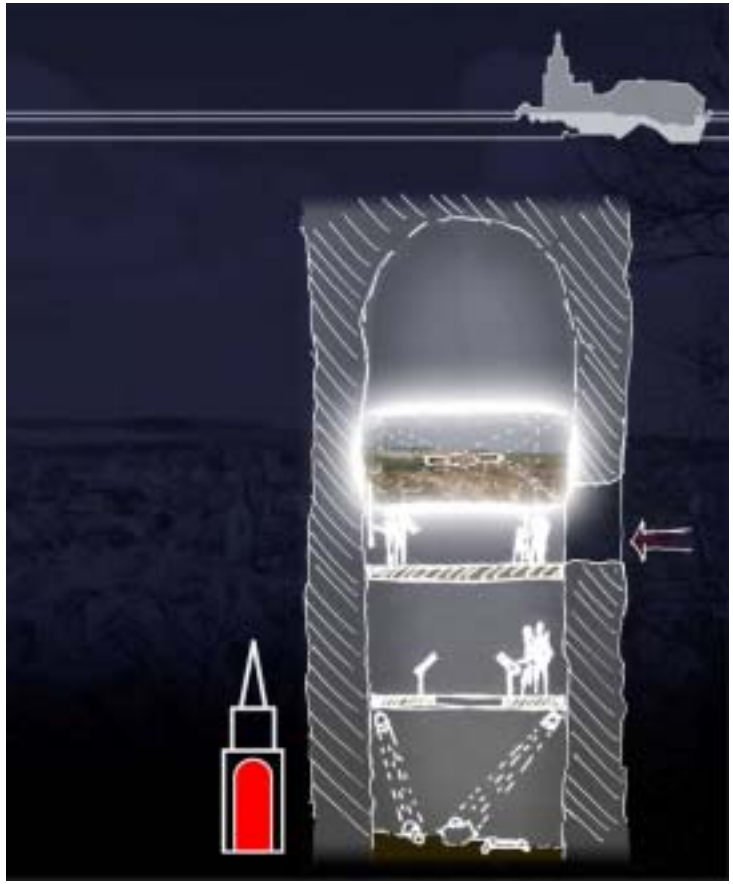






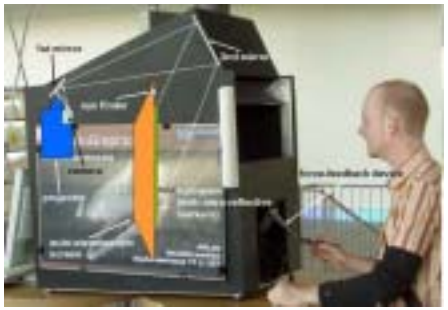
Augmentation of Ancient Water Reservoir

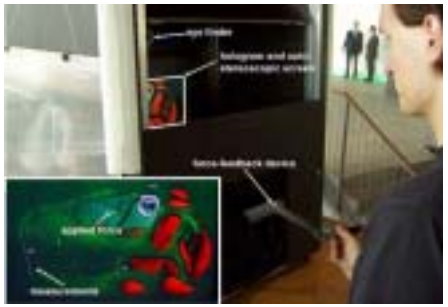
Cylindrical Surround View

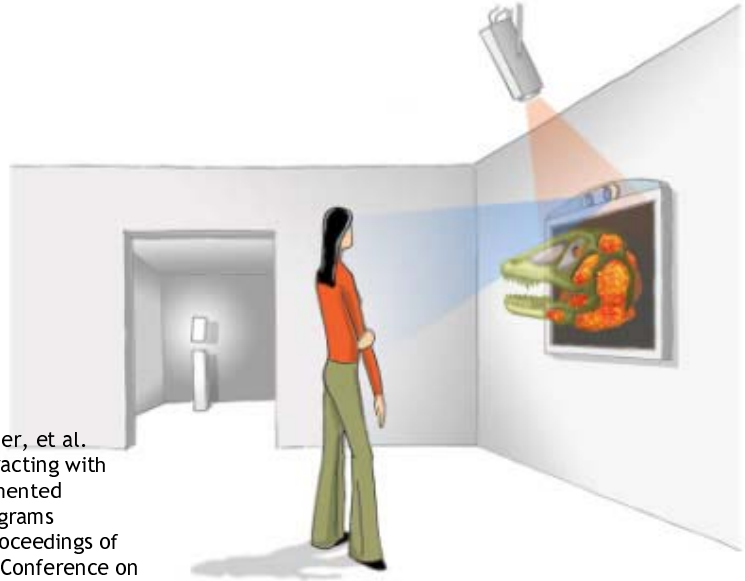


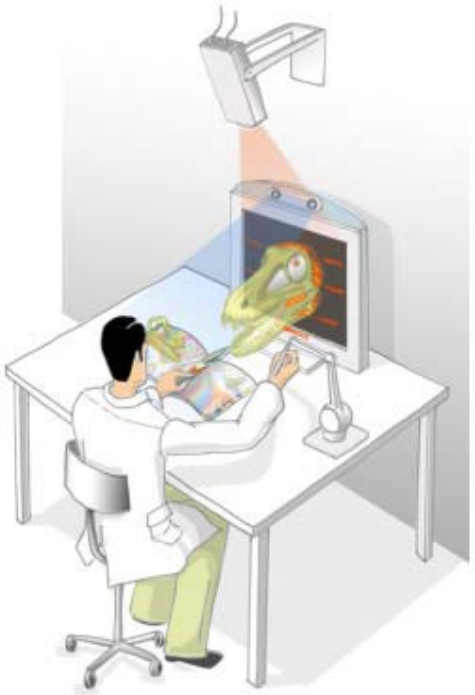
In cooperation with the
Bennert Group

HoloGraphics

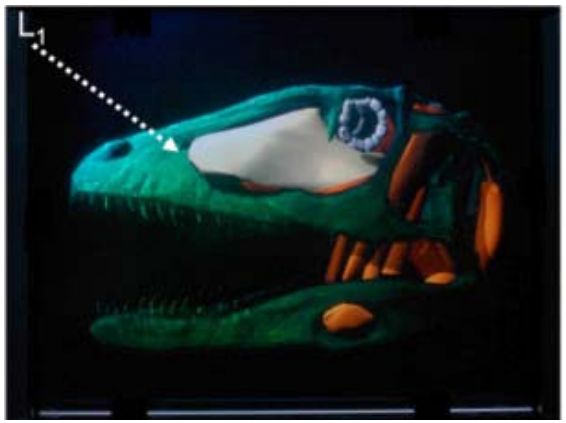


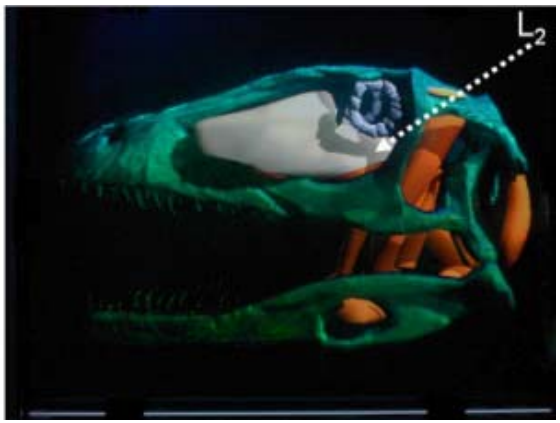






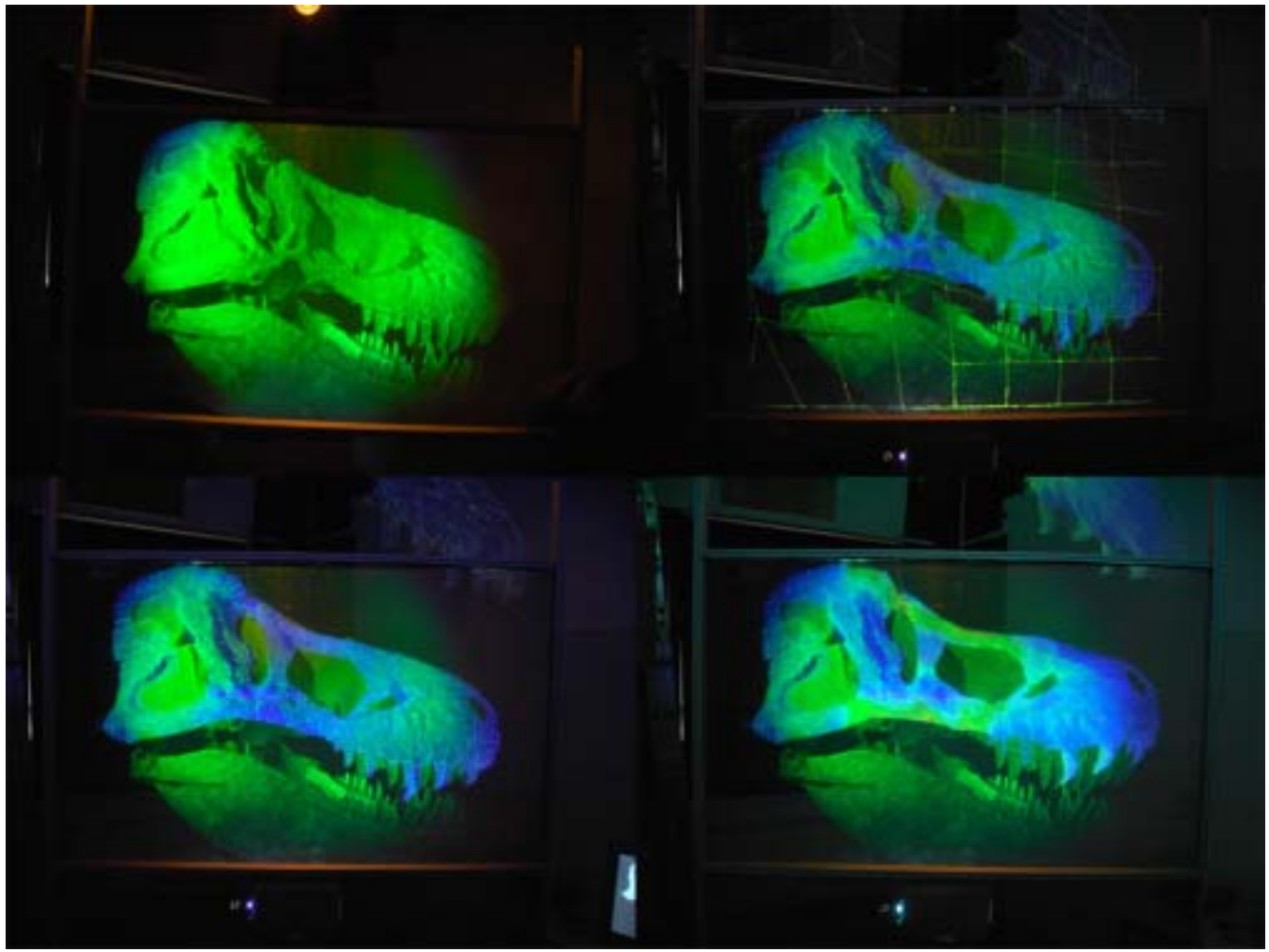
Bimber, et al.
Interacting with Augmented Holograms
In proceedings of SPIE Conference on Practical Holography XIX: Materials and Applications, January 2005





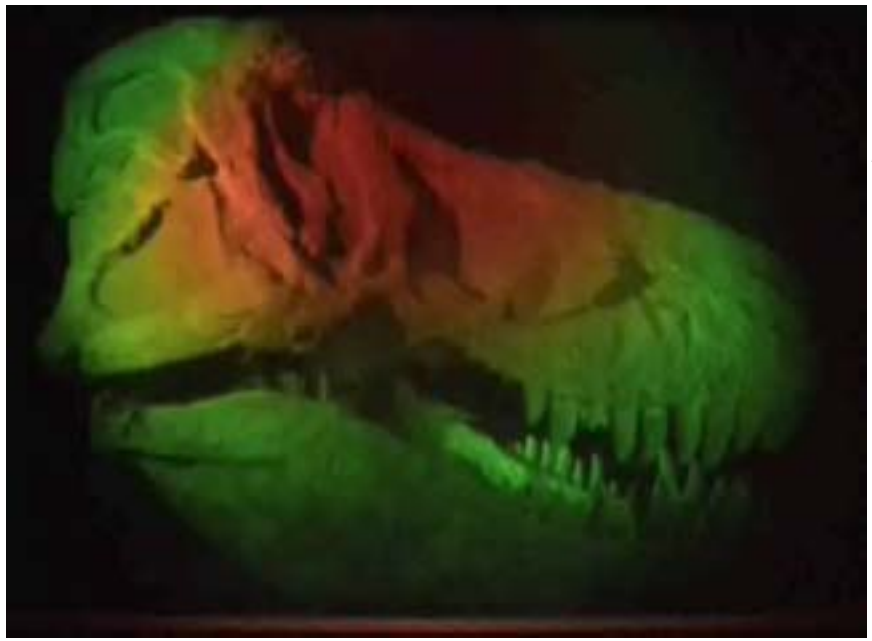
Bimber
Combining Optical Holograms with Interactive Computer Graphics
IEEE Computer (Cover Featuer), pp. 85-91, January 2004

HoloGraphics

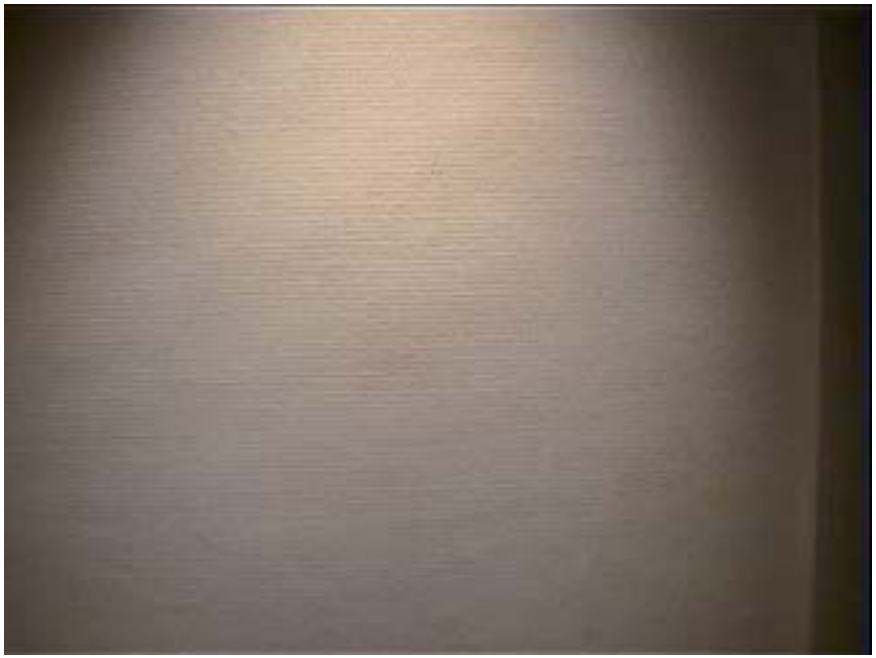


Special thanks to Emily J. Rayfield, Natural History Museum London

HoloGraphics



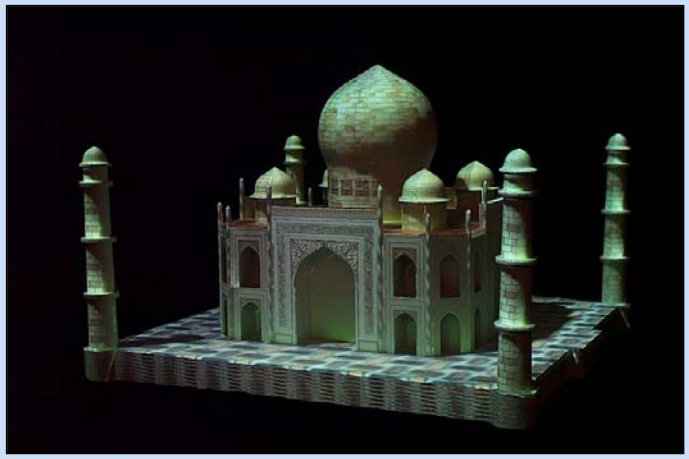
Special thanks to
Emily J. Rayfield,
Natural History
Museum London

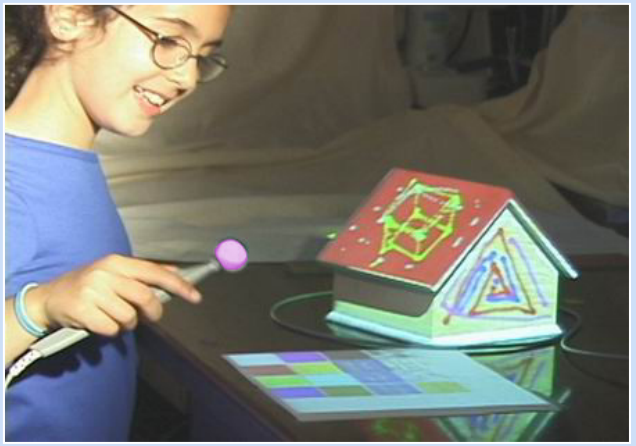


Special thanks to
Lawrence Witmer,
Ohio University

Prototypes: Spatial Augmentation

- More Examples ..





More info : www.ShaderLamps.com , Code available

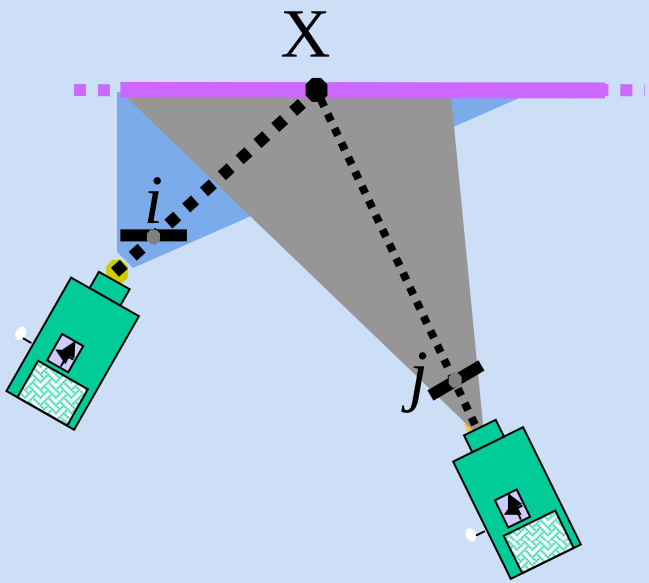
Quardic curved shape Displays

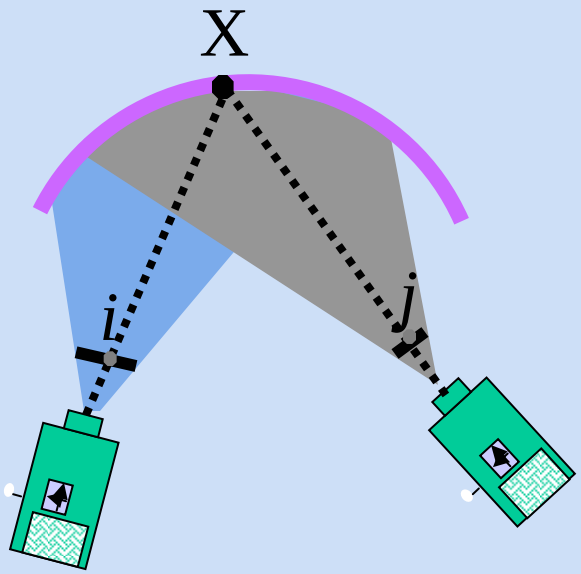
Planetarium

Sim/Viz Center

Raskar, vanBaar, Willwacher, Rao
‘Quadric Transfer for Immersive Curved Displays’,
EuroGraphics 2004

Parametric Image Transfer





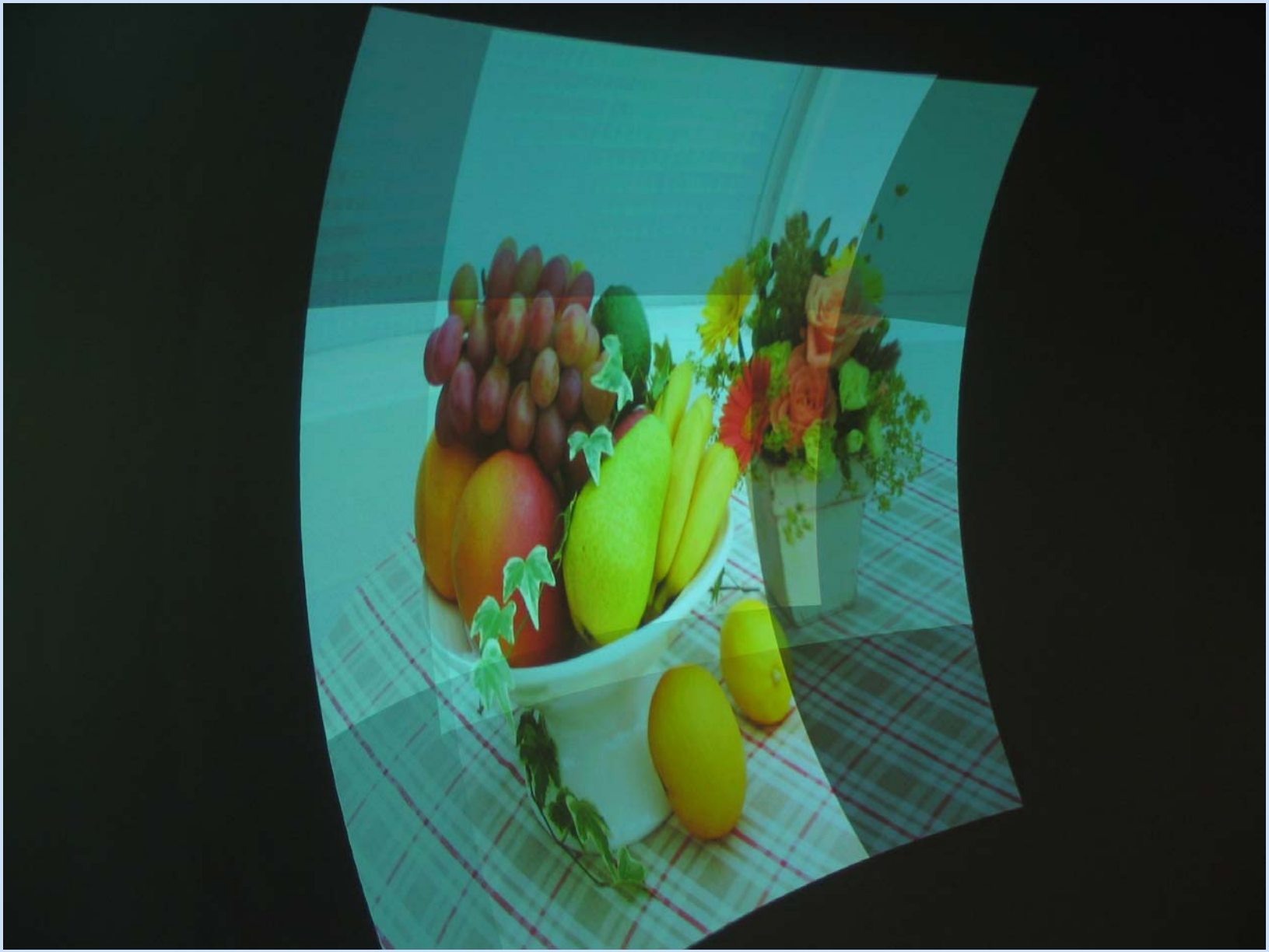
Planar Homography

$$j \cong A_{3x3}\, i$$

Quadric Transfer

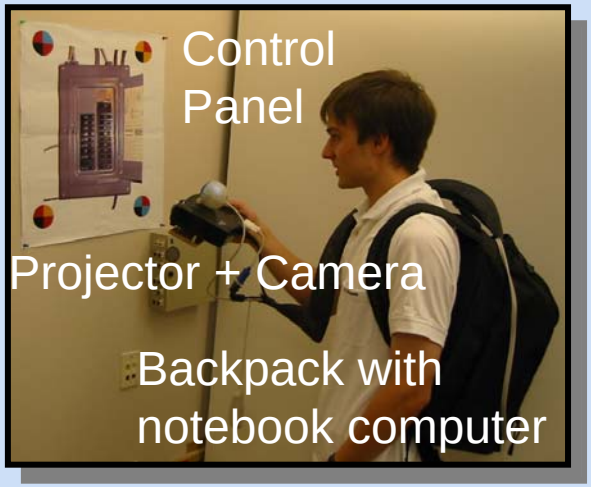
$$j \cong A_{3x3}\, i \pm \left(\sqrt{i^T E\, i}\right) e$$

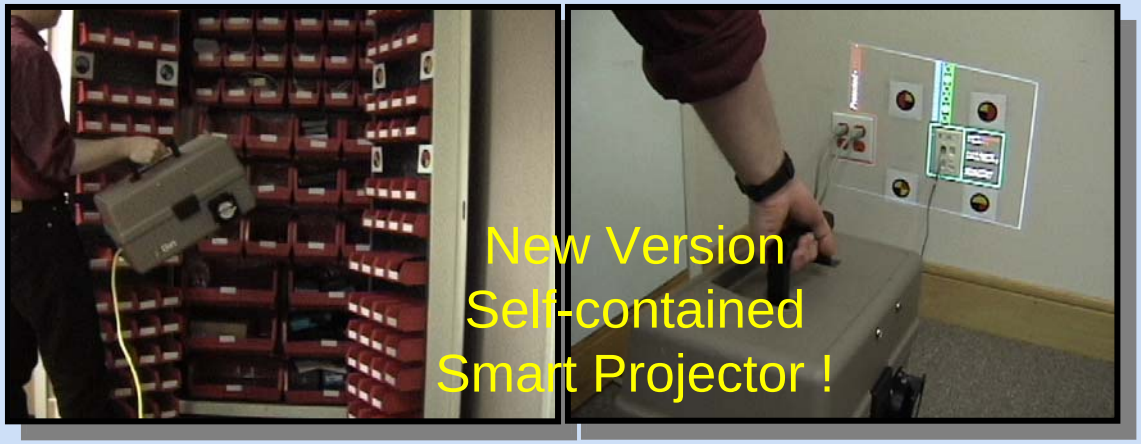
Overlap on Quadric Screens



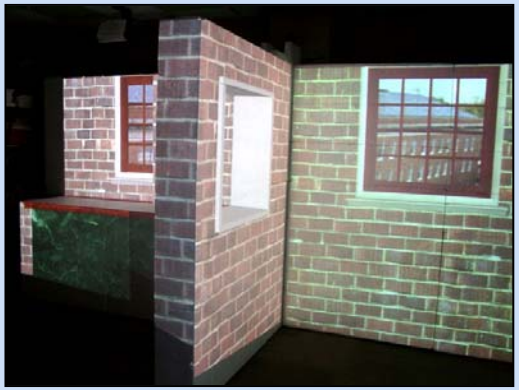
Training and Maintenance
(Projector-based Augmented Reality)
Raskar, Beardsley, Forlines

- Automatically add projected information
  - Training videos
  - Instruction manuals
- Detect pose and identity from pie-codes

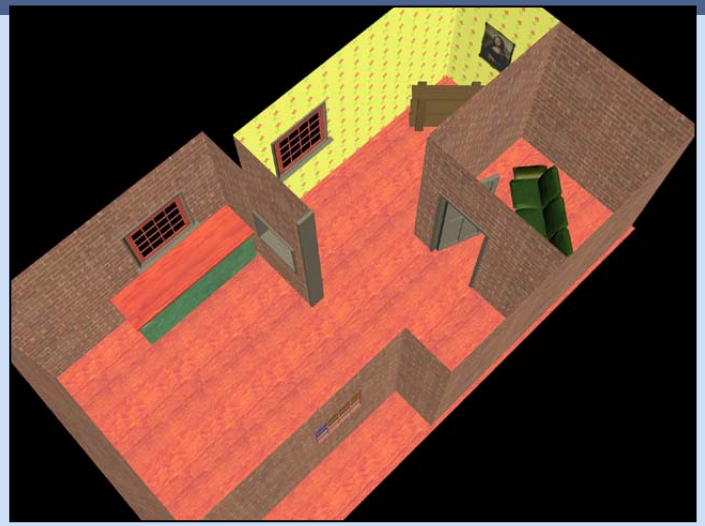




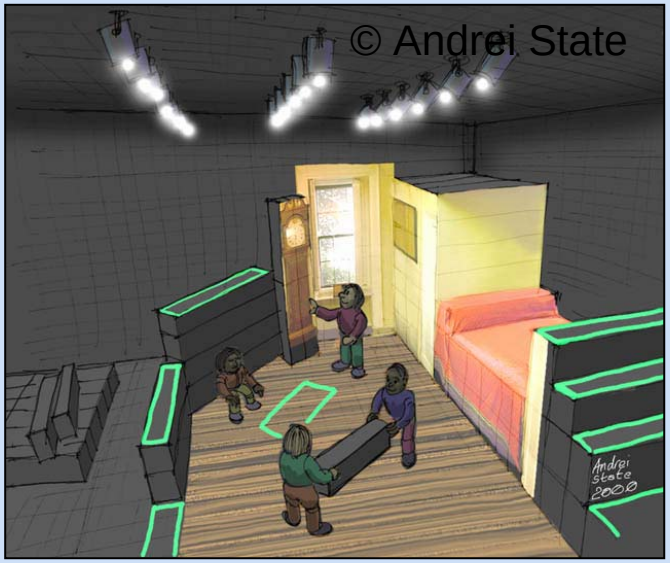
- Recreate Large Environments
  - ‘BeingThere’, walk-around
  - Human sized environments
  - Museums, Exhibitions



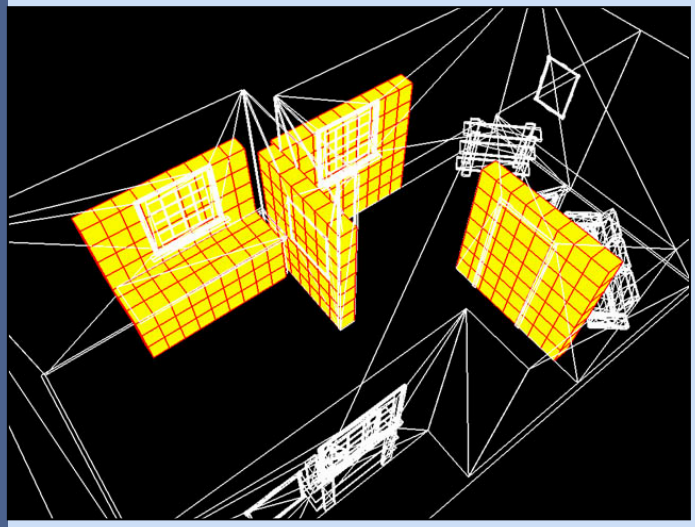
Kok-Lim Low, Greg Welch, Anselmo Lastra, Henry Fuchs. “Life-Sized Projector-Based Dioramas,” Proc. ACM Symposium on Virtual Reality Software and Technology 2001 (VRST 2001), November 2001.

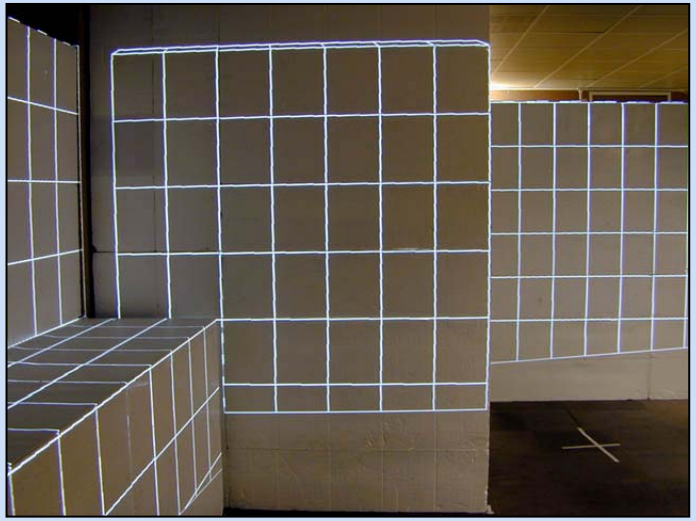


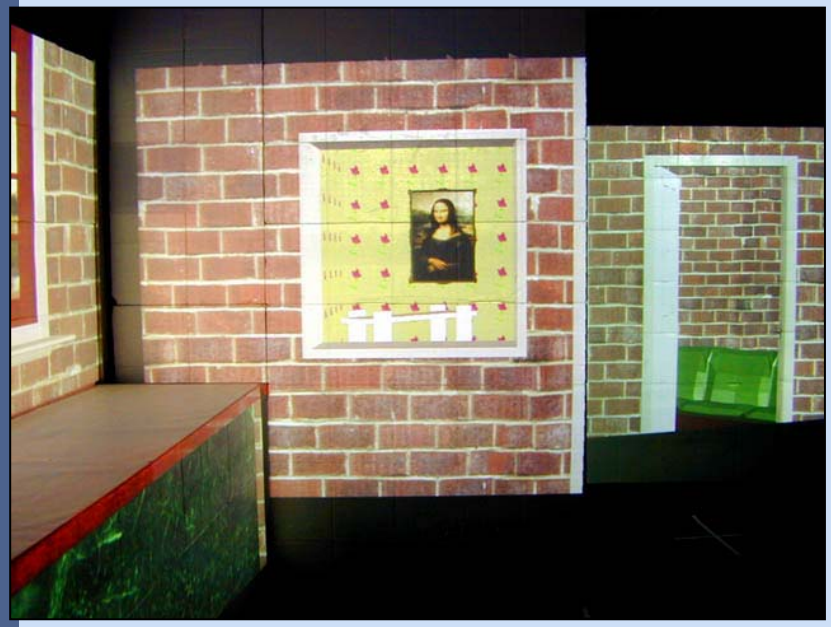
Desired Virtual Model

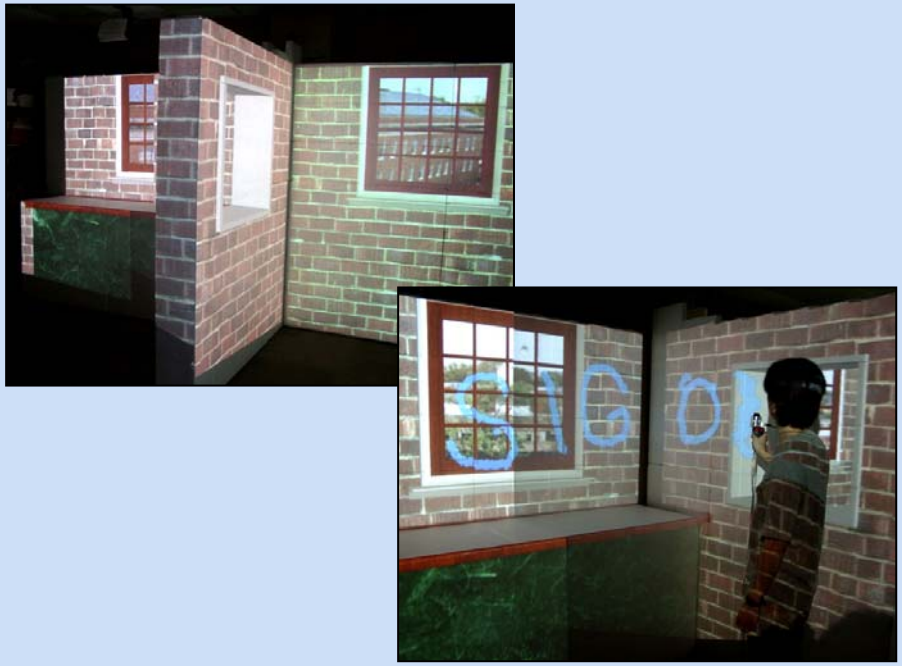


Projected Guidance for Placement

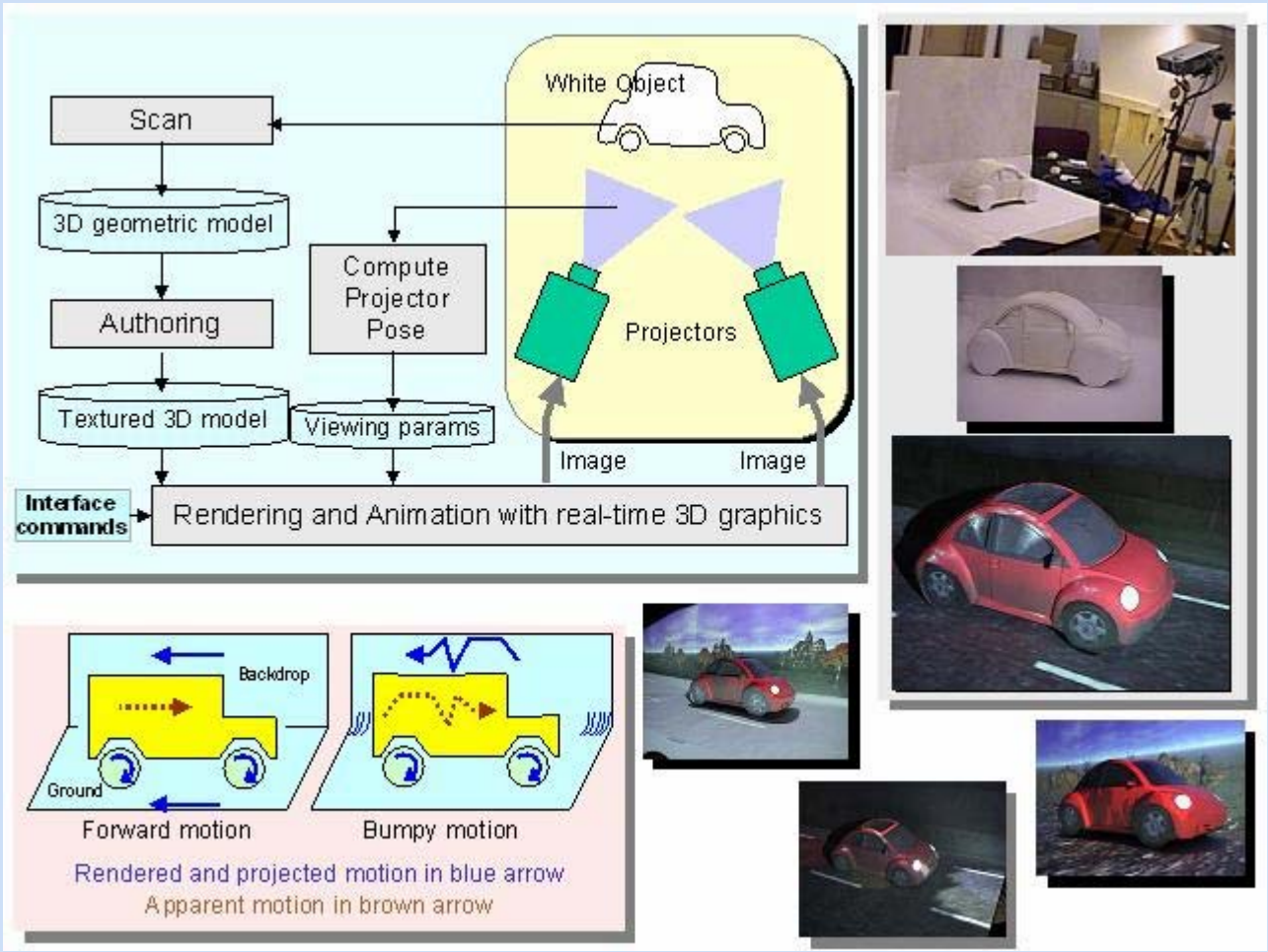






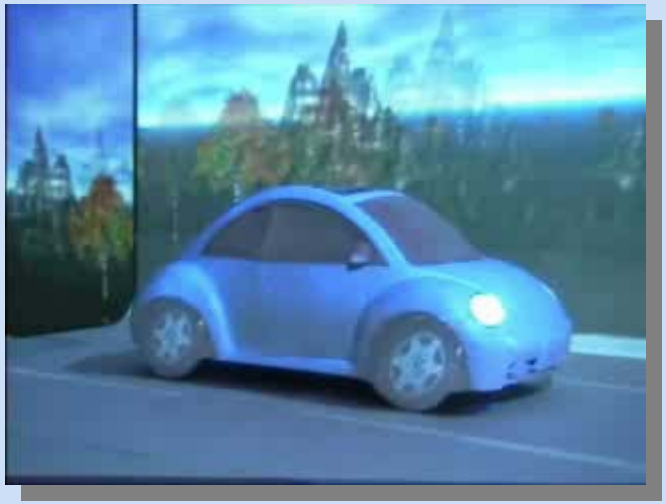


Apparent Motion



Ramesh Raskar, Remo Ziegler, Thomas Willwacher, “Cartoon Dioramas in Motion,” Proc. ACM Symposium on Nonphotorealistic Animation and Rendering (NPAR 2002)

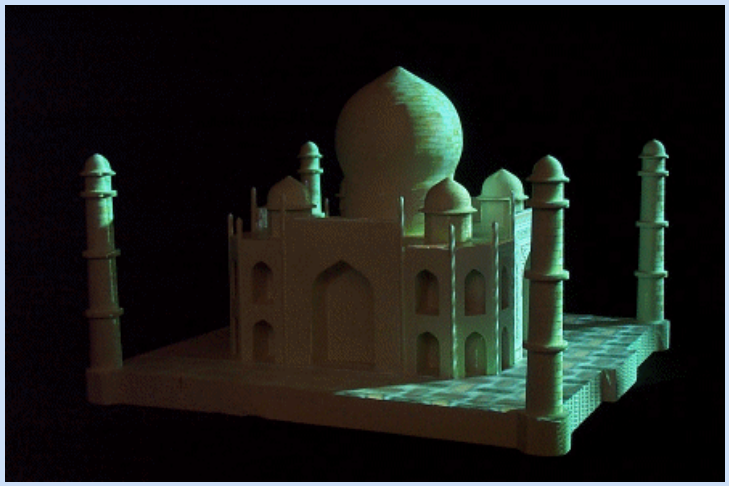
Virtual Motion



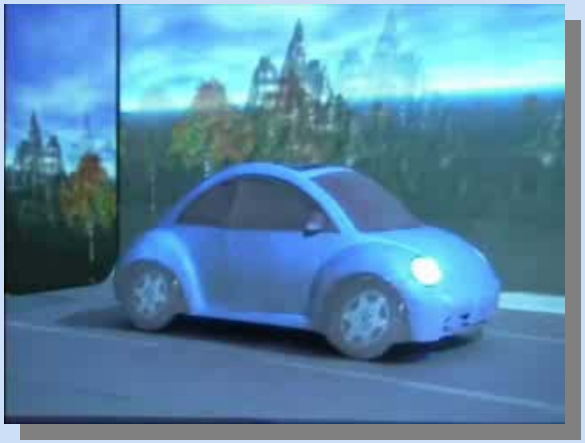
ShaderLamps

Virtual Reflectance

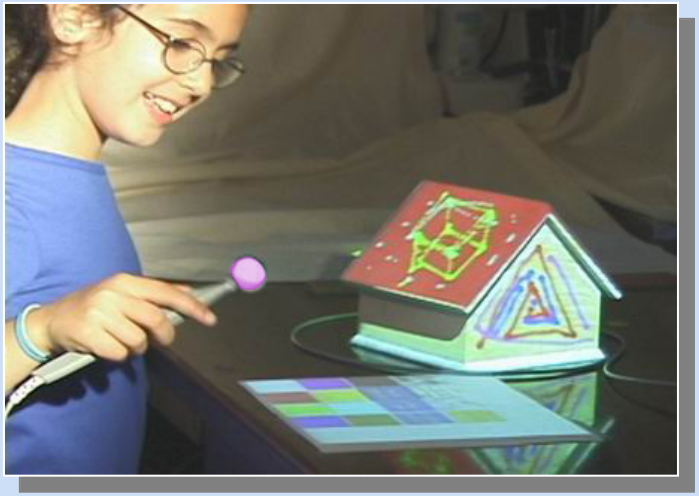
Virtual Illumination



Virtual Motion



Interaction



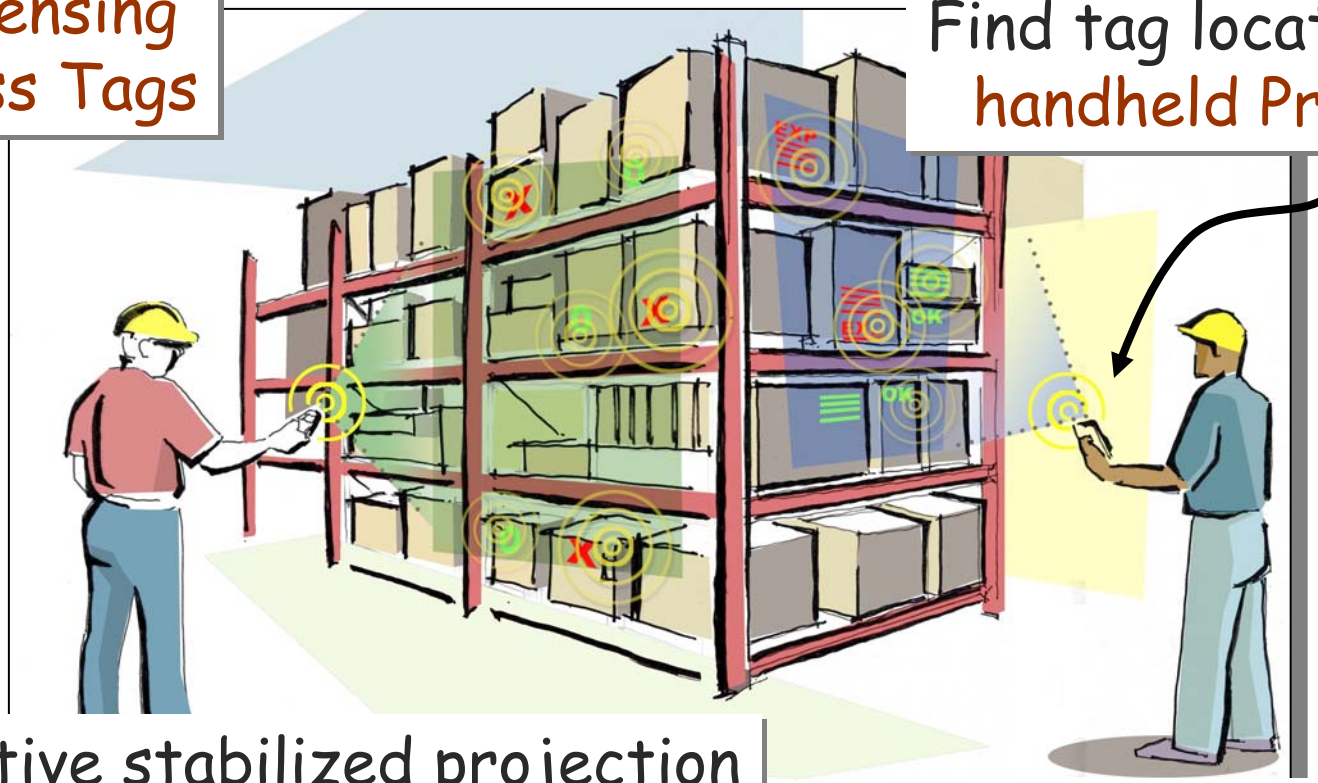
www.ShaderLamps.com

RFID ➡ RFIG

(Radio Frequency Id & Geometry)

Photosensing Wireless Tags

Find tag location using handheld Projector



Interactive stabilized projection

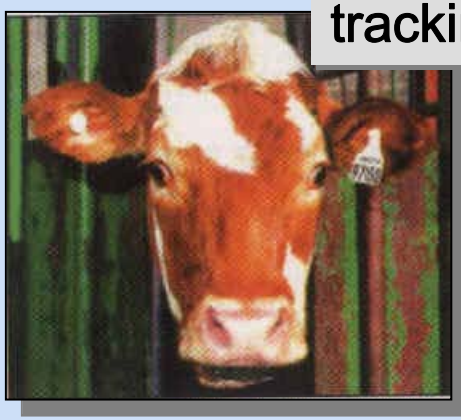
Many geometric ops

Siggraph 2004

Warehousing

Routing

Livestock tracking

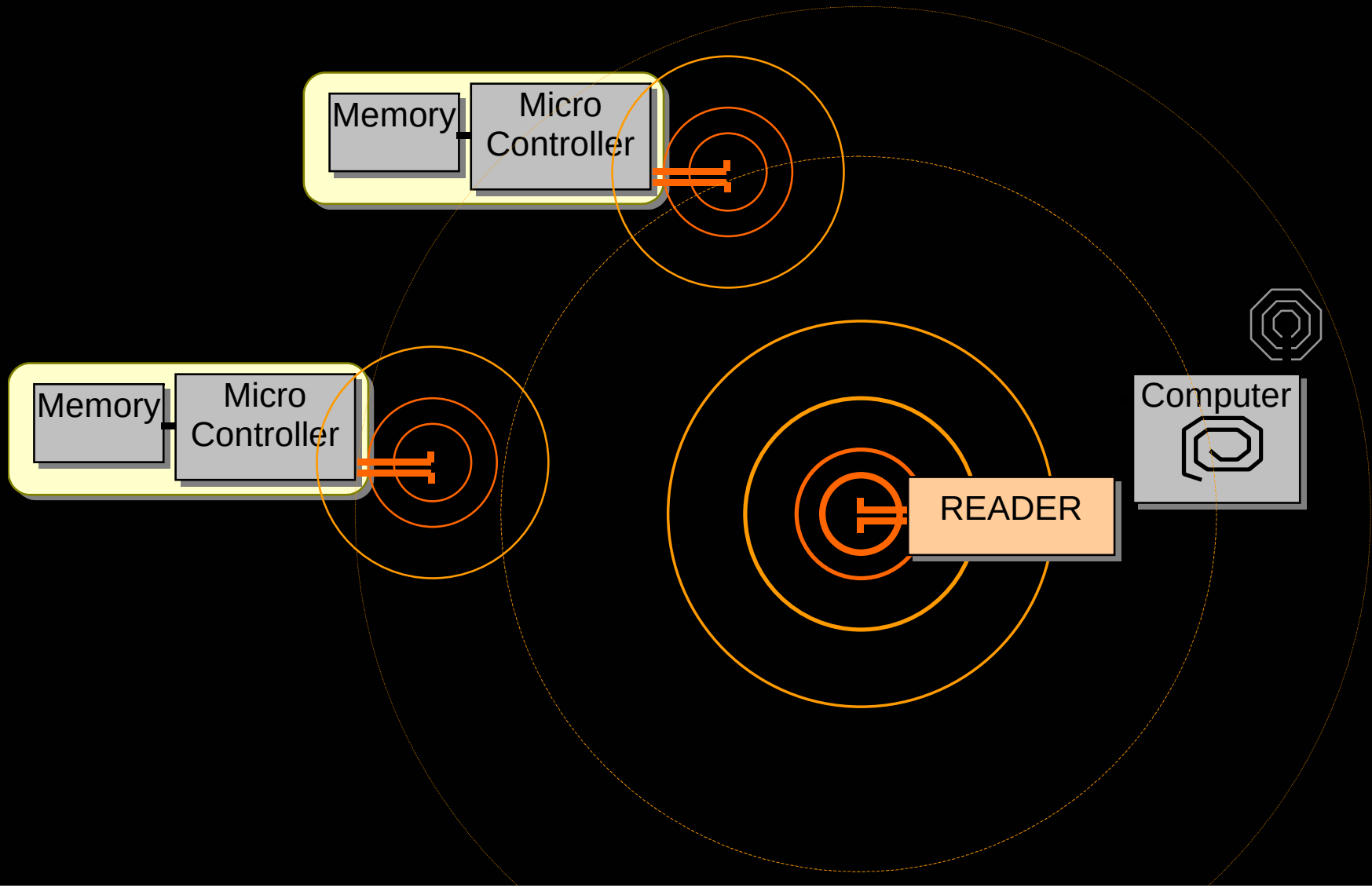


Library

Baggage handling

Currency

Conventional Passive RFID



Tagged Books in a Library

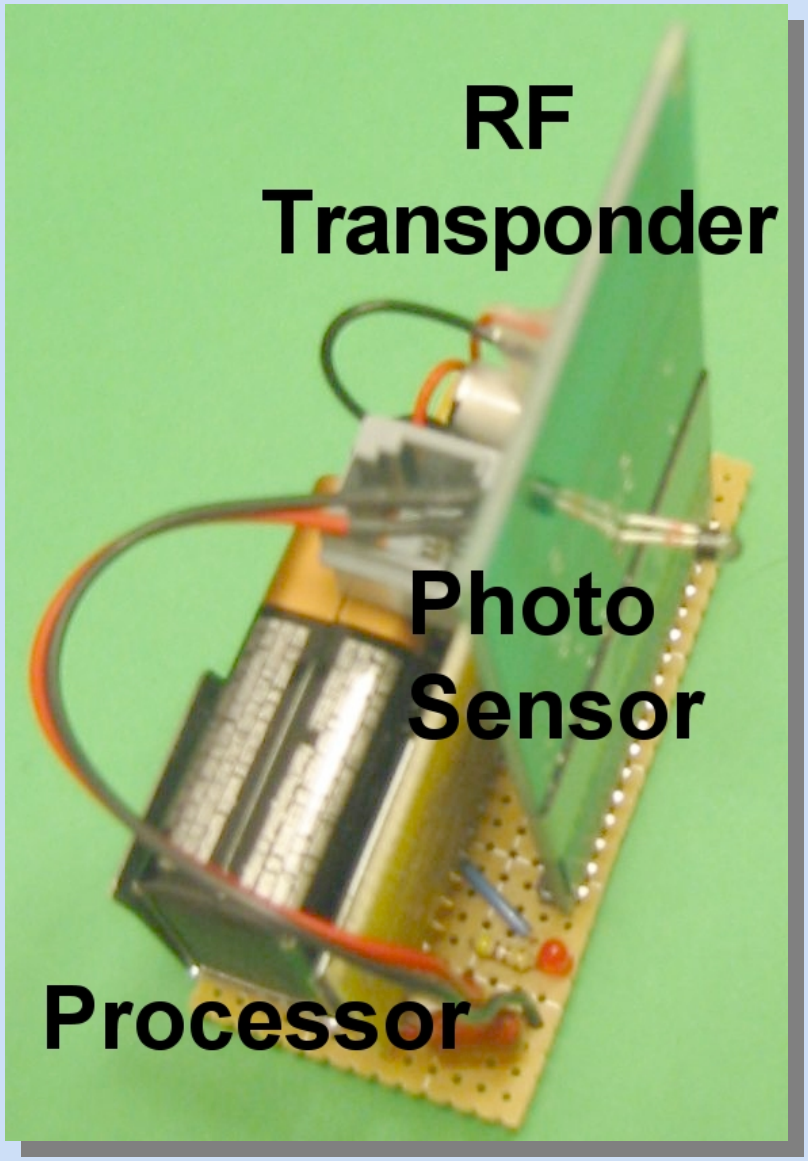
✓ **Id** : List of books in RF range

✘ **No Precise Location Data**

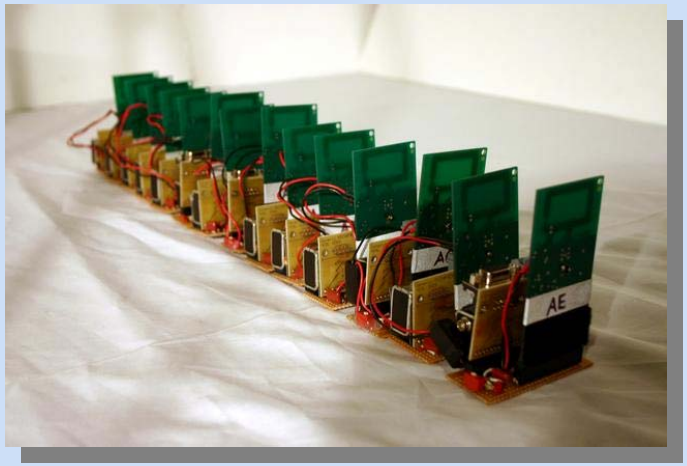
Are books in sorted order ?

Which book is upside down ?

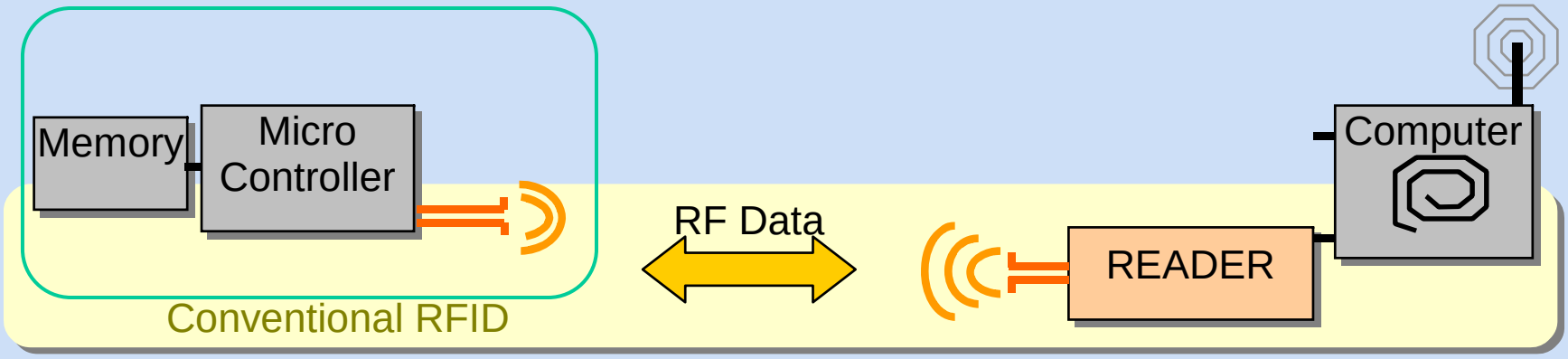
Prototype Tag



RF tag +
photosensor



Conventional RF tag



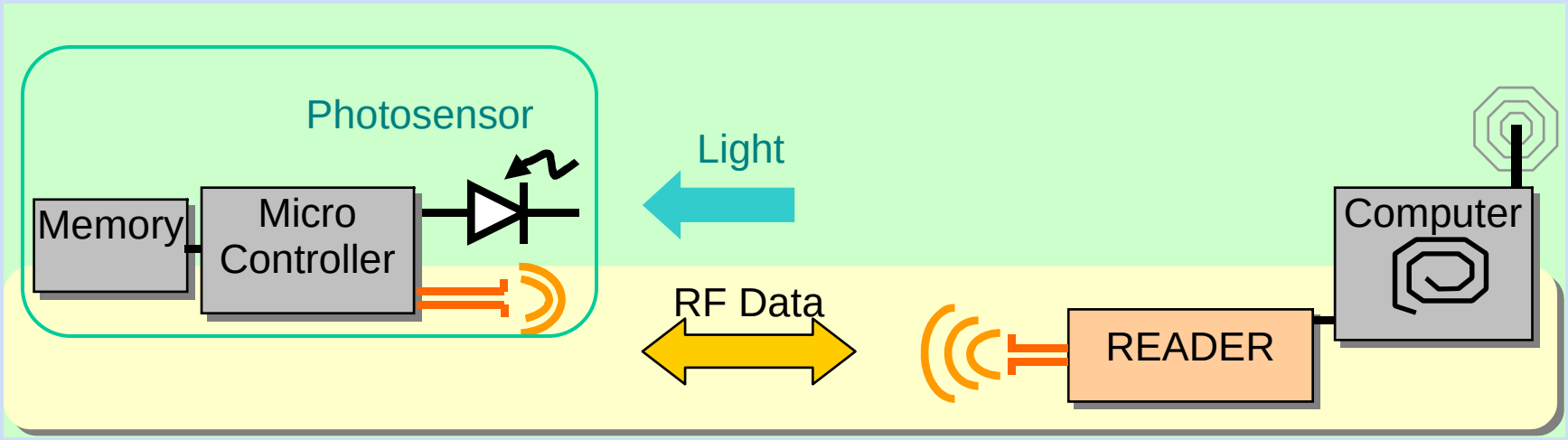
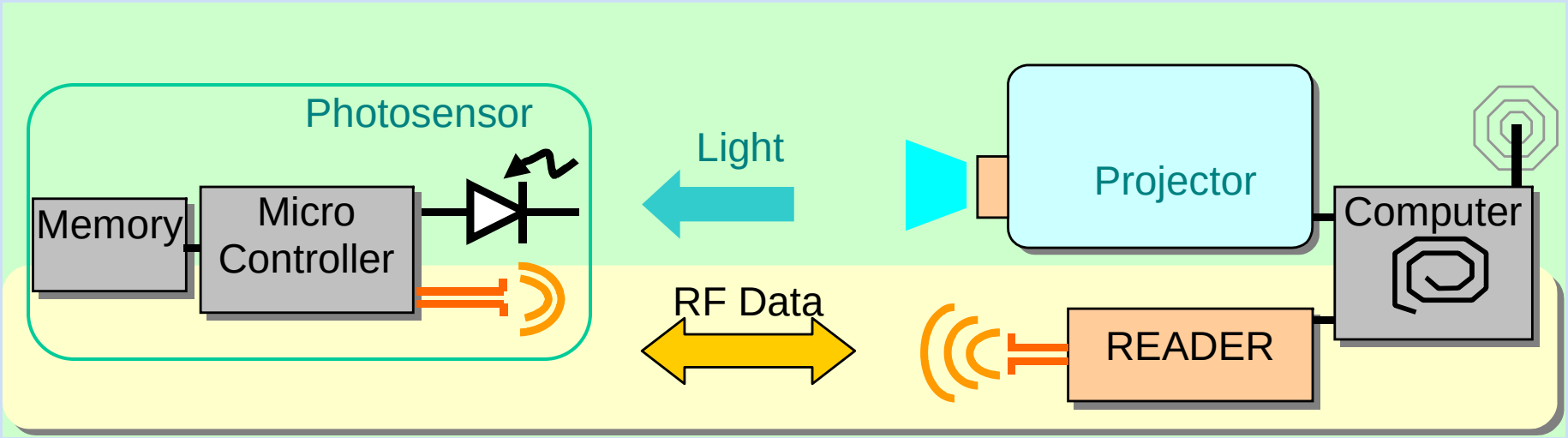


Photo-sensing RF tag

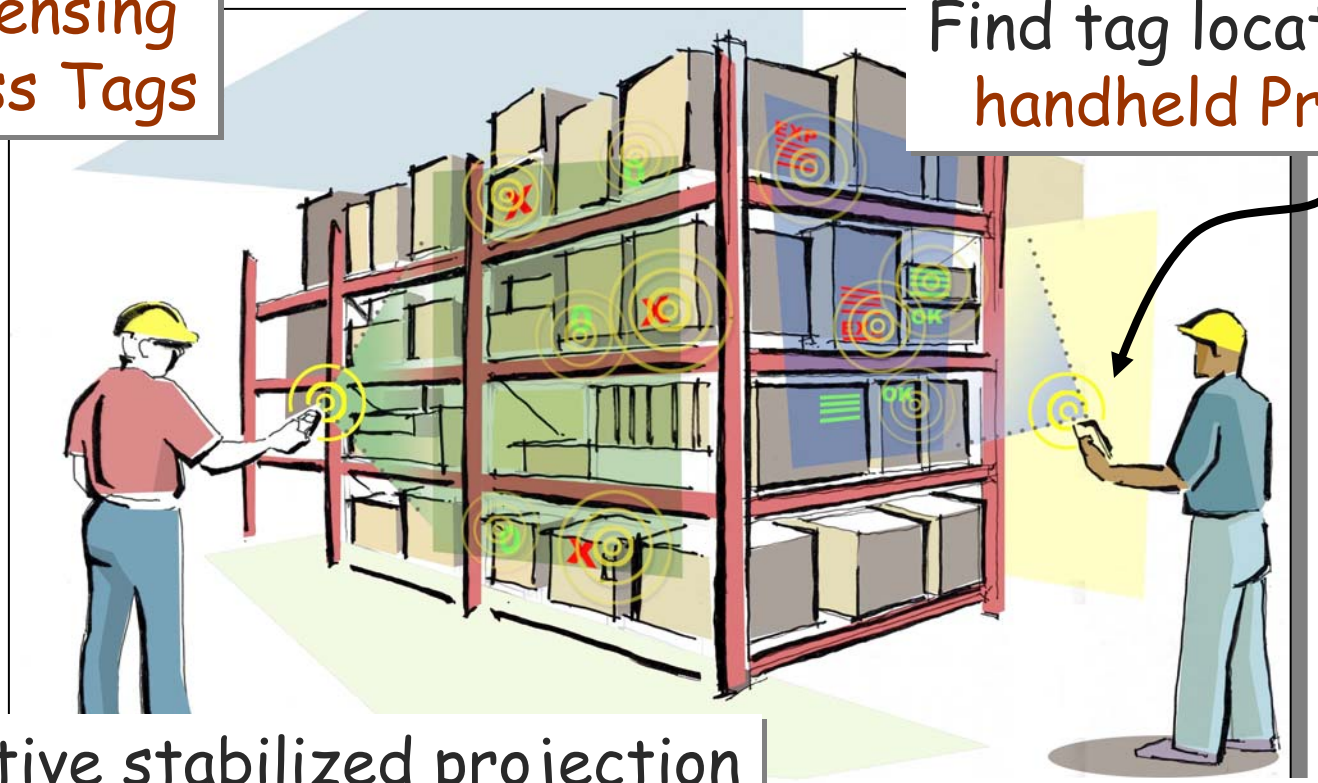

Photosensor
Memory
Micro Controller
Light
RF Data
Projector
Computer
READER

RFID ➡ RFIG

(Radio Frequency Id & Geometry)

Photosensing Wireless Tags

Find tag location using handheld Projector

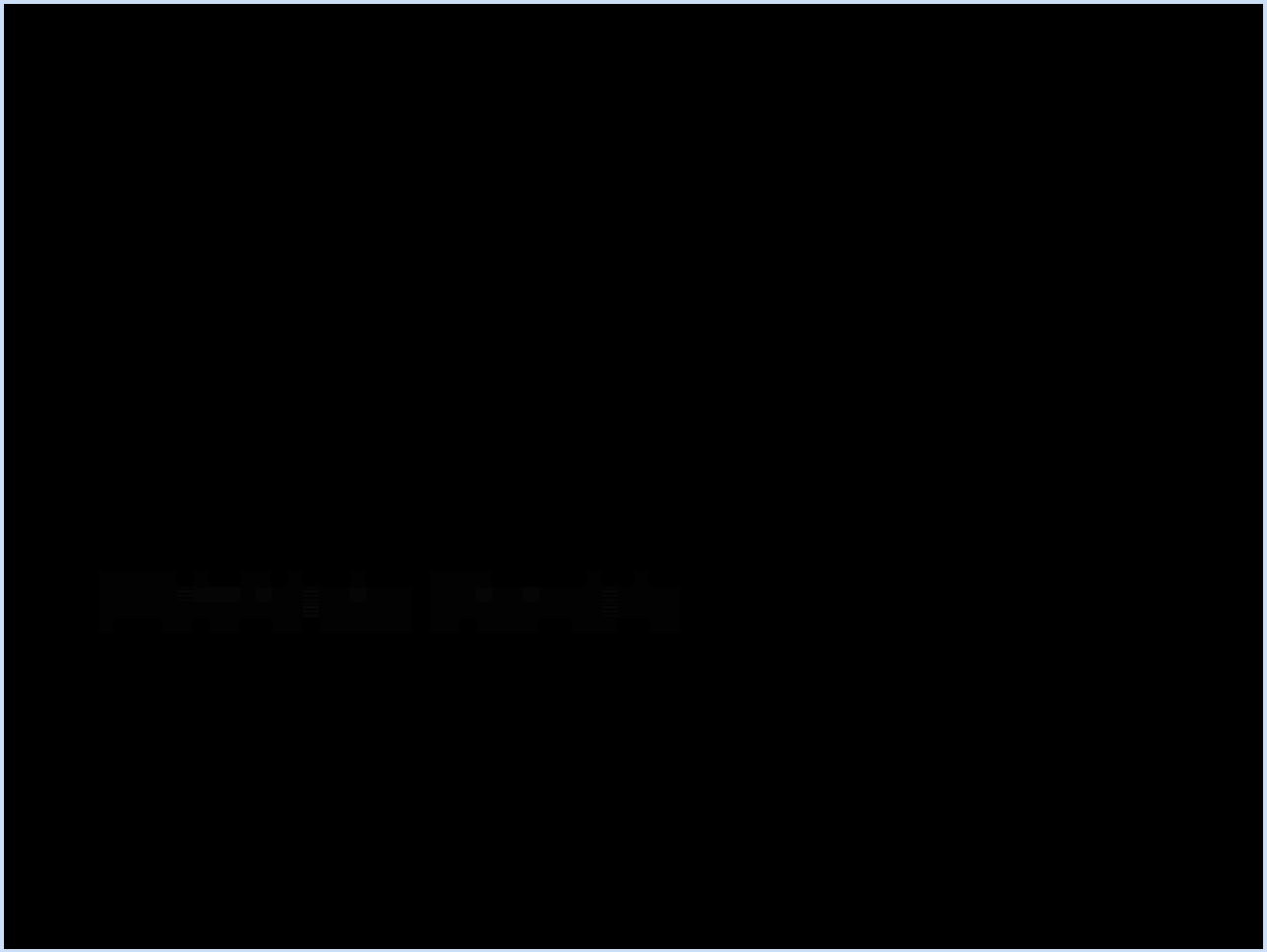


Interactive stabilized projection

Many geometric ops

Siggraph 2004

AR with Photosensing RFID and Handheld Projector



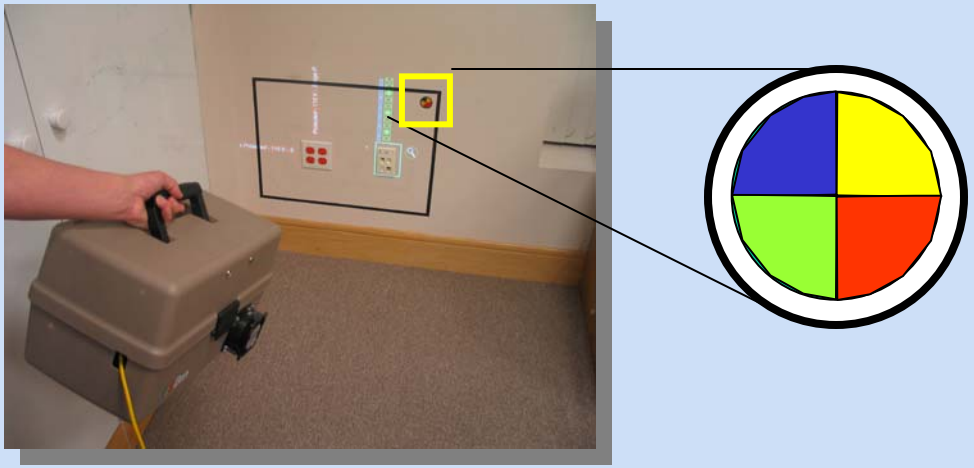
RFID

(Radio Frequency Identification)

RFIG

(Radio Frequency Id and Geometry)

AR Issues



- Preprocessing:
  - Authoring
- Runtime:
  - **Identification:** Recognition of objects
    - Using markers and visual tags
  - **Registration:** Finding relative pose of display device
    - Dynamic estimate of translation and rotation
    - Render/Warp images
  - **Interaction:**
    - Widgets, Gesture recognition, Visual feedback

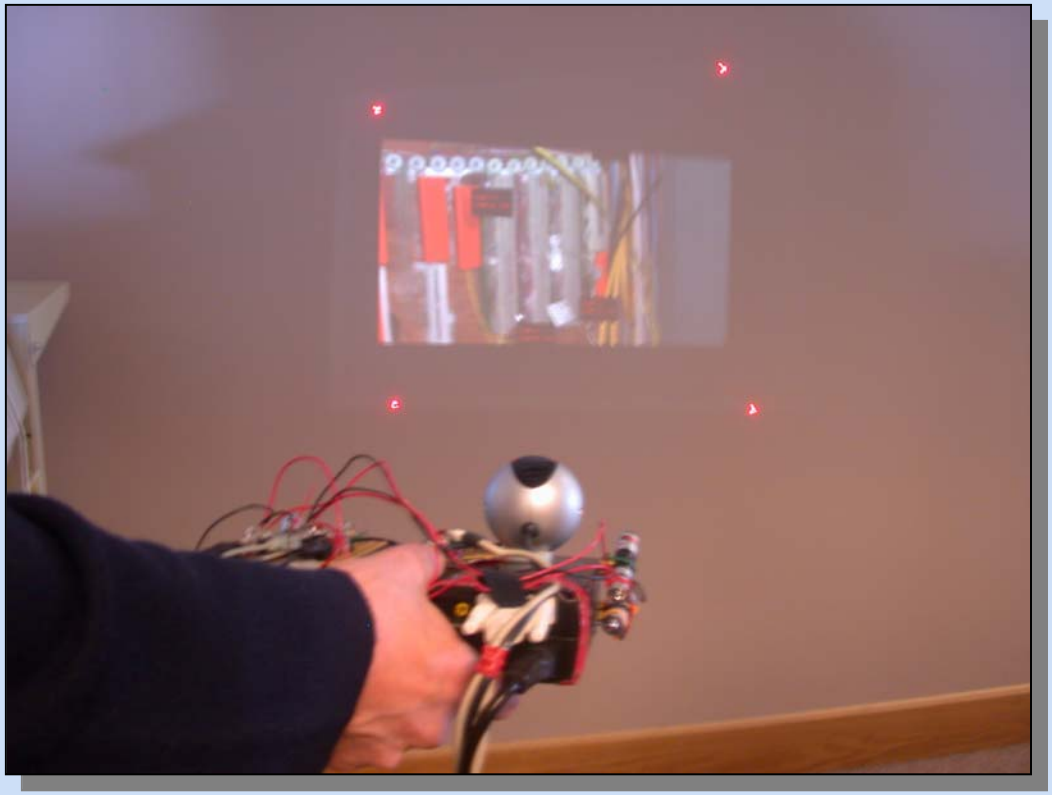
Visual feedback of 2D position

a. Receive via RF {(x_1,y_1), (x_2,y_2), …} pixels

b. Illuminate those positions

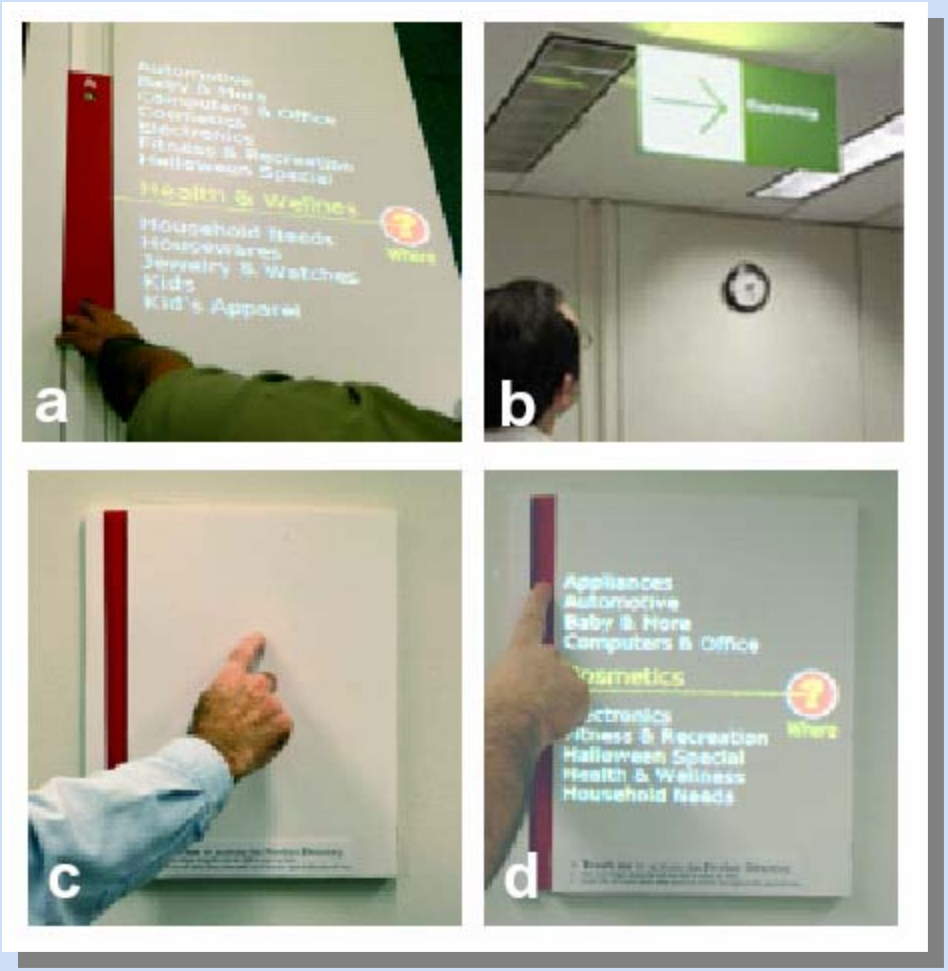
Interactive Projection

Adaptive Projection



‘Copy and Paste’
Geometric and Photometric compensation

Steerable Projector



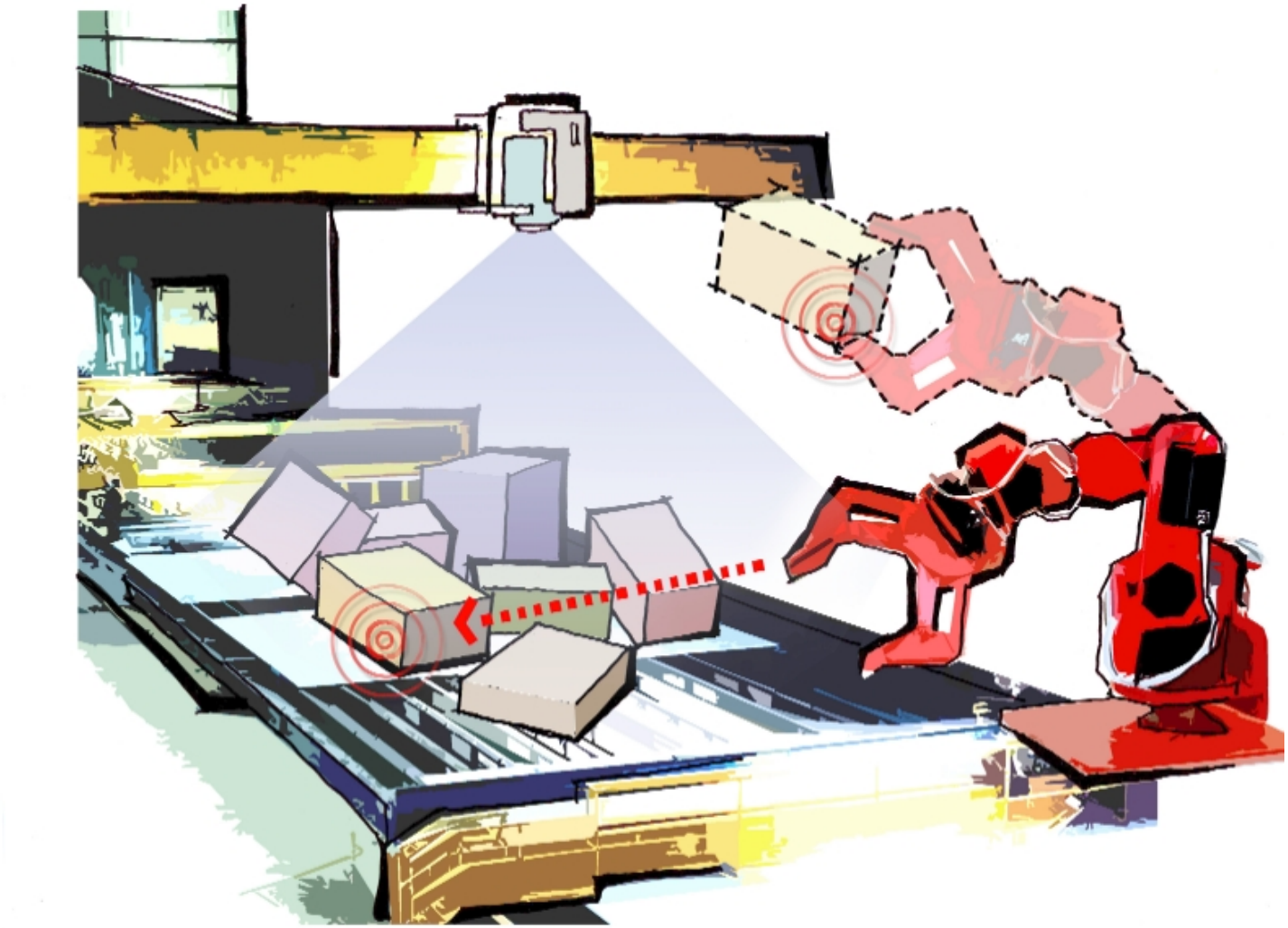
Pinhanez et al 2003

Machine AR

- AR for cameras and machines

- Face Dome [Debevec 2001]
- 4D lighting [MPI, MERL]

Robot ‘Laser’ Guidance
Picking and Sorting Tagged Objects



Acknowledgements

- MERL
  - Jeroen van Baar, Paul Beardsley, Remo Ziegler, Thomas Willwacher, Srinivas Rao, Cliff Forlines, Paul Dietz, Joe Marks
- Office of the Future group at UNC Chapel Hill
  - Greg Welch, Kok-lim Low, Deepak B'padhyay, Aditi Majumder, Michael Brown, Ruigang Yang
  - Henry Fuchs, Herman Towles
  - Wei-chao Chen
- Mitsubishi Electric, Japan
  - Yoshihiro Ashizaki, Masatoshi Kameyama, Masato Ogata, Keiichi Shiotani
- Images
  - Marc Pollefeys (UNC Chapel Hill)
  - Apologies
    - (Not able to include recent work by others)

Goals

- Understand advantages of Spatial AR
- Discuss issues in traditional AR approaches
- Explore alternative AR methods
  - Graphics, Vision, Optics techniques
  - Learn math of rendering and calibration
- See new applications in art and industry

Opportunities

- Think beyond goggle-bound AR
- Learn techniques using projectors, flat displays and optics
- Explore more realistic augmented environments
- Learn how to build your own spatial AR displays (only covered in tutorial notes).
- Learn how to extend your own software framework to support spatial AR displays
- Get an impression on applicability

Discussion

- AR Display Approaches
  - Traditional, Goggle-bound, Alternative
- Spatial Augmentation
  - Projectors, Non-planar, Mobile objects, Change appearance, Interaction
- Exploiting Optical Elements
  - Mirrors, Beam splitters, Holograms
- Applications
  - Prototypes in Art, Research and Industry

Feedback

- Course website http://Spatial-AR.com
- Please complete the survey form

- Feedback form
  - What material would you like to see ?
  - Would you like more programming details ?
  - What are your future directions of AR work ?

Cartoon Dioramas in Motion

Ramesh Raskar Remo Ziegler Thomas Willwacher
MERL, Mitsubishi Electric Research Labs
{raskar, ziegler, thomasw}@merl.com

Abstract

Cartoon animations delight the audience with moving characters but they remain on a flat 2D screen. The cartoon dioramas, on the other hand, are detailed, three-dimensional and allow physical interaction but they are static. We present techniques to combine the two in some limited cases. We illuminate static physical models with projectors. The images are generated with real time three dimensional computer graphics. We describe a system to demonstrate various visual effects such as non-photorealistic shading, apparent motion and virtual lighting on a toy-car model.

CR Categories: I.3.7 [Computing Methodologies]: Computer Graphics—3D Graphics

Keywords: perception, immersive environments, virtual reality, augmented reality, non-photorealistic rendering.

1 Introduction

Cartoon animations are made from a series of drawings simulating motion by means of small progressive changes in the drawings. The gross movement is achieved using relative displacement between layers of characters and backgrounds. The stored animation then can be seen at a desired frame rate as a video. Life-sized or scaled versions of cartoon characters and environments, such as funny shaped vehicles and buildings are also seen in shops, amusements parks and design studios. Inspired by recent advances in augmented reality, our goal is to combine the two mediums: two-dimensional cartoon animation and static three dimensional dioramas of physical models into lively non-realistic setups and non-realistic movement. We substitute pre-recorded sequence of images with real-time three dimensional computer graphics renderings to create an animated physical world. In this paper we describe a system to show some limited effects on a static toy-car model and present techniques that can be used in similar setups. Our focus is on creating apparent motion for animation.

1.1 Motivation

Our approach is based on techniques in two very different fields, projector-based augmentation and illusions for creating apparent motion.

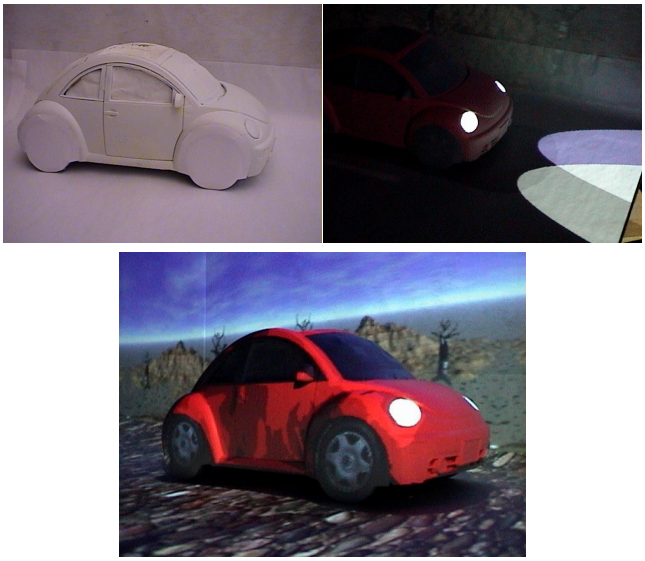


Figure 1: (Top-left) The diorama of a car and approximate background. (Bottom) A cartoon-like environment created by image projection in which the car appears to move (Top-right) A night-time simulation using virtual headlights.

1.1.1 Augmenting Physical Models

Projector can be used to change the surface appearance of neutral colored physical models by illuminating them with rendered images. The projectors are treated as 'Shader Lamps' [Raskar et al. 2001]. In this type of projector-based augmentation, complete illumination of complex 3D shapes is made possible by rendering geometric models of those shapes from multiple overlapping projectors. The geometric models are pre-authored with detailed color, texture and material properties to reproduce desired surface appearance. We use the same paradigm to create a cartoon-like appearance. Although, reproduction of surface reflectance is well studied, simulating apparent motion from static models remains unexplored.

1.1.2 Apparent Motion

The motion analysis by humans is divided into three levels of processing: retinal motion sensing, 2D integration and 3D interpretation. The need to sense retinal motion, and analyze it as quickly as possible, places great demands on the visual system. A great deal of perceptual and physiological research has been conducted to discover the properties of these mechanisms. With several (retinal and other) cues, some potentially conflicting, the visual system attempts to integrate them into a meaningful 'best' solution, which may actually be incorrect [Mather et al 1998]. The illusions that may result in such cases provide us an opportunity.

We are specifically interested in exploiting the perception of motion when there is no corresponding physical displacement in space. This perception of motion is known as *apparent motion* [Smith and Snowden 1994]. There are four well-known types of apparent motion: motion after effect (MAE) (e.g. upward motion of objects near a waterfall), phi phenomenon (e.g. filled in motion in a movie of flickering animation stills) , induced motion (e.g. perceived movement due to a slowly moving train on the next track) and autokinetic movement (e.g. perturbation seen at small light in a dark room). In addition, one type of physical motion may result in some other types of apparent motion due to either 2D integration or 3D interpretation error. A good example is moving shadows which act as a strong cue to depth change.

Using 3D computer graphics, in this paper, we mainly exploit the induced motion effect and the 3D interpretation error for some very constrained cases. To induce motion, we segment a continuous static physical scene into sub-parts. We then illuminate each sub-part with rendered animations so that the multiple movements appear compatible. We also force temporary and invalid 3D interpretation using shadows, lighting and texture movements.

Motion perception is, however, a complex topic. New models in understanding apparent motion are still evolving. There may be many other ways of using the errors in human motion analysis.

1.2 Previous Work

In cartoon and in live action films, characters are made to appear in motion relative to the background by moving the background in the opposite direction. However, there are surprisingly few efforts at generating apparent and sustained motion in the real world. A well-known example is the singing heads at Disney's "Haunted Mansion" [Liljegren 1990]. Pre-recorded video, synchronized with sound, is projected on semi-transparent busts of men to create a compelling effect that the busts are changing shapes corresponding to facial expressions.

Based on the underlying complexity, the previous work in using projectors for augmented reality can be divided into two main groups (i) projection of useful information on a planar surface, and (ii) more complex insertion of 3D shapes and attributes into the real world. There are several examples of 2D augmentation - interactive 2D images projected on flat surfaces in a room to enhance the user's environment, Luminous room [UnderKoffler et al 1997], digital desk [Wellner 1993], and smart whiteboards. The 3D augmentation examples are fewer, Shader Lamps [Raskar et al. 2001][Bandyopadhyay et al. 2001] and the related project Being There [Low et al. 2001].

1.3 Contribution

Our main contribution is a general set of techniques to allow a limited combination of non-photorealistic images with physical models to generate non-realistic appearance and movement. We demonstrate the ideas in the context of a system, discuss methods to make them practical and describe user experiences. We hope to contribute to the body of research in immersive non-realistic animation and spatially augmented reality.

2 Effects

In this paper, our demonstration system includes augmentation of a diorama made up of a toy-car model and its surroundings. In that context, we discuss and describe new rendering techniques that take into account the relationships between the user position, real objects, virtual objects, light projectors and the surface attributes of the virtual objects. The techniques can also be applied to other types of dioramas.

There are effectively two steps: augmenting objects so that they have cartoon appearance and simulating apparent motion.

2.1 Spatial Augmentation

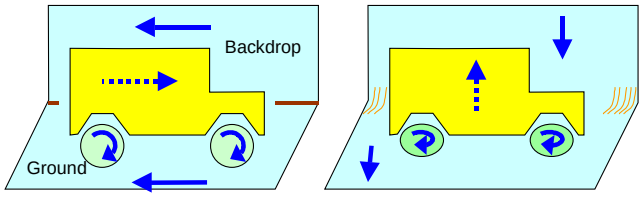
Many physical attributes can be effectively incorporated into the light source to achieve a perceptually equivalent effect on a neutral object. While the results are theoretically equivalent for only a limited class of surfaces and attributes, our experience is that they are quite convincing and compelling for a broad range of applications. The task is easier when a non-realistic appearance is desired. This subsection describes a technique very similar to the one used in [Raskar et al. 01]. The approach for a general projector-based augmentation system is summarized below.

During pre-processing
- Create 3D graphics model, *G*, of physical object
- Create 3D graphics model, *B*, of background
- Approximately position the projector
- Find perspective pose, *P*, of the projector wrt the physical object

During run-time
- Get user location, *U*
- Get animation transformation, *T*
- Modify *G*'s surface attributes
- Render *G* using the pose *P*, and user location *U*
- Transform *B using* T^{-1}, *B'*
- Render *B'* using the pose *P*, and user location *U*
- Modify image intensity to compensate for surface orientation

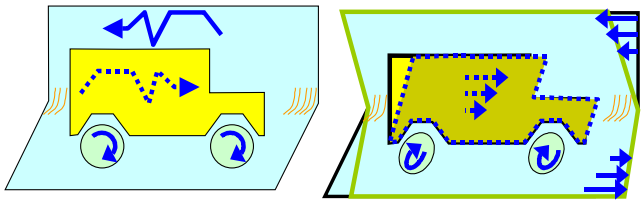
To reproduce purely view-independent surface appearance (diffuse reflectance), user location is not required. For view-dependent effects such as specular highlights, approximate user location is necessary. The user's location, *U*, can be tracked using magnetic or optical tracking technology. In our case, we assume the user is at a sweet-spot standing in front of the car and do not track the user.

The projector projection matrix, *P*, is obtained using an off-line calibration process similar to the technique used for finding internal and external parameters of a camera [Faugeras 1993]. We take a set of fiducials with known 3D locations on the physical object and find the corresponding projector pixels that illuminate them. This allows us to compute a 3x4 perspective projection matrix, up to scale, which is decomposed to find the internal and the external parameters of the projector. The internal parameters represent the focal length and principal point, while the external parameters represent the rigid transformation (translation and rotation) with respect to the coordinate system of the physical object. The rendering process uses the same internal and external parameters to render the pre-authored geometric model, *G*, so that the projected images are registered with the physical objects. During run-time, instead of the object, *G*, the background, *B*, is transformed to create the apparent motion. At each frame, an intensity correction stage pre-multiplies the projected image with intensity weights to compensate for the local surface orientation. Otherwise, surfaces normal to the incident light will appear



(i) Perceived horizontal forward motion, background texture moves backward, wheels rotate clockwise

(ii) Wavy motion due to wheels which are non-circular but attached to the car (up-down motion)



(iii) Bumpy movement on a rough surface

(iv) Shear during acceleration

Figure 2: Types of apparent motion for a car. Rendered motion is shown with blue arrow. The resultant apparent motion is shown with dashed blue arrow.

brighter than surfaces illuminated obliquely due to the cosine fall-off.

2.2 Cartoon Shading

Our focus is mainly creating cartoon-like (apparent) movements. To enhance the cartoon-feel, however, we implemented well-known non-photorealistic shading techniques with some modifications. The scene is made up of a toy-car model, with a simple background of a horizontal and a vertical surface. The horizontal surface coarsely represents the ground surface, either a road or rough terrain. The vertical surface is the backdrop which represents everything else including bushes, trees and sky. To improve the visual fidelity, the background can be made arbitrarily complex as long it has a constant profile along any plane perpendicular to the direction of apparent motion..

We use the technique proposed by [Lake et al. 2000] to render a flat-shaded look. As described below, we modify the color quantization procedure to take into account the cut-off angle of the spot-beams for street lights. For static scenes, we also highlight sharp edges as seen in the traditional 2D cartoon animation stills [Raskar 2001]. However, we realized that for objects with apparent motion, the sharp features are distracting because they work against the notion of motion blur. For dynamic three dimensional scenes, human visual system segments scenes based on grouping of objects in motion [Mather 1998]. Hence, we speculate that, in the retinal image, the boundaries between regions with different optical flow are more important than edges corresponding to discontinuity in surface normal (i.e. sharp features). However, this aspect needs more investigation. Edges corresponding to discontinuity in depth (i.e. silhouettes) are generally a subset of the boundaries between regions with different directions or rates of optical flow.

Figure 3: Vertical displacement by shifting the background and shadows. These two pictures are taken from the same camera position, so the car in both images is at the same location. Note (i) the change in position of the virtual shadows and (ii) the parallax for the edge between ground and backdrop with respect to the right headlight.

2.3 Motion Effects

The car is simulated as driving along a road, on a rough surface, or in various other environments. To create apparent motion, we illuminate wheels with images of rotating wheels. The images of the background (made up of the backdrop and ground) move in a direction opposite to the intended car movement. In the simplest case, as seen in the video, the car appears to move forward i.e. left to right. To create this effect, the wheels rotate clockwise and the background moves right to left (Figure 2(i)). The crucial task for a believable movement is maintaining consistency between the angular wheel movement and corresponding translation of the ground (and the backdrop). For any duration,

$$\int (\text{wheel perimeter arc length}) = \left| \text{displacement of the bkgrnd} \right|$$

This ensures that the wheels are not sliding while the car is in motion. A small amount of motion blur is added to the wheels. Along the direction of the motion, the background geometry with the associated texture maps is infinitely long, and is implemented using a simple sink and a source.

We experimented with many types of cartoon or non-realistic motions. Two additional factors that add to the effect are sound and removal of (physical) sharp edges in the background.

2.3.1 Wobbly Motion

Slow moving cartoon cars usually have a wavy or bumpy movement resulting in a small periodic or random displacement along the vertical direction. This sometimes is emphasized by non-circular wheels. In our case, the car is static. Hence, to create the desired apparent vertical shift while the car is in motion, we instead translate the background in the opposite (vertical) direction (Figure 3). During rendering, the 3D geometric model of the backdrop as well as the ground is translated vertically. The amount of translation, in the case of non-circular wheels, is determined by distance between the point of contact of the wheels from wheel axis (Figure 2(ii)). The distance traveled is again consistent with integration of the arc length of wheel perimeter. We encourage the reader to watch the accompanying video to view both types of motions. (The video gives a good idea of the effect of the transitions, but it is stronger when experienced in person with all the three dimensional cues. Please see the subsection on user reactions.)

It is important to note that to create apparent motion, sometimes, the underlying physical geometry does not match up with the virtual model. For example, with vertical displacement, the boundary between vertical and horizontal surface (shown as

Figure 4: Shading due to street lights as spot lights.

brown sharp edge in Figure 2(i)) does not correspond to the projection of the boundary between the backdrop and ground. Hence, first we need to eliminate any sharp edges that the user may be able to use as a frame of reference. A conflicting cue affects the perception of vertical component of the motion. Second, we need to represent the surface with the approximate average of corresponding variation in the geometry. The solution is to the smooth the edge using a paper with small curvature (see brown lines in Figure 2(ii), (iii)).

2.3.2 Lights and Shadows

Shadows provide a very important cue in apparent motion [Kersten et al. 1997]. For example, in a two-dimensional image sequence (i.e. without any other depth cues), a ball moving diagonally upwards across the screen can be made to appear as if it is gradually moving away from the viewer by rendering a shadow that appears to stay with and under the ball. The cue overrides contradictory information regarding the unchanging size of the ball. The strength of the effect does depend on the properties of the shadow region. For example, shadows with a penumbra, and which fall below the object, work best.

According to [Kersten et al. 1997] the existence of the illusion shows that our perception of the spatial layout of scenes relies on the assumption that the light source responsible for the shadow is stationary, so that any motion of the shadow is attributed to the object casting it.

We enhance the wobbly motion effect by rendering virtual shadows from directional light source (sun) or local lights (street lights). The change in shadow position creates the illusion that the change is a result of changing vertical distance between the car and the ground. We noticed that, it is not necessary to use the same light position to calculate shading and shadows! Further, perceptually, the movement of virtual shadows (Figure 3) is not affected by the fixed real shadows of the physical car on the background.

For night time simulation, the only cues are headlight beams and shading due to the street lights. The spread of the parabolic projection of the headlights and change in overlap between the two beams indicates the vertical shift. The color of the two beams is intentionally chosen slightly different, so that the lateral shift as the car moves up and down is clearly seen (Figure 1 and 7). We also exploit spot lights from the street lights. The basic cartoon shading equation is modified to take into consideration the spot-cut off angle (Figure 4).

2.3.3 Acceleration dependent modifications

Cartoon animators emphasize acceleration, such as a sudden start or a screeching halt, by geometric deformation of the object. For example, a car appears to 'work hard' to move forward while starting when the top of the car is sheared in the direction of the acceleration. Similarly a hard brake and stop is indicated by shearing the top backwards. Since we cannot shear the physical model, we enhance this effect using two tricks.

First, we implement a shear in the background that is opposite of the shear expected in the car (Figure 2(iv)). The shear is along one dimension, along the vertical axis. Hence, for example, during a sudden start, the background shears backward and the shear at a point is proportional to the vertical distance from the center of the car. The points above the vertical center of the car translate backwards while points below translate forward (Figure 5). Without a loss of generality, lets assume that the car center is at the origin, the vertical direction is parallel to the z-axis, and the forward direction is parallel to the x-axis, Then the shear at a given frame is achieved using a simple transformation matrix [1, 0, -*a*, 0; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1]. Where *a* is proportional to the acceleration at that frame.

Since the acceleration is positive during starting, negative during braking, and zero during constant speed and velocity, the same shear equation can be used throughout the animation.

For the second trick, we observe that rapid acceleration also means relative slide between the ground and the wheels. Hence, for example, a sudden brake results in halt in rotation of the wheels, but the background continues to move (Please see the video).

2.3.4 Tracked Illumination

We synchronize and update the rendering of the objects so that the surface textures appear glued to the objects even as the objects move. In this case, as seen in the video, we rotate the car (along with the background) on a turntable. Keeping the virtual light sources fixed, we see corresponding shading changes.

2.3.5 Issues

The type of effects possible are obviously limited. For example, we played with motion blur and motion lines typically seen in cartoon animations, but without success. A simple problem is lack of corresponding physical surface to show motion lines behind the car. We also could not bring in effects such as fog, smoke or fumes. Displaying fog simply washes out the image on the model. During the night time simulation, the backdrop, which is supposed to be at a large depth and hence dark, gets illuminated by the secondary scattering of headlight beams.

3 Other Applications

A setup to introduce non-realistic appearance and motion can be used for many other applications. Some examples are educational

Figure 5: Shearing of the background during acceleration.

interactive setups (e.g. physics experiments), science and art museums, stage drama shows, showroom displays, virtual sets and physical mock-ups for special effects. We assumed the user to be at a sweet-spot, but with head-tracking many view-dependent effects can be incorporated. They include simulating complete (possibly non-realistic) reflectance properties (BRDF), inserting animated floating 3D objects close to the physical surface (e.g. crawling ants or flowing water) or advanced real-time non-photorealistic shading effects [Praun et al. 2001].

We used a simple turn-table to demonstrate illumination of a moving physical object. This can be extended further in many ways depending on the degrees of freedom of the movement. For example, current car motion is mostly uniform. By adding small and sudden turntable rotations in and out of the plane, the car can be made to appear to be taking turns around a corner or avoiding a obstacle in the middle of the road.

While there is a growing body of physiological research and studies of apparent motion, the work so far is limited to two-dimensional experiments and in many cases limited to computer screen simulations. It will be interesting to use the proposed setup to study other type of stimulation for the human visual system and better understand the model of motion analysis.

4 Benefits and limitations

A key benefit of this type of spatial augmentation is that a visually rich, large field-of-view diorama can be generated with greater amount of integration of virtual (surface and motion) attributes with the real world. Multiple people can enjoy the enhanced diorama just as in the traditional dioramas.

A crucial problem is the dependence on display surface properties. A light colored diffuse object with smooth geometry is ideal. It is practically impossible to render vivid images on highly specular, low reflectance or dark surfaces. The ambient lighting as well as secondary scattering from diffuse surfaces can also affect the contrast in the images. This limits the applications to controlled lighting environments with restrictions on the type of objects with which virtual objects will be registered. Shadows can also be a problem if close human interaction is desired.

5 Implementation and lessons

Currently we use a Mitsubishi X80U LCD projector, 1000 lumen, 1024x768 resolution. The toy-car (Barbie Beetle, bought at a toy-store) is 50cm x 25cm x 20cm (Figure 6). The 3D model is generated from multiple photographs using PhotoModelerPro and then authored in 3DSMax. The model is very coarse, 602 triangles. We struggled with creating smoothly varying normals across the car surface, resulting in shading artifacts. A better 3D scanning method should overcome those problems. The animations are rendered at approx. 15-20 fps. To determine projector pose during calibration, we used 15 pairs of pre-determined 3D points on the car surface and the corresponding projector pixels that illuminate them. The projectors pixels are interactively found by projecting a cross-hair and moving the mouse till the cross-hair lines up with one of the 3D feature point. This process takes only two minutes. The resulting re-projection error is less than one pixel allowing near-perfect static registration. To find the axis of rotation of the turn table, a second set of projector pixels for the same 3D points, after angular displacement, is used.

Figure 6: Setup with a projector and simple diorama.

5.1 Use Reactions

We do not have systematic user experience data but the system has been seen by over two thousand people. The working system has been demonstrated to visitors of the lab, at an art show, during UIST'01 conference, and to dozens of Disney researchers in Orlando, Florida. Most users get excited at the following stages: (i) when the car *starts*, (ii) when the car motion *switches* to bumpy motion, (iii) when the car is rotated on the turn table while maintaining its augmented surface appearance. As mentioned earlier, the effect is most noticeable during any changes in apparent motion and it lasts for a short time. The human visual motion detection system quickly becomes balanced again. Hence, it is important to constantly change the motion parameters. The synchronized car engine (or brake) sound seems to have a huge impact, it generates anticipation before the car starts and reinforces the notion of a car movement (as opposed to movement of the background). We implemented a photo-realistic version as well as the cartoon-like version (Figure 7). The cartoon-like version appeals more when the car is in motion. Viewers, not familiar with the technique, are unimpressed to see the static cartoon version. This is probably because we are all used to looking at much higher quality cartoon-like setups. But, the

Figure 7: Photo-realistic diorama (left column) vs. cartoon diorama (right column). Note the headlight beams, specular appearance and textures.

interest increased when we interactively changed the surface appearance of the car or the background. Many viewers, even after explaining that we are projecting on white surfaces, are very surprised when the projector light is blocked to show the underlying dull setup of simple static models. Comparing figure 6 and figure 7, it is surprising to see the amount of fine detail possible with purely projected images.

So far, two viewers (both women in their 30s) have complained of feeling dizzy (a well known side effect of VR/VE worlds). Children show a tendency to run and catch the moving car. Advanced technical viewers after careful observation sometimes notice the lack of complete illumination (because we are using only one projector and, for example, shadow of the car on the backdrop is not filled in by multiple projectors). Please view the video and more images at http://www.shaderlamps.com.

6 Conclusion and Future work

We have presented a system and associated techniques to show apparent cartoon-like motion on a simple diorama. The techniques allow only a limited combination of rendered animation and static neutral colored physical objects. But, the proposed methods are sufficiently general and can be used in a variety of applications.

Similar to *non-realistic* rendering, there is an interesting unexplored area of *non-realistic* motion simulation. We have presented some preliminary ideas and a prototype system. There are many possible directions for combining apparent motion effects with real time 3D rendering in projector-based augmentation. For example, despite the visually compelling results, the visual adaptation effect lasts for a short time. How can we make it last longer ? For most effects, the motion detection system quickly becomes balanced and the apparent movement becomes weaker or stops. However, the four-stroke apparent motion [Mather et al. 1998] is a wonderful illusion where the motion appears to continue forever in the same direction. The animation involves a repeating cycle of four different frames. e.g. a car moves forward and then backward, but the backward step is accompanied by a reversal in intensities. The effect is to reverse the direction of perceived motion, resulting in forward and then again forward motion. Can we achieve similar effects on three dimensional objects ?

We currently limit effects to static models. However, one can imagine using phi phenomenon to create apparently morphing shapes out of two or more static models. Shape change over time presents the visual system with a correspondence (matching) problem. The 'best' solution is interpreted based on a few geometric properties. Thus, successively illuminating different shapes creates the apparent and smooth interpolating morph. This has been explored in 2D computer simulations and may be extended to 3D objects by changing focus of attention from one static shape to the next with projected light.

Lastly, our use of synchronized realistic sound was limited to enhancing the visual effects. But, as noted recently by [Shams et al. 2000] sound can create virtual motion perception. It should be worth exploring the use of realistic or even non-realistic spatialized sound to further intensify the effects.

Acknowledgements

We would like to thank members of the ShaderLamps project, Greg Welch, Kok-lim Low and Deepak Bandyopadhyay for useful discussions.

References

BANDYOPADHYAY D., RASKAR R., FUCHS H. 2001. Dynamic Shader Lamps: Painting on Movable Objects. In *Proceedings of Int. Symp. On Augmented Reality*.

FAUGERAS, O. 1993 *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, Cambridge, Massachusetts.

KERSTEN, D., MAMASSIAN P., AND KNILL D. 1997. Moving cast shadows induce apparent motion in depth. *Perception*, 26, 171-192.

LAKE, A., MARSHALL, C., HARRIS, M. AND BLACKSTEIN, M. 2000. Stylized Rendering Techniques for Scalable Real-Time 3D Animation, In *Proceedings of Non-Photorealistic Animation and Rendering*, Annecy, France, June 5-7.

LILJEGREN, G. E. AND FOSTER, E. L. 1990. Figure with Back Projected Image Using Fiber Optics. US Patent # 4,978.216, Walt Disney Company, USA.

LOW, K., WELCH, G., LASTRA, A., FUCHS H. 2001. Life-Sized Projector-Based Dioramas. *Symposium on Virtual Reality Software and Technology*.

MATHER, G., VERSTRATEN, F., AND ANSTIS, S. 1998. *The Motion Aftereffect: a Modern Perspective.* MIT Press, Cambridge, Massachusetts.
(http://www.biols.susx.ac.uk/home/George_Mather/Motion/)

MILGRAM, P., TAKEMURA, H., UTSUMI, A., AND KISHINO, F. 1994. Augmented Reality: A class of displays on the reality-virtuality continuum. SPIE Vol. 2351-34, Telemanipulator and Telepresence Technologies.

PRAUN, E., HOPPE, H., WEBB, M., FINKELSTEIN, A. 2001 Real-Time Hatching. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM.

RASKAR, R., WELCH, G., LOW, K., BANDYOPADHYAY D. 2001. Shader Lamps, Animating Real Objects with Image Based Illumination. In *Proceedings of the 12th Eurographics Workshop on Rendering*.

RASKAR, R. 2001. Hardware Support for Non-photorealistic Rendering. In *Proceedings of the ACM/Eurographics Workshop on Graphics Hardware*.

SHAMS, L., KAMITANI Y. AND SHIMOJO, S. 2000. What you see is what you hear. *Nature*, pp 788. Dec 14, 2000. (http://neuro.caltech.edu/publications/shams.shtml)

SMITH A, AND SNOWDEN R 1994. *Visual Detection of Motion*. London: Academic Press.

UNDERKOFFLER, J. 97. A View From the Luminous Room. Springer-Verlag London Ltd., *Personal Technologies* (1997) 1:49–59.

WELLNER, P. 1993. Interacting with paper on the DigitalDesk. *Communications of the ACM*, 36(7):87–96. July

http://www.shaderlamps.com

iLamps: Geometrically Aware and Self-Configuring Projectors

Ramesh Raskar Jeroen van Baar Paul Beardsley Thomas Willwacher Srinivas Rao Clifton Forlines

Mitsubishi Electric Research Labs (MERL), Cambridge MA, USA *

Abstract

Projectors are currently undergoing a transformation as they evolve from static output devices to portable, environment-aware, communicating systems. An enhanced projector can determine and respond to the geometry of the display surface, and can be used in an ad-hoc cluster to create a self-configuring display. Information display is such a prevailing part of everyday life that new and more flexible ways to present data are likely to have significant impact. This paper examines geometrical issues for enhanced projectors, relating to customized projection for different shapes of display surface, object augmentation, and co-operation between multiple units.

We introduce a new technique for adaptive projection on non-planar surfaces using conformal texture mapping. We describe object augmentation with a hand-held projector, including interaction techniques. We describe the concept of a display created by an ad-hoc cluster of heterogeneous enhanced projectors, with a new global alignment scheme, and new parametric image transfer methods for quadric surfaces, to make a seamless projection. The work is illustrated by several prototypes and applications.

CR Categories: B.4.2 [Input/output and Data Communications]: Input/Output Devices—Image display ; H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities I.4.1 [Image Processing and Computer Vision]: Digitization and Image Capture—Imaging geometry

Keywords: projector, calibration, seamless display, augmented reality, ad-hoc clusters, quadric transfer.

1 Introduction

Traditional projectors have been static devices, and typical use has been presentation of content to a passive audience. But ideas have developed significantly over the past decade, and projectors are now being used as part of systems which sense the environment. The capabilities of these systems range from simple keystone correction to augmentation overlay on recognized objects, including various types of user interaction.

Most such systems have continued to use static projectors in a semi-permanent setup, one in which there may be a significant calibration process prior to using the system. Often too the systems are specifically designed for a particular configuration. But the increasing compactness and cheapness of projectors is enabling much more flexibility in their use than is found currently. For example, portability and cheapness open the way for clusters of projectors which are put into different environments for temporary deployment, rather than a more permanent setup. As for hand-held use, projectors look like a natural fit with cellphones and PDAs. Cellphones provide access to the large amounts of wireless data which surround us, but their size dictates a small display area. An attached projector can maintain compactness while still providing a reasonably-sized display. A hand-held cellphone-projector becomes a portable and easily-deployed information portal.

These new uses will be characterized by opportunistic use of portable projectors in arbitrary environments. The research challenge is how to create *Plug-and-disPlay* projectors which work flexibly in a variety of situations. This requires generic application-independent components, in place of monolithic and specific solutions.

This paper addresses some of these new problems. Our basic unit is a projector with attached camera and tilt-sensor. Single units can recover 3D information about the surrounding environment, including the world vertical, allowing projection appropriate to the display surface. Multiple, possibly heterogeneous, units are deployed in clusters, in which case the systems not only sense their external environment but also the cluster configuration, allowing self-configuring seamless large-area displays without the need for additional sensors in the environment. We use the term *iLamps* to indicate intelligent, locale-aware, mobile projectors.

1.1 Overview

The focus of this paper is geometry. Successive sections address issues about the geometry of display surfaces, 3D motion of a hand-held projector, and geometry of a projector cluster. Specifically, we make the following contributions –

Shape-adaptive display: We present a new display method in which images projected on a planar or non-planar surface appear with minimum local deformation by utilization of conformal projection. We present variations to handle horizontal and vertical constraints on the projected content.

Object-adaptive display: We demonstrate augmentation of objects using a hand-held projector, including interaction techniques.

Planar display using a cluster of projectors: We present algorithms to create a self-configuring ad-hoc display network, able to create a seamless display using self-contained projector units and without environmental sensors. We present a modified global alignment scheme, replacing existing techniques that require the notion of a master camera and Euclidean information for the scene.

Curved display using a cluster of projectors: We extend planar surface algorithms to handle a subset of curved surfaces, specifically quadric surfaces. We introduce a simplified parameterized transfer equation. While several approaches have been proposed for seamless multi-projector planar displays, as far as we know, literature on seamless displays is lacking in techniques for parameterized warping and registration for curved screens.

We omit discussion of photometric issues, such as the interaction between color characteristics of the projected light [Majumder et al.

*email:[raskar, jeroen, pab, willwach, raos, forlines]@merl.com
http://www.merl.com/projects/Projector/

2000] and environmental characteristics like surface reflectance, orientation, and ambient lighting. We also omit discussion about non-centralized cluster-based systems and issues such as communication, resource management and security [Humphreys et al. 2001; Samanta et al. 1999]. Finally a full discussion of applications is outside the scope of the paper, though we believe the ideas here will be useful in traditional as well as new types of projection systems.

1.2 Evolution of Projectors

Projectors are getting smaller, brighter, and cheaper. The evolution of computers is suggestive of the ways in which projectors might evolve. As computers evolved from mainframes to PCs to hand-held PDAs, the application domain went from large scientific and business computations to small personal efficiency applications. Computing has also seen an evolution from well-organized configurations of mainframes to clusters of heterogeneous, self-sufficient computing units. In the projector world, we may see similar developments – towards portable devices for personal use; and a move from large monolithic systems towards ad-hoc, self-configuring displays made up of heterogeneous, self-sufficient projector units.

The most exploited characteristic of projectors has been their ability to generate images that are larger in size than their CRT and LCD counterparts. But the potential of other characteristics unique to projector-based displays is less well investigated. Because the projector is decoupled from the display (i) the size of the projector can be much smaller than the size of the image it produces, (ii) overlapping images from multiple projectors can be effectively superimposed on the display surface, (iii) images from projectors with quite different specifications and form factors can be easily blended together, and (iv) the display surface does not need to be planar or rigid, allowing us to augment many types of surfaces and merge projected images with the real world.

1.3 Relevant Work

The projector's traditional roles have been in the movie, flight simulator, and presentation markets, but it is now breaking into many new areas.

Projector-based environments There are many devices that provide displays in the environment, and they are becoming more common. Some examples are large monitors, projected screens, and LCD or plasma screens for fixed installations, and hand-held PDAs for mobile applications. Immersion is not a necessary goal of most of these displays. Due to shrinking size and cost, projectors are increasingly replacing traditional display mediums. We are inspired by projector-based display systems that go beyond the traditional presentation or multi-projector tiled displays: Office of the Future [Raskar et al. 1998], Emancipated Pixels [Underkoffler et al. 1999], Everywhere Display [Pinhanez 2001] and Smart Presentations [Sukthankar et al. 2001]. Many new types of projector-based augmented reality have also been proposed [Raskar et al. 2001; Bimber et al. 2002]. From a geometric point of view, these systems are based on the notion of one or more environmental sensors assisting a central intelligent device. This central hub computes the Euclidean or affine relationships between projector(s) and displays. In contrast, our system is based on autonomous units, similar to self-contained computing units in cluster computing (or ubiquitous computing). In the last four years, many authors have proposed automatic registration for seamless displays using a cluster of projectors [Yang et al. 2001; Raskar et al. 2002; Chen et al. 2002; Brown and Seales 2002]. We improve on these techniques to allow the operation without environmental sensors and beyond the range of any one sensor. We also extend the cluster based approach to second-order display surfaces.

Enhanced projectors Another related area of research is the enhancement of projectors using sensors and computation. Underkoffler et al. [1999] described an I/O bulb (co-located projector and camera). Hereld et al. [2000] presented a smart projector with an attached camera. Raskar and Beardsley [2001] described a geometrically calibrated device with a rigid camera and a tilt sensor to allow automatic keystone correction. Many have demonstrated user interactions at whiteboards and tabletop surfaces including the use of gestural input and recognition of labeled objects [Rekimoto 1999; Rekimoto and Saitoh 1999; Crowley et al. 2000; Kjeldsen et al. 2002]. We go further, by adding network capability, and by making the units self-configuring and geometrically-aware. This allows greater portability and we investigate techniques which anticipate the arrival of hand-held projectors.

Grids Our approach to ad-hoc projector clusters is inspired by work on grids such as ad-hoc sensor networks, traditional dynamic network grids, and ad-hoc computing grids or network of workstations (NoW, an emerging technology to join computers into a single vast pool of processing power and storage capacity). Research on such ad-hoc networks for communication, computing and data-sharing has generated many techniques which could also be used for context-aware 'display grids'.

2 Geometrically Aware Projector

What components will make future projectors more intelligent? We consider the following elements essential for geometric awareness – sensors such as camera and tilt-sensor, computing, storage, wireless communication and interface. Note that the projector and these components can be combined in a single self-contained unit with just a single cable for power, or no cable at all with efficient batteries.

Figure 1 illustrates a basic unit. This unit could be in a mobile form factor or could be a fixed projector. Because we do not wish to rely on any Euclidean information external to the device (e.g., markers in the room, boundaries on screens, or human aid), we use a completely calibrated projector-camera system.

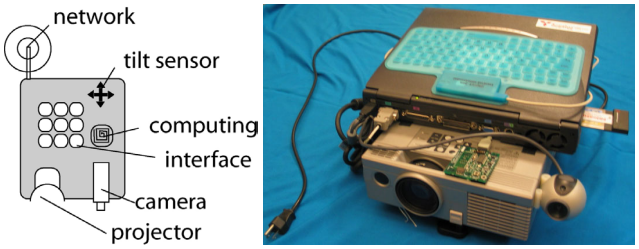


Figure 1: *Our approach is based on self-contained iLamps. Left: components of enhanced projector; Right: our prototype, with a single power cable.*

In isolation, the unit can be used for several applications including (i) smart keystone correction (ii) orientation compensated intensities (iii) auto brightness, zoom, focus (iv) 3D scanner for geometry and texture capture (with auto zippering of piecewise reconstructions by exploiting the camera with accelerometer) (v) smart flash for cameras, with the projector playing a secondary role to provide intelligent patterns or region specific lighting.

The unit can communicate with other devices and objects to learn geometric relationships as required. The ability to learn these relationships on the fly is a major departure from most existing projector-based systems that involve a preconfigured geometric setup or, when used in flexible environments, involve detailed calibration, communication and human aid. Even existing systems that

use a simple planar homography and avoid complete calibration require some Euclidean information on the screen (e.g., screen edges or markers) [Sukthankar et al. 2001] or assume the camera is in the ideal sweet-spot position [Yang et al. 2001; Raskar et al. 2002; Brown and Seales 2002].

3 Shape-adaptive Display

When using projectors casually and portably, an ideal planar display surface is not always available, and one must take advantage of other surfaces such as room corners, columns, or oddly shaped ceilings. The shape-adaptive display in this section emulates existing examples of texture on curved surfaces, such as large advertisements and news tickers on curved displays, and product labels on curved containers. The issue is how to generate images that appear 'correctly' to multiple simultaneous viewers. This is a different problem to pre-warping an input image so that it appears perspectively correct from a *single* sweet-spot location [Raskar et al. 1999]. Human vision interprets surface texture in the context of all three-dimensional cues – when viewing a poster on the wall from one side, or reading the label of a cylindrical object such as a wine bottle, or viewing a mural on a curved building. The goal therefore is to create projected texture which is customized to the shape of the surface, to be consistent with our usual viewing experience.

3.1 Conformal Projection

This section describes how to display an image that has minimum stretch or distortion over the illuminated region. Consider first a planar surface like a movie screen – the solution is to project images as if the audience is viewing the movie in a fronto-parallel fashion, and this is achieved by keystone correction when the projector is skewed. Now consider a curved surface or any non-planar surface in general. Intuitively, we wish to 'wallpaper' the image onto the display surface, so that locally each point on the display surface is undistorted when viewed along the surface normal.

Since the normal may vary, we need to compute a map that minimizes distortion in some sense. We chose to use conformality as a measure of distortion. A conformal map between the input image and the corresponding areas on the display surface is angle preserving. A scroll of the input image will then appear as a smooth scroll on the illuminated surface, with translation of the texture but no change in size or shape.

A zero-stretch solution is possible only if the surface is developable. Example developable surfaces are two planar walls meeting at a corner, or a segment of a right cylinder (a planar curve extruded perpendicular to the plane). In other cases, such as three planar walls meeting in a corner, or a partial sphere, we solve the minimum stretch problem in the least squares sense. We compute the desired map between the input image and the 3D display surface using the least squares conformal map (LSCM) proposed in Levy et al. [2002]. An example of texture projection using this approach is shown in Figure 2.

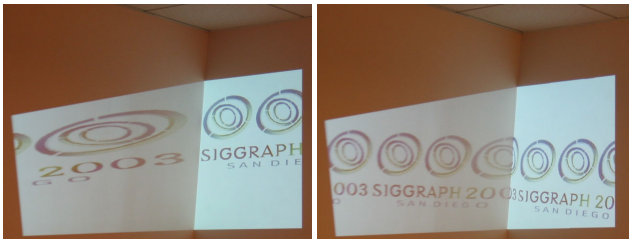


Figure 2: *Shape-adaptive projection. Left: the projector is skew relative to the left wall so direct, uncorrected projection of texture gives a distorted result; Right: the projector still in the same position but use of LSCM removes the distortion in the projected image.*

LSCM minimizes angle deformation and non-uniform scaling between corresponding regions on a 3D surface and its 2D parameterization space, the solution being fully conformal for a developable surface. For a given point X on the 3D mesh, if the 2D texture coordinates (u,v) are represented by a complex number $(u+iv)$ and the display surface uses coordinates in a local orthonormal basis $(x+iy)$, then the goal of conformal mapping is to ensure that tangent vectors to the iso-u and iso-v curves passing through X are orthogonal and have the same norm, i.e.,

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \qquad\qquad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$$

In Levy et al. [2002], the authors solve this problem on a per triangle basis and minimize the distortion by mapping any surface homeomorphic to a disk to a (u,v) parameterization. The steps of our algorithm are as follows.

1. Project structured light from the projector, capture images with a rigidly attached calibrated camera, and create a 3D mesh D of the surface.
2. Use LSCM to compute texture coordinates U of D, thereby finding a mapping D^{Π} of D in the (u,v) plane.
3. Find the displayable region in D^{Π} that (a) is as large as possible and (b) has the vertical axis of the input image aligned with the world vertical. The method for determining the rotation between the input image and the world vertical is described below.
4. Update U into U' to correspond to the displayable region.
5. Texture-map the input image onto the original mesh D, using U' as texture coordinates, and render D from the viewpoint of the projector.

3.2 Vertical Alignment

The goal of vertical alignment is to ensure the projected image has its vertical direction aligned with the world vertical. There are two cases – (a) if the display surface is non-horizontal, the desired texture vertical is given by the intersection of the display surface and the plane defined by the world vertical and the surface normal, (b) if the display surface is horizontal, then the texture vertical is undefined. Regarding condition (b), the texture orientation is undefined for a single horizontal plane, but given any non-horizontal part on the display surface this will serve to define a texture vertical which also applies to horizontal parts of the surface.

The update of U into U' involves a rotation, R, for vertical alignment, in addition to a scale and shift in the (u,v) plane. If the computed 3D mesh were perfect, the computation of R could use a single triangle from the mesh. But the 3D data is subject to error, so we employ all triangles in a least-squares computation. The approach is as follows.

1. For each non-horizontal triangle, t_j, in the 3D mesh D, (i) Compute the desired texture vertical as the 3D vector $p_j = n \times (v \times n)$, where n is the surface normal of the triangle and v is the world vertical (obtained from the tilt-sensor), and $\times$ is the cross-product operator, (ii) Use the computed LSCM to transform p_j into normalized vectors $q_j = (u_j, v_j)$ in the (u,v) plane.
2. Find the rotation which maximizes the alignment of each q_j with direction (0,1) – compute the covariance matrix $M = \sum_j [u_j \quad v_j]^T [0 \quad 1]$, find the singular value decomposition of M as TSV^T, and compute the desired 2D rotation as $R = TV^T$.

3.3 Shape Constraints

It is sometimes desirable to constrain the shape of projected features in one direction at the cost of distortion in other directions. For example, banner text projected on a near-vertical but non-developable surface such as a sphere-segment should appear with all the text characters having the same height, even if there is distortion in the horizontal direction. Additional constraints on the basic four partial derivatives in LSCM are obtained by introducing equations of the form $\lambda_{vert} \cdot (\frac{\partial v}{\partial y} - const) = 0$. Typically, only one such equation will be used. The equation above, for example, keeps stretch along the vertical direction to a minimum, i.e., it penalizes and minimizes the variance in $\partial v/\partial y$ over all triangles. This modification also requires that the local orthonormal x,y-basis on the triangles is chosen appropriately – in this case, the x-axis must point along the horizontal everywhere on the surface. Figure 3 shows an example.

Results Surfaces used for conformal display, shown here and in the accompanying materials, include a two-wall corner, a concertina-shaped display, and (as an example of a non-developable surface) a concave dome.

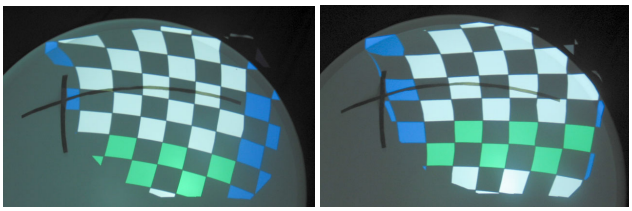


Figure 3: *Left: uncorrected projection from a skew projector; Right: correction of the texture using constrained LSCM. Observe the change in the area at upper-left. Image is world horizontal aligned. Vertical stretch is minimized (at the cost of horizontal distortion) so that horizontal lines in the input texture remain in horizontal planes.*

4 Object-adaptive Display

This section describes object augmentation using a hand-held projector, including a technique for doing mouse-style interaction with the projected data. Common to some previous approaches, we do object recognition by means of fiducials attached to the object of interest. Our fiducials are 'piecodes', colored segmented circles like the ones in Figure 4, which allow thousands of distinct color-codings. As well as providing identity, these fiducials are used to compute camera pose (location and orientation) and hence projector pose since the system is fully calibrated[1]. With projector pose known relative to a known object, content can be overlaid on the object as required.

Advantages of doing object augmentation with a projector rather than by annotated images on a PDA include (a) the physical size of a PDA puts a hard limit on presented information; (b) a PDA does augmentation in the coordinate frame of the camera, not the user's frame, and requires the user to context-switch between the display and physical environment; (c) a PDA must be on the user's person while a projector can be remote; (d) projection allows a shared experience between users. Eye-worn displays are another important augmentation technique but they can cause fatigue, and there are stringent computational requirements because of the tight coupling between user motion and the presented image (e.g., a user head rotation must be matched precisely by a complementary rotation in the displayed image). Projection has its own disadvantages – it is poor on dark or shiny surfaces, and can be adversely affected by ambient light; it does not allow private display. But a key point is that projector-based augmentation naturally presents to the user's own viewpoint, while decoupling the user's coordinate frame from the processing. This helps in ergonomics and is easier computationally.

A hand-held projector can use various aspects of its context when projecting content onto a recognized object. We use proximity to the object to determine level-of-detail for the content. Other examples of context for content control would be gestural motion, history of use in a particular spot, or the presence of other devices for cooperative projection. The main uses of object augmentation are (a) information displays on objects, either passive display, or training applications in which instructions are displayed as part of a sequence (Figure 4(top)); (b) physical indexing in which a user is guided through an environment or storage bins to a requested object (Figure 4(bottom)); (c) indicating electronic data items which have been attached to the environment. Related work includes the Magic Lens [Bier et al. 1993], Digital Desk [Wellner 1993], computer augmented interaction with real-world environments [Rekimoto and Nagao 1995], and Hyper mask [Yotsukura et al. 2002].

Figure 4: *Context-aware displays. Top: augmentation of an identified surface; Bottom: guidance to a user-requested object in storage bins.*

Mouse-Style Interactions with Augmentation Data. The most common use of projector-based augmentation in previous work has been straightforward information display to the user. A hand-held projector has the additional requirement over a more static setup that there is fast computation of projector pose, so that the augmentation can be kept stable in the scene under user motion. But a hand-held projector also provides a means for doing mouse-style interactions – using a moving cursor to interact with

[1] We use four coplanar points in known position in a homography-based computation for the pose of the calibrated camera. The points are obtained from the segments of a single piecode, or from multiple piecodes, or from one piecode plus a rectangular frame. For good results, augmentation should lie within or close to the utilized points.

the projected augmentation, or with the scene itself.

Consider first the normal projected augmentation data – as the projector moves, the content is updated on the projector's image plane, so that the projected content remains stable on the physical object. Now assume we display a cursor at some fixed point on the projector image plane, say at the center pixel. This cursor will move in the physical scene in accordance with the projector motion. By simultaneously projecting the motion-stabilized content and the cursor, we can emulate mouse-style interactions in the scene. For example, we can project a menu to a fixed location on the object, track the cursor to a menu item (by a natural pointing motion with the hand-held projector), and then press a button to select the menu item. Alternatively the cursor can be used to interact with the physical scene itself, for example doing cut-and-paste operations with the projector indicating the outline of the selected area and the camera capturing the image data for that area. In fact all the usual screen-based mouse operations have analogs in the projected domain.

5 Cluster of Projectors

The work so far has been on individual projector units. This section deals with ad-hoc clusters of projector units. Each individual unit senses its geometric context within the cluster. This can be useful in many applications. For example, the geometric context can allow each projector to determine its contribution when creating a large area seamless display. Multiple units can also be used in the shape- and object-adaptive projection systems described above.

This approach to display allows very wide aspect ratios, short throw distance between projectors and the display surfaces and hence higher pixel resolution and brightness, and the ability to use heterogeneous units. An ad-hoc cluster also has the advantages that it (a) operates without a central commanding unit, so individual units can join in and drop out dynamically, (b) does not require environmental sensors, (c) displays images beyond the range of any single unit, and (d) provides a mechanism for bypassing the limits on illumination from a single unit by having multiple overlapping projections.

These concepts are shown working in the context of planar display, and also for higher order surfaces, such as quadric surfaces. For the latter, we present a new image transfer approach. In the work here, each projector unit has access to the same full-size image, of which it displays an appropriate part. If bandwidth were an important constraint, one would want to decompose content and transmit to an individual projector only the pixels which it requires, but that topic is not discussed.

5.1 Planar Display using Ad-Hoc Clusters

This section deals with a cluster projecting on the most common type of display surface, a plane. Existing work on projector clusters doing camera-based registration, such as [Raskar et al. 2002; Brown and Seales 2002], involves projection of patterns or texture onto the display plane, and measurement of homographies induced by the display plane. The homographies are used together with some Euclidean frame of reference to pre-warp images so that they appear geometrically registered and undistorted on the display.

However, creating wide aspect ratios has been a problem. We are able to overcome this problem because a single master camera sensor is not required and we use a new global alignment strategy that relies on pair-wise homographies between a projector of one unit and the camera of the neighboring unit. Figure 5 shows a heterogeneous cluster of five units, displaying seamless images after accurate geometric registration. The pair-wise homographies are used to compute a globally consistent set of homographies by solving a linear system of equations. Figure 6(left) is a close-in view demonstrating the good quality of the resulting registration.

Units can dynamically enter and leave a display cluster, and the alignment operations are performed without requiring significant pre-planning or programming. This is possible because (a) every unit acts independently and performs its own observations and calculations, in a symmetric fashion (b) no Euclidean information needs to be fed to the system (such as corners of the screen or alignment of the master camera), because tilt-sensors and cameras allow each projector to be geometrically aware. In contrast to our approach, systems with centralized operation for multi-projector display quickly become difficult to manage.

The approach is described below in the context of creating a large planar display . A *group* of projectors display a seamless image, but there may be more than one group in the *vicinity*.

Joining a group When a unit, U_k, containing a projector, P_k, and a camera, C_k, wants to join a group, it informs the group in two ways. Over the proximity network (such as wireless Ethernet, RF or infrared) it sends a 'request to join' message with its own unique id, which is received by all the m units, U_i for $i = 1..m$, in the vicinity. This puts the cameras, C_i for $i = 1..m$, of all the units in attention mode and the units respond with 'ready' message to U_k. The second form of communication occurs via light. Unit U_k projects a structured pattern, which may interrupt the display and is observed by all the m cameras embedded in the units. If any one camera from the existing group views the projected pattern, the whole groups moves onto a quick calibration step to include P_k in their display. Otherwise, the group assumes that U_k is in the vicinity but does not overlap with its own extent of the display. Without a loss of generality let us assume that the first n units now form a group.

Pairwise Geometric affine relationship A well-known method to register overlapping projectors is to express the relationship using a homography. The mapping between two arbitrary perspective views of an opaque planar surface in 3D can be expressed using a planar projective transfer, expressed as a 3x3 matrix defined up to a scale. The 8 degrees of freedom can be computed from four or more correspondences in the two views. In addition, due to the linear relationship, homography matrices can be cascaded to propagate the image transfer.

Unit U_k directs, using wireless communication, each projector , P_i for $i = 1..n$, in the group to project a structured pattern (a uniform checkerboard), one at a time. Projection is simultaneously viewed by the camera of each unit in the group. This creates pairwise homographies H_{PiCj} for transferring the image of projector P_i into image in camera C_j.

We calculate pairwise projector homography, H_{PiPj}, indirectly as $H_{PiCi}H_{PjCi}{}^{-1}$. For simplicity, we write H_{PiPj} as H_{ij}. In addition, we store a confidence value, h_{ij}, related to the percentage of overlap in image area and it is used during global alignment later. Since we use a uniform checkerboard pattern, a good approximation for the overlap percentage is the ratio r_{ij} of (the number of features of projector P_j seen by camera C_i) to (the total number of features projected by P_j). We found confidence $h_{ij} = r_{ij}{}^4$ to be a good metric. The value is automatically zero if the cameras i did not see the projector j.

Global Alignment In the absence of environmental sensors, we compute the relative 3D pose between the screen and all the projectors to allow a seamless display. Without a known pose, the computed solution is correct up to a transformation by a homography and will look distorted on screen. Further, if the screens are vertical planes, our approach automatically aligns the projected image with the world horizontal and vertical.

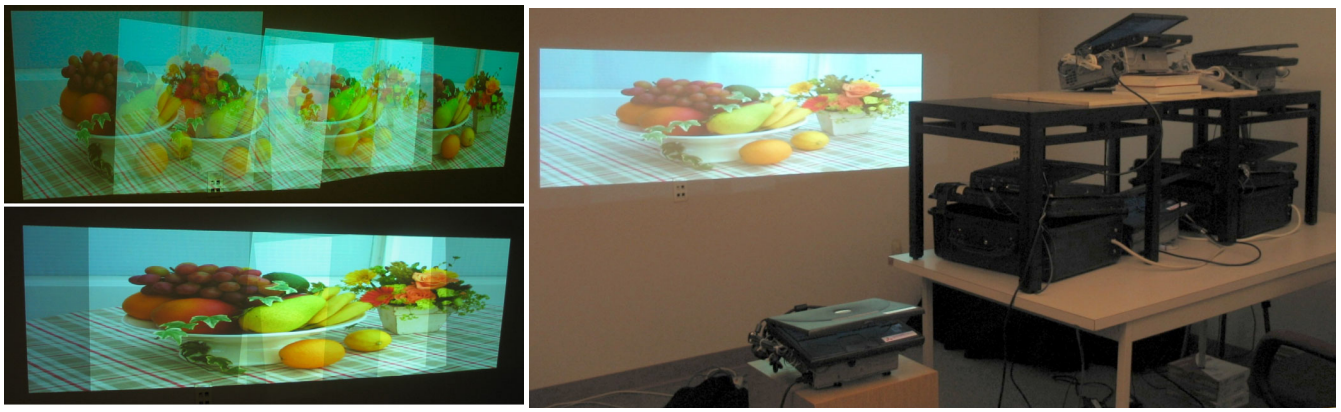


Figure 5: *Application of self-configuring projectors in building wide aspect ratio displays. Left-top: Uncorrected projection from each of the five projectors; Left-bottom: Registered images; Right: Setup of self-contained units and seamless display.*

However, to compute the 3D pose for global alignment, we avoid low confidence information such as stereo calibration and tilt-sensor parameters of units, and exploit relatively robust image space calculations such as homographies. Thus, we first compute globally consistent homographies. A set of homography transforms cascaded along a cycle $\prod_{i=0}^{k-1} H_{(i+1\%k)(i)}$ should equal identity. This is seldom the case due to feature location uncertainty. Thus the goal is to create a meaningful mosaic in the presence of minor errors. A simple solution is to use a single camera viewing the display region [Yang et al. 2001; Chen et al. 2000] and perform computation in that space. When a single camera cannot view the whole display, Chen et al. [2002] use a minimum spanning tree-based approach. The graph nodes are projectors which are connected to overlapping projectors by edges. The solution keeps only a minimal set of homographies. By using a projector near the center of the display as the root node of the tree, this approach reduces the number of cascaded homographies and thereby lowers the cumulative error.

We instead present a simple and efficient method for finding a globally consistent registration by using information from all the available homographies *simultaneously*. The basic approach is based on a scheme proposed by [Davis 1998]. Cumulative errors due to homography cascading are concurrently reduced with respect to a chosen reference frame (or plane) by solving a linear system of equations. The linear equations are derived from independently computed pairwise homography matrices. We modify the method to consider the confidence h_{ij} in measured pairwise projector homography H_{ij}.

A global alignment homography G_i transfers image I_i in P_i into the global reference frame. But it can also be written as a transfer of I_i to I_j with H_{ij} followed by transfer to global frame with G_j.

$$G_j H_{ij} \cong G_i$$

Thus, we can build a sparse linear system of equations, where each H_{ij} is known, to compute the unknown global transfers. Since, each homography is defined up to a scale factor, it is important that we normalize so we can directly minimize $|G_j H_{ij} - G_i|$. The determinant of normalized homography, $\hat{H}$ is 1 i.e., $\hat{H} = H/(det|H|)^{1/3}$. We omit the hat for simplicity. We further scale individual linear equations by the confidence in H_{ij} computed as h_{ij} above. Thus, the set of equations are, for each projector pair (i, j),

$$h_{ij}(G_j H_{ij} - G_i) = 0$$

If the number of independent pairwise homographies are larger than the number of projector units, the system of linear equations is over constrained and we solve it in a least squares sense to produce a set of global registration matrices. After pairwise homographies are exchanged, each unit performs the global adjustment separately (in parallel) by treating its own image as the reference plane. Since the projector image plane is not aligned with the screen, at this stage the solution for global image transfer is known up to an unknown homography, H_{i0}, between each projector P_i and the screen.

Euclidean reconstruction In the absence of Euclidean information in the environment, we find H_{i0} by computing the 3D pose of P_j with respect to the screen. Although pose determination in 3D is noisy, our image transfer among projectors remains the same as before, i.e., based on global transfer ensuring a seamless image. Our approach to compute H_{i0} is based on [Raskar and Beardsley 2001]. Note that this computation is performed in parallel by each unit U_i. The steps are as follows.

1. Triangulate corresponding points in P_i and C_i image space to generated 3D points $X = \{x1, x2..\}$.
2. Find a best fit plane, Π, for X. Assign a local coordinate system with the plane Π as the *x-y* coordinate plane Find rotation R^{π}, between local and projector coordinate system.
3. Apply the rotation due to tilt, $X_0 = R^{tilt} R^{\pi} X$, so that X_0 is now world and screen aligned i.e., z coordinate is 0, points in a world vertical plane have the same x coordinate and points in a world horizontal plane have the same y coordinate.
4. Compute homography, H_{i0} , mapping the image of X in P_i framebuffer to the corresponding 2D points in *x-y* plane for X_0 (ignoring *z*-value)
5. Finally for all other projectors, $H_{j0} = H_{i0} H_{ji}$, where H_{ji} is the global homography G_j computed by unit *i*.

Display Finding a suitable projection matrix for rendering for a projector unit involves first computing the region of contribution of the projector on the screen and then re-projecting that complete display extent into the projector image.

In the first step, we find the extent of the illuminated quadrilateral in screen-aligned (X_0) coordinates by transforming the four corners of the normalized projector image plane $[\pm 1 \pm 1]$ using $H_{j0}[\pm 1 \pm 1\ 1]^T$. The union of all projected quadrilaterals in screen X_0 space is a (possibly concave) planar polygon, L. In the Appendix we describe a method to compute the largest inscribed rectangle, S in L.

Each of n units calculates a slightly different rectangle S_i depending on errors in Euclidean reconstruction of X_0. For a global agreement for S, we take the weighted average for each of the four vertices of S_i. The weight is the distance of the projector P_i from that vertex.

Figure 6: *Left : closeup of projector overlap for a four-unit cluster in a 2x2 arrangement. Left-top: projection of ordinary images (without normalization of illuminatioin the overlap area); Left-bottom: a projected grid for the same configuration provides a clear visual indication that the registration after global alignment is good. Middle: four self-contained units displaying overlapping images on a spherical surface with discernible mis-registration after linear estimate of quadric transfer; Right: seamless geometrically corrected display after non-linear refinement of quadric transfer parameters and intensity normalization.*

In the second step, we find the projection of the corners of S in the projector image space using H_{j0}^{-1}. Note that the re-projected corners of S will most likely extend beyond the physical image dimensions of the projector. Since S represents the displayable region, it indicates the extents of the input image, T, to be displayed. Thus, we can find the homography between input image and the projector image, H_{Tj}.

We texture map the input image onto a unit rectangle (of correct aspect ratio) and render with a projection matrix derived from H_{Tj} [Raskar 2000]. The intensity blending is implemented using the alpha-channel in graphics hardware. The blending weights are assigned proportional to the distance to the image boundary.

All the computations are performed symmetrically. After casual installation, it takes about 5 seconds per projector to find pairwise homographies. Global alignment, inscribed rectangle and blending weights computations take an additional 3 seconds. For a six projector unit setup, the total time after casual installation is 30 seconds.

We show several demonstrations in the accompanying video. (a) various projector configurations, (b) very wide aspect ratio, (c) high-brightness display by superimposition of projected texture.

5.2 Curved Display using Ad-Hoc Clusters

The second type of display surface we consider for ad-hoc cluster projection is the quadric. Examples of quadric surfaces are domes, cylindrical screens, ellipsoids, or paraboloids. A solution for quadric surfaces may inspire future work in ad-hoc cluster projection on higher-order and non-planar display surfaces.

In computer vision literature, some relevant work has used quadrics for image transfer [Shashua and Toelg 1997]. In multi-projector systems however, although several approaches have been proposed for seamless multi-projector planar displays based on planar transfer (planar homography) relationships [Raskar 2000; Chen et al. 2000; Yang et al. 2001], there has been little or no work on techniques for parameterized warping and automatic registration of higher order surfaces. This is an omission because quadrics do appear in many shapes and forms in projector-based displays. Large format flight simulators have traditionally been cylindrical or dome shaped, planetariums and OmniMax theaters use hemispherical screens, and many virtual reality setups [Trimension Systems Ltd 2002] use a cylindrical shaped screen.

Alignment is currently done manually. Sometimes this manual process is aided by projecting a 'navigator' pattern [Trimension Systems Ltd 2002; Jarvis 1997]. We propose a completely automatic approach similar to the approach for planar clusters. Parametric approaches lead to reduced constraints on camera resolution, better tolerance to pixel localization errors, faster calibration and finally a simpler parameterized warping process.

Our main contribution here is the re-formulation of the quadric transfer problem and its application for seamless display. See Figure 6.

Simplification of Quadric Transfer Mapping between two arbitrary perspective views of an opaque quadric surface in 3D can be expressed using a quadric transfer function, Ψ. While a planar transfer can be computed from 4 or more pixel correspondences, quadric transfer requires 9 or more correspondences. If a homogeneous point in 3D, X (expressed as a 4×1 vector) lies on the quadric surface Q (expressed as a symmetric 4 by 4 matrix), then $X^T Q X = 0$ and the homogeneous coordinates of the corresponding pixels x in the first view and x' in the second view are related by

$$x' \cong Bx - (q^T x \pm \sqrt{(q^T x)^2 - x^T Q_{33} x})e$$

Given pixel correspondences (x, x'), this equation is traditionally used to compute the 21 unknowns: the unknown 3D quadric $Q = [\, Q_{33} \quad q;\ q^T\ 1\,]$, a 3x3 homography matrix B and the epipole in homogeneous coordinates, e. The epipole is the image of the center of projection of the first view in second view. This form used in Shashua and Toelg [1997] and even in later papers such as Wexler and Shashua [1999] contains 21 variables, 4 more than needed. Our method is based on a simple observation that we can remove part of this ambiguity by defining

$$A = B - eq^T \qquad E = qq^T - Q_{33}$$

and obtain the form we use,

$$x' \cong Ax \pm \left(\sqrt{x^T E x}\right) e$$

Here $x^T E x = 0$ defines the outline conic of the quadric in the first view and A is the homography via the polar plane between the second and the first view. Note that this equation contains (apart from the overall scale) only one ambiguous degree of freedom resulting from relative scaling of E and e. This can be removed by introducing an additional normalization constraint, such as $E(3,3) = 1$. Further, the sign in front of the square root is fixed within the outline conic in the image.

The suggested method to calculate parameters of quadric transfer, Ψ, i.e., $\{A, E, e\}$, directly from point correspondences involves estimating the quadric, Q, in 3D [Shashua and Toelg 1997], [Cross and Zisserman 1998] using a triangulation of corresponding pixels and a linear method. If the internal parameters of the two views are not known, all the calculations are done in projective space after computing the fundamental matrix. However, we noticed that when projectors rather than cameras are involved, the linear method produces very large reprojection errors, in the order of 20 or 30 pixels for XGA projectors. The computation of the fundamental matrix is inherently ill-conditioned given that the points on the quadric illuminated by a single projector do not have significant depth variation in most cases. We instead use known internal parameters and estimated Euclidean rigid transformations. Hence, unlike the planar case, computation of accurate image transfer in this case, involves three-dimensional quantities early in the calculation.

Our algorithm The steps are (a) Compute correspondences between cameras and projectors (b) triangulate and find the equation of the quadric in 3D (c) compute the quadric transfer parameters and (d) use quadric transfer to prewarp the input images.

As in the planar cluster case, each projector P_k, for $k = 1..n$ projects a structured pattern on the quadric, one at a time and is viewed by cameras C_i, for $i = 1..n$, of the n units in the group. Let us consider the problem of computing the quadric transfer, T_{CkPi}, mapping image in camera C_k to projector P_i. However, we cannot directly find quadric transfer, Ψ_{CkPi}, without first finding a rigid transformation , Γ_{Pi}^{Ck}, between P_i and C_k.

The steps for a camera C_k are as follows.

1. Since Γ_{Pi}^{Ci} is known, we triangulate corresponding points in P_i and C_i to get 3D points and a mesh, D_i, of the display surface, and store them in C_i coordinate system.
2. Given 3D points in D_i (projected by P_i) and corresponding pixels observed by camera, C_k, of neighboring units, we find Γ_{Ci}^{Ck}. Then,
$$\Gamma_{Pi}^{Ck} = \Gamma_{Ci}^{Ck} \Gamma_{Pi}^{Ci}$$
3. Fit a 3D quadric Q_i to points in D_i transformed into C_k coordinate system
4. Find parameters of Ψ_{CkPi} from (Q_i, projection matrix M_{Ck} and M_{Pi} and pixel correspondences) using the new formulation of simplified quadric transfer
5. Perform nonlinear refinement of Ψ_{CkPi} , i.e., A_{CkPi},E_{CkPi} and e_{CkPi} to minimize pixel reprojection error i.e., distance between pixels in projector P_i and transferred corresponding pixels from camera C_k

Note that finding the pose of a camera from known 3D points on a quadric is error-prone because the 3D points are usually quite close to a plane. Since we know the camera internal parameters, we first find an initial guess for the external parameters based on a homography and then use an iterative algorithm described in Lu et al. [2000]. We use Powell's method for nonlinear refinement of the reprojection error.

Rendering For rendering, we treat the quadric transfer as a homography via the polar plane (i.e., A) plus a per-pixel shift defined by E and e. Similar to the cluster for planar display, without the aid of any environmental sensor or Euclidean markers in the scene, we exploit this homography along with tilt-sensor reading at U_k to align the display with the world horizontal and vertical. Given the relationship between the input image and the image in C_k, as well as quadric transfer between C_k and all P_i, $i = 1..n$, each unit warps the input image into its own image space via Ψ_{CkPi}. We defer the discussion on computing relationship between input image and image in C_k, plus intensity blending to an upcoming paper. Note that warping an image using a quadric transfer is different than rendering quadric surfaces [Watson and Hodges 1989].

We have implemented the rendering using a simple vertex shader program. For each projector unit, we map the input image as a texture onto a densely tessellated rectangular mesh, and compute the projection of each vertex of the mesh using the quadric transfer.

Results There are two demonstrations in the accompanying video. We show registration of three and four units on a concave spherical segment (Figure 6)and on a convex spherical segment. Our calibration process is relatively slow (about one minute per projector) compared to the planar case. The two time-consuming steps are computing the camera pose from near-planar 3D points using an iterative scheme in Lu et al. [2000], and a non-linear refinement of A, E, e to minimize the pixel reprojection error.

The techniques are ideal for creating single or multi-projector seamless displays without expensive infrastructure using the self-contained projector units. New possible applications are low-cost and flexible dome displays, shopping arcades, cylindrical columns or pillars. The approach and proposed ideas can also be treated as an intermediate step between planar to arbitrary free form shaped displays.

6 Implementation

We have built several prototypes. The oldest is a box which includes an XGA (1024x768) Mitsubishi X80 projector, a Dell Inspiron with ATI Radeon graphics board, a Logitech USB camera (640x480), a tilt-sensor by Analog Devices ADXL202A (angular resolution about 1 degree), and a wireless LAN card. The box is closed with two circular holes in the front, one for the projector and one for the camera. Other components inside the box are a power supply, cooling fan, and a numerical keypad. The only cable coming out of the box is for power. Our newest prototype uses an XGA Plus V1080 projector with a Sony Vaio.

The Euclidean calibration of a single display unit uses an auxiliary camera. First the camera of the display unit and the auxiliary camera undergo full stereo calibration using a checkerboard. The projector then projects onto a blank plane, for two or more orientations of the plane, and the projected points are reconstructed in 3D. Finally the projection matrix of the projector is determined using the correspondences between projector image points and the 3D data.

The display surfaces are mostly everyday surroundings like walls and corners, but we used an Elumens VisionStation dome for the quadric projection to ensure a truly quadric surface.

For the object augmentation in Section 4, the projected augmentation shows wobbles of a few mm for a projector which is 1-2m away, but this seems not particularly detrimental to the viewing experience (there is currently no smoothing on the camera motion).

For the planar transfer in Section 5, the reprojection error is about 2 pixels with pairwise homographies and about 0.3 pixels with global alignment. As seen in Figure 6(left), the images appear registered to sub-pixel accuracy.

For the quadric transfer in Section 5, reprojection error after linear estimation of the 3D quadric directly from point correspondences is about 20 pixels. After estimating the external parameters using method in Lu et al. [2000], the error is about 4 pixels. Finally, non-linear minimization of reprojection using Powell's method reduces the error to about 1.0 pixel. As seen in the video, when displayed on the half-dome, registration is about 1 pixel accurate.

7 Future Directions

Several new modifications and applications are possible with the proposed intelligent projector: (a) a steady-projection projector – a handheld projector that creates a stable image by responding to the geometric relationship between the projector and display surface and the instantaneous acceleration (measured by a tilt sensor); (b) intelligent flash – the projector can provide direction-dependent illumination for images captured by the camera in the unit, (c) shadow elimination – cooperation between overlapping projectors to fill in shadows during augmentation [Jaynes et al. 2001].

A geometrically aware projector can be further improved by considering photometric aspects. The projection can adapt for variations in surface reflectance or surface orientation and an intelligent augmentation may look for suitable surfaces to project on, avoiding low reflectance areas.

The system we have built is still fairly bulky, but the trend is for miniaturization of projectors. The techniques we have presented are ideal for a mobile form factor or temporary deployments of projectors. LEDs are replacing lamps and reflective instead of transmissive displays (DLPs, LCOS) are becoming popular. Both lead to improved efficiency requiring less power and less cooling. Several efforts are already on-going and show a great promise. For example Symbol Technologies [Symbol 2002] has demonstrated a small laser projector (two tiny steering mirrors for vertical and horizontal deflection) and has even built a handheld 3D scanner based on such a projector. Siemens has a built a 'mini-beamer' attachment for mobile-phones [Siemens 2002]. Cam3D has built a 'Wedge' display where a projector can be converted into a 'flat panel' display by projecting images at the bottom of a wedge shaped glass [Travis et al. 2002]. A future mobile projector may double up as 'flat panel' when there is no appropriate surface to illuminate, or ambient light is problematic. Super bright, sharp infinite focus laser projectors are also becoming widespread [Jenoptik 2002] which may allow shape-adaptive projection without focus and ambient lighting problems. In addition suitable input devices are also appearing e.g., Canesta [2002] has built a projected laser pattern on which you can type. The finger movement is detected by IR sensing. Finally novel lamp designs, especially those based on LEDs or lasers are creating smaller, lighter, efficient and long-life solutions.

8 Conclusion

Projectors are showing the potential to create new ways of interacting with information in everyday life. Desktop screens, laptops and TVs have a basic constraint on their size – they can never be smaller than the display area. Hand-helds such as PDAs are compact but the display size is too limited for many uses. In contrast, projectors of the near future will be compact, portable, and with the built-in awareness which will enable them to automatically create satisfactory displays on many of the surfaces in the everyday environment. Alongside the advantages, there are limitations, but we anticipate projectors being complementary to other modes of display for everyday personal use in the future, and to have new application areas to which they are especially suited.

This paper has investigated how to use projectors in a flexible way in everyday settings. The basic unit is a projector with sensors, computation, and networking capability. Singly or in a cluster, it can create a display that adapts to the surfaces or objects being projected on. As a hand-held, it allows projection of augmentation data onto a recognized object, plus mouse-style interaction with the projected data. It works with other units in an ad hoc network to create a seamless display on planar and curved surfaces. The ideas presented provide geometric underpinnings for a new generation of projectors – autonomous devices, easily adapting to operation within a cluster, and adaptive to their surroundings.

Appendix: Largest inscribed rectangle

Here we describe how to compute the largest world axis-aligned rectangle with given aspect ratio, a, inside a (possibly concave) polygon L. L is formed by the union of projected quadrilaterals. The solution for the three unknowns, two for position and one for scale of the rectangle, can be solved by stating a set of linear inequalities for convex polygons but not for concave polygons. We provide a simple re-parameterization of the problem.

Imagine L is drawn in $z = 1$ plane and a rectangle, R, of aspect ratio a is drawn in $z = 0$ plane. A center of projection W=(x,y,z), z in [0,1], to map R into a rectangle S in z=1 plane, is considered valid if S remains completely inside L. We search for a center of projection (CoP) with minimum z (blue circle) because it will create the largest inscribed rectangle.

Consider the forbidden zones for W. Any CoP, that is inside the set of pyramids, created by a point on L with R as base, is invalid (yellow). Since the faces of the pyramid connecting z=0 and z=1 are all triangles, our algorithm computes the intersection of each triangle triple and keeps the one with smallest z value. We only need to consider two types of triangles, those connecting a vertex of L with edge of R and those connecting edge of L with a vertex or R. For n-sided polygon, L, we have $8n$ triangles with an O(n^4) algorithm. This is clearly suboptimal in comparison to the O(n^2) algorithms [Agarwal et al. 1996]. Yet this algorithm is very easy to implement in a few lines of code. Since n is O(number of projectors), runtime of the algorithm is still negligible.

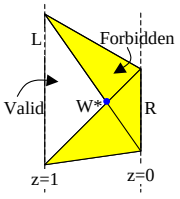


Figure 7: *Search for inscribed rectangle reparameterized as search for a COP*

Acknowledgements We would like to thank Mr Yoshihiro Ashizaki, Mr Masatoshi Kameyama, and Dr Keiichi Shiotani for helping us by providing industrial applications to motivate the work; Paul Dietz, Darren Leigh, and Bill Yerazunis who advised about and built the devices; Shane Booth for sketch in Figure 1; Joe Marks and Rebecca Xiong for many helpful comments; Debbi VanBaar for proof-reading, and Karen Dickie for timely and unrelenting administrative support.

References

AGARWAL, P., AMENTA, N., ARONOV, B., AND SHARIR, M. 1996. Largest Placements and Motion Planning of a Convex Polygon. In *2nd International Workshop on Algorithmic Foundation of Robotics, 1996*, 28–37.

BIER, E. A., STONE, M. C., PIER, K., BUXTON, W., AND DEROSE, T. D. 1993. Toolglass and Magic Lenses: The See-Through Interface. In *Proceedings of ACM SIGGRAPH 1993*, 73–80.

BIMBER, O., GATESY, S., WITMER, L., RASKAR, R., AND ENCARNAO, E. 2002. Merging Fossil Specimens with Computer-Generated Information. In *IEEE Computer*, 32–39.

BROWN, M. S., AND SEALES, W. B. 2002. A Practical and Flexible Large Format Display System. In *The Tenth Pacific Conference on Computer Graphics and Applications*, 178–183.

CANESTA, 2002. Miniature Laser Projector Keyboard. http://www.canesta.com, Cited December 2002.

CHEN, Y., CHEN, H., CLARK, D. W., LIU, Z., WALLACE, G., AND LI, K. 2000. Automatic Alignment of High-Resolution Multi-Projector Displays Using An Un-Calibrated Camera. In *IEEE Visualization 2000*, 125–130.

CHEN, H., SUKTHANKAR, R., WALLACE, G., AND LI, K. 2002. Scalable Alignment of Large-Format Multi-Projector Displays Using Camera Homography Trees. In *Proceedings of Visualization, 2002*, 135–142.

CROSS, G., AND ZISSERMAN, A. 1998. Quadric Surface Reconstruction from Dual-Space Geometry. In *Proceedings of 6th International Conference on Computer Vision(Bombay, India)*, 25–31.

CROWLEY, J., COUTAZ, J., AND BERARD, F. 2000. Things That See. *Communications of the ACM* (Mar.), 54–64.

DAVIS, J. 1998. Mosaics of Scenes with Moving Objects. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 354–360.

HERELD, M., JUDSON, I. R., AND STEVENS, R. L. 2000. Introduction to Building Projection-based Tiled Display Systems. *IEEE Computer Graphics and Applications 20*, 4, 22–28.

HUMPHREYS, G., ELDRIDGE, M., B., I., STOLL, G., EVERETT, M., AND HANRAHAN, P. 2001. WireGL: A Scalable Graphics System for Clusters. In *Proceedings of SIGGRAPH 2001*, 129–140.

JARVIS, K. 1997. Real Time 60Hz Distortion Correction on a Silicon Graphics IG. *Real Time Graphics 5*, 7 (Feb.), 6–7.

JAYNES, C., WEBB, S., STEELE, R., BROWN, M., AND SEALES, B. 2001. Dynamic Shadow Removal from Front Projection Displays, . In *IEEE Visualization 2001*, 152–157.

JENOPTIK, 2002. Laser Projector. http://www.jenoptik.com, Cited December 2002.

KJELDSEN, R., PINHANEZ, C., PINGALI, G., HARTMAN, J., LEVAS, T., AND PODLASECK, M. 2002. Interacting with Steerable Projected Displays. In *Proc. of the 5th International Conference on Automatic Face and Gesture Recognition*, 12–17.

LEVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. In *ACM Transactions on Graphics*, vol. 21, 3, 162–170.

LU, C., HAGER, G., AND MJOLSNESS, E. 2000. Fast and Globally Convergent Pose Estimation from Video Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 6, 610–622.

MAJUMDER, A., HE, Z., TOWLES, H., AND WELCH, G. 2000. Color Calibration of Projectors for Large Tiled Displays. In *IEEE Visualization 2000*, 102–108.

PINHANEZ, C. 2001. The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces. In *Ubiquitous Computing 2001 (Ubicomp"01)*, 12–17.

RASKAR, R., AND BEARDSLEY, P. 2001. A Self Correcting Projector. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 626–631.

RASKAR, R., WELCH, G., CUTTS, M., LAKE, A., STESIN, L., AND FUCHS, H. 1998. The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays. In *Proceedings of ACM SIGGRAPH 1998*, 179–188.

RASKAR, R., BROWN, M., RUIGANG, Y., CHEN, W., WELCH, G., TOWLES, H., SEALES, B., AND FUCHS, H. 1999. Multiprojector Displays using Camera-based Registration. In *IEEE Visualization*, 161–168.

RASKAR, R., WELCH, G., LOW, K., AND BANDYOPADHYAY, B. 2001. Shader Lamps: Animating Real Objects With Image-Based Illumination. In *Rendering Techniques 2001, The Eurographics Workshop on Rendering*, 89–102.

RASKAR, R., VANBAAR, J., AND CHAI, X. 2002. A Low Cost Projector Mosaic with Fast Registration. In *Fifth Asian Conference on Computer Vision*, 114–119.

RASKAR, R. 2000. Immersive Planar Display using Roughly Aligned Projectors. In *IEEE VR 2000*, 27–34.

REKIMOTO, J., AND NAGAO, K. 1995. The World Through the Computer: Computer Augmented Interaction with Real World Environments. In *Proceedings of UIST'95*, 29–36.

REKIMOTO, J., AND SAITOH, M. 1999. Augmented Surfaces: A Spatially Continuous Workspace for Hybrid Computing Environments. In *Proceedings of CHI'99*, 378–385.

REKIMOTO, J. 1999. A Multiple-device Approach for Supporting Whiteboard-based Interactions. In *Proceedings of CHI'98*, 344–351.

SAMANTA, R., ZHENG, J., FUNKHOUSER, T., LI, K., AND SINGH, J. P. 1999. Load Balancing for Multi-Projector Rendering Systems. In *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, 12–19.

SHASHUA, A., AND TOELG, S. 1997. The Quadric Reference Surface: Theory and Applications. In *IJCV*, vol. 23(2), 185–189.

SIEMENS, 2002. Siemens Mini Beamer. http://w4.siemens.de/en2/html/press//newsdesk_archive/2002/-foe02121b.html, Cited December 2002.

SUKTHANKAR, R., STOCKTON, R., AND MULLIN, P. 2001. Smarter Presentations: Exploiting Homography in Camera-Projector Systems . In *International Conference on Computer Vision*, 82–87.

SYMBOL, 2002. Laser Projection Display. http://www.symbol.com/products/oem/lpd.html, Cited December 2002.

TRAVIS, A., PAYNE, F., ZHONG, J., AND MOORE, J. 2002. Flat panel display using projection within a wedge-shaped waveguide. http://ds.dial.pipex.com/cam3d/technology/technology01.html, Cited December 2002.

TRIMENSION SYSTEMS LTD, 2002. . http://www.trimension-inc.com/ , Cited Dec 2002.

UNDERKOFFLER, J., ULLMER, B., AND ISHII, H. 1999. Emancipated Pixels: Real-world Graphics in the Luminous Room. In *Proceedings of ACM SIGGRAPH 1999*, 385–392.

WATSON, B., AND HODGES, L. 1989. A Fast Algorithm for Rendering Quadratic Curves on Raster Displays. In *Proc. 27th Annual SE ACM Conference*, 160–165.

WELLNER, P. 1993. Interacting with paper on the DigitalDesk. *Communications of the ACM 36*, 7, 86–97.

WEXLER, Y., AND SHASHUA, A. 1999. Q-warping: Direct Computation of Quadratic Reference Surfaces. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June, 1999*, 333–338.

YANG, R., GOTZ, D., HENSLEY, J., TOWLES, H., AND BROWN, M. 2001. PixelFlex: A Reconfigurable Multi-Projector Display System. In *IEEE Visualization 01*, 68–75.

YOTSUKURA, T., MORISHIMA, S., NIELSEN, F., BINSTED, K., AND PINHANEZ, C. 2002. Hyper Mask - Talking Head Projected onto Real Object. *The Visual Computer 18*, 2, 111–120.

COVER FEATURE

Combining Optical Holograms with Interactive Computer Graphics

Merging optical holograms with 3D graphical elements can provide an acceptable tradeoff between quality and interactivity: The holographic data provides high-quality but static content, while additional graphical information can be generated, inserted, modified, and animated at interactive rates.

Oliver Bimber
Bauhaus University

Today, many applications for optical holograms exist, including interferometry, copy protection, data storage, and holographic optical elements. A hologram is a photometric emulsion that records interference patterns of coherent light. The recording itself stores the amplitude, wavelength, and phase information of light waves. In contrast to simple photographs, which can record only amplitude and wavelength information, holograms can reconstruct complete optical wavefronts. This results in the captured scenery having a three-dimensional appearance that can be observed from different perspectives.

Museum exhibits often use optical holograms because they can present 3D objects with almost no loss in visual quality. Displaying artifacts virtually removes the need to build physical replicas of the original objects. In addition, this holographic technology can be used to make medical, dental, archaeological, and other recordings—both for teaching and documentation.

Optical holograms are static, however, and lack interactivity. Multiplex holograms offer an apparent exception. Built from multiple vertical-strip holograms that contain recordings of the same scenery at different time intervals, they let observers perceive the recorded scene in motion while moving around the hologram or by spinning a cylindrically shaped version around its principal axis. Multiplex holograms are not, however, truly interactive.

Combining 3D computer graphics with stereoscopic presentation techniques provides an alternative that allows interactivity. Although state-of-the-art rendering methods and graphics hardware can produce realistic images at interactive rates, they do not approach the quality and realism of holographic recordings.

Autostereoscopic displays allow for glass-free observation of computer-generated scenes. These displays can present several perspective views at one time, thus supporting multiple users simultaneously. Resolution and rendering speed, however, decrease with the number of views generated. Holographic images, in contrast, can provide all depth cues—perspective, binocular disparity, motion parallax, convergence, and accommodation—and theoretically can be viewed simultaneously from an unlimited number of positions.

PRINTING AND RENDERING

The two main types of computer-generated holograms, *digital holography* and *electroholography*, have become the focus of several groups researching applications for them.

Digital holography[1] uses holographic printers to sequentially expose small fractions of the photo-

0018-9162/04/$17.00 © 2004 IEEE

Optical Holograms

Both of the two basic optical hologram types—*transmission* and *reflection*—are reconstructed by illuminating them with monochromatic light, as described in Figure A. This approach requires that the light hit the emulsion at the same angle as the reference laser beam used to record the hologram.

Transmission and reflection holograms

To view a transmission hologram, the light source and observer must be on *opposite* sides of the holographic plate. The light is transmitted through the plate before it reaches the observer's eyes. Those portions of the emulsion not recorded or illuminated remain transparent.

To view a reflection hologram, the light source and observer must be on the *same* side of the holographic plate. The light reflects from the plate toward the observer's eyes. As with transmission holograms, unrecorded or nonilluminated portions of the emulsion remain transparent.

These two hologram types have generated a spectrum of variations. Although some holograms can be reconstructed only with laser light, others can be viewed only under white light. Rainbow holograms, one of the most popular white-light transmission hologram types, diffract each wavelength of the light through a different angle. This lets viewers observe the recorded scene from different horizontal viewing positions but also makes the scene appear in different colors when observed from different points.

In contrast to rainbow holograms, white-light reflection holograms can provide full parallax and display the recorded scene in a consistent—but in most cases monochrome—color for different viewing positions.

It is possible to produce both transmission and reflection variations of color white-light holograms. Usually, this process requires recording the same content on several emulsion layers while exposing each layer to laser light with a different wavelength. When reconstructed, each object wave from each layer contributes its individual wavelength, merging together into a multicolor image.

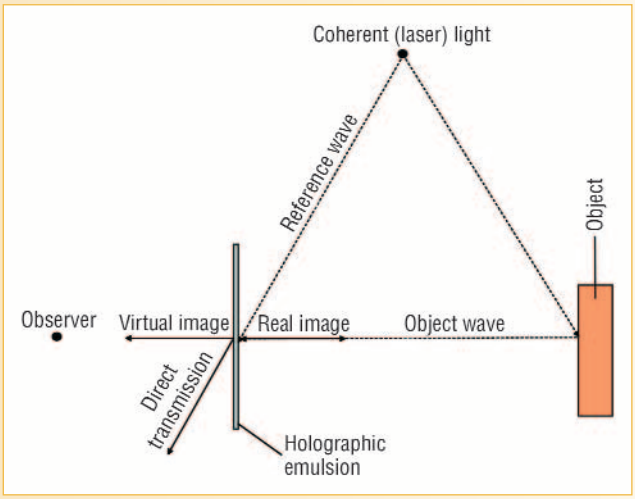


Figure A. Optical holographic recording and reconstruction. A laser beam can be split into two identical beams. One beam, the reference wave, illuminates the holographic emulsion directly while the other beam illuminates the object to be recorded. If the emulsion is illuminated with a copy of the reference wave, it interacts with the recorded interference fringes and reconstructs the object wave, which is visible to the observer.

Parallax displays

These CRT or LCD display screens are overlaid with an array of light-directing or light-blocking elements. Using these elements, the display directs emitted light to both eyes differently—allowing each eye to see individual portions of the displayed image. The observer's visual system interprets corresponding light rays as being emitted by the same spatial point. Dividing the screen space into left and right image portions allows for a glass-free separation of stereo pairs into two or more viewing zones.

Some displays control the parallax array mechanically or electronically, depending on the observer's viewpoint, to direct the viewing zones more precisely toward the eyes. Others generate many dense viewing zones—each showing a slightly different perspective of the rendered scene at the same time. Such displays support multiple users simultaneously but do not yet allow for high frame rates.

One parallax display type includes a barrier that applies an array of light-blocking elements—for example, a light-blocking film or liquid crystal barrier—in front of a screen. The light-blocking elements cover portions of the screen for one eye that are visible from the other eye.

Another parallax example, a lenticular-sheet display, utilizes the refraction of a lens array—consisting of small, cylindrical, prismlike or spherical lenses—to direct the light into the different viewing zones. Images generated with lenticular-sheet displays appear brighter than those displayed on barrier displays. Although prisms and cylinders provide a horizontal parallax only, spherical lenses support a full parallax.

metric emulsion with a computer-generated image. This process results in conventional holograms that display computer-generated content. This technique also can be used to construct large-scale, tiled holograms.[2] Although digital holograms can be multiplexed to display scenes in motion, they remain noninteractive.

Electroholography facilitates the computer-based generation and display of holograms in real time.[3,4] Holographic fringes can be computed by either

- rendering multiple perspective images, then combining them into a stereogram[5] or
- simulating the optical interference and calculating the interference pattern.[6]

Once computed, the system dynamically visualizes the fringes with a holographic display. Since creating a hologram requires processing, transmitting, and storing a massive amount of data, today's computer technology still sets electroholography's lim-

its. To overcome some of these performance issues, researchers have developed advanced reduction and compression methods that create truly interactive electroholograms. Unfortunately, these holograms are small, low resolution, and monochrome. However, recent advances in consumer graphics hardware may reveal potential acceleration possibilities that can overcome these limitations.[7]

PARTIALLY RECONSTRUCTING OBJECT WAVES

Combining optical holograms with 2D or 3D graphical elements can provide an acceptable tradeoff between quality and interactivity. While the holographic content provides high-quality but static content—as described in the "Optical Holograms" sidebar—the combined technology can generate, insert, modify, and animate additional graphical information at interactive rates.

Technically, researchers can use optical combiners such as mirror beam splitters or semitransparent screens to visually overlay the output rendered on a screen over a holographic plate. However, the hologram's reconstructed light will interfere with the overlaid light of the rendered graphics, making an effective combination impossible.

To solve this problem, researchers can reconstruct the object wave only partially, leaving gaps at those places where they have inserted graphical elements. Doing this requires a point light source capable of selectively emitting light in different directions to create an incomplete reference wave. Conventional video projectors provide such light sources and are well suited to viewing white-light reflection or transmission holograms because today's high-intensity discharge lamps can produce a very bright light.

If we use autostereoscopic displays, such as parallax displays, to render 3D graphics registered to the hologram, both holographic and graphical content appear three-dimensional within the same space. This effect can also be achieved by using stereoscopic displays with special glasses that separate the stereo images.

Both reflection holograms, which lack an opaque backing layer, and transmission holograms remain transparent if not illuminated. Thus, they can serve as optical combiners themselves—leading to very compact displays. The illumination and rendering techniques work the same for both hologram types.

Figure 1 shows how a transmission hologram can be combined effectively with a flat-panel lenticular-lens sheet display—a variation of a parallax display that utilizes the refraction of a lens array to direct light into the different viewing zones.

Placing a transmission hologram in front of a mirror beam splitter illuminates it from the front and augments it with graphics from the back. Reflection holograms do not need this beam splitter.

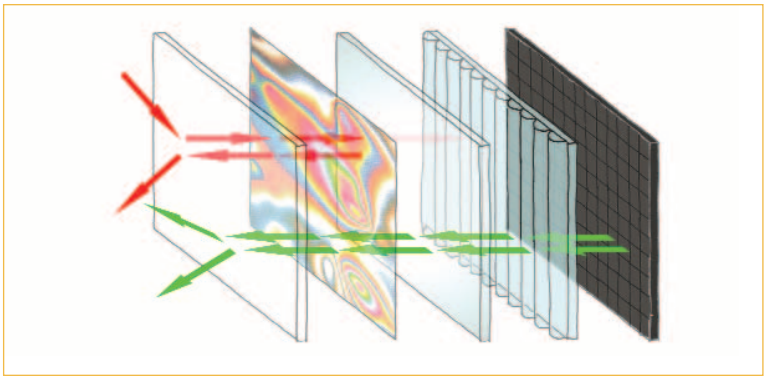


Figure 1. Optical functioning. The explosion model of the optical layers' stacked structure shows, from left to right: glass protection, holographic emulsion, mirror beam splitter for transmission holograms only, lenticular-lens sheet, and LCD array. Reflected light rays (red arrows) reconstruct the object wave on their return through the emulsion. Stereoscopic images (green arrows) pass through all layers until they merge with the hologram.

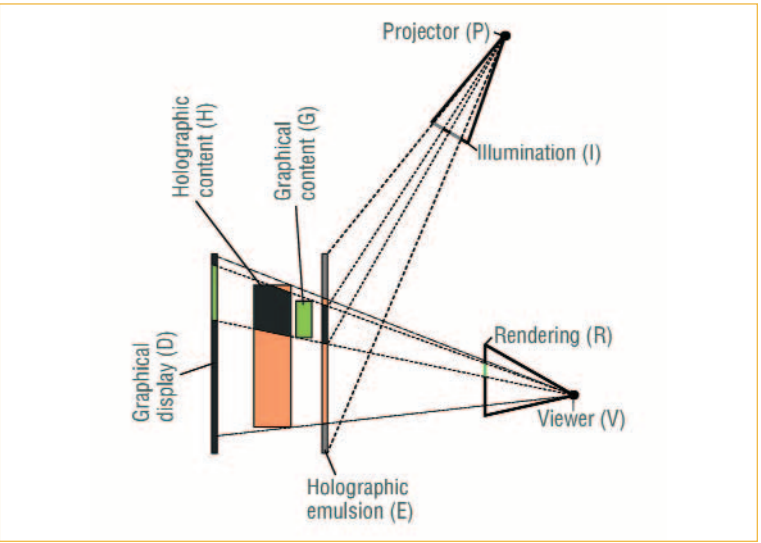


Figure 2. Conceptual sketch of the display constellation. Colored areas on the graphical display and holographic emulsion show which visible image areas comprise the hologram (red) and which comprise the graphics (green).

A thin glass plate protects the emulsion from being damaged and keeps it flat to prevent optical distortion. The lenticular-lens sheet directs the light emitted from the LCD array through all layers toward the observer's eyes. The projected light is transmitted through the first two layers and partially reflected back—either by the beam splitter in combination with a transmission hologram or by a reflection hologram—to reconstruct the recorded content. The screen mostly absorbs the remaining portion of light transmitted through all layers. Figure 2 shows how the system computes the selective illumination on the holographic plate to reconstruct the portion of the hologram not occluded by graphics.

Assuming that information about the holographic content's depth and a description of the graphical content area are available, researchers can use conventional graphics hardware for ren-

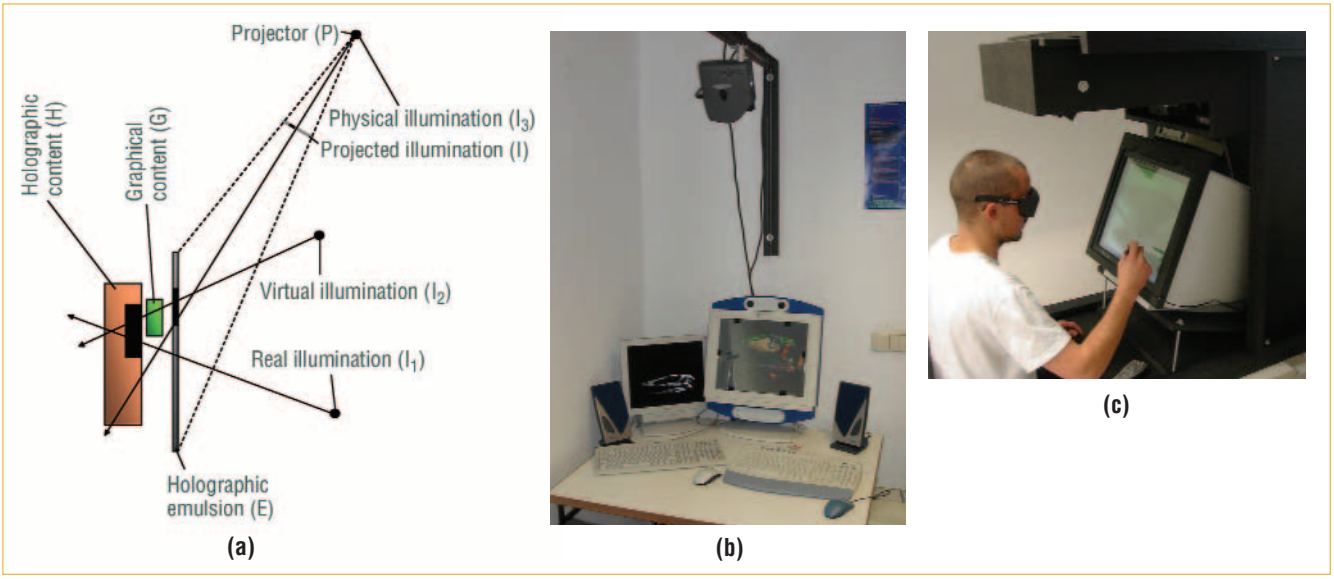


Figure 3. Light-interaction and proof-of-concept prototypes. (a) To simulate virtual shading and shadow effects on the holographic content, the recorded and physical illumination effects must first be neutralized. (b) Autostereoscopic prototype with parallax display and head-finder. (c) Active stereoscopic prototype with CRT monitor, infrared tracking, and touch screen.

dering and illumination. Fortunately, the depth information required to approximate the holographic content's surface can be very coarse. For this example, it consists of a mesh with 5,000 triangles that totals 44.4 Kbytes. This image does not come close to approaching the quality of its optical counterpart but can be rendered easily in real time.

The system geometrically aligns this content during an offline registered step. If the recorded hologram includes optical markers with the actual content, cameras can automatically perform this registration. In addition, the researchers must perform an offline calibration to determine the video projector's extrinsic and intrinsic parameters with respect to the holographic emulsion. If it is mechanically possible to mount the holographic emulsion close to the graphical display, these two planes can be considered identical for the rendering algorithm.

To partially reconstruct the object wave, the system first creates an intermediate texture image by rendering the holographic content from the viewer into the graphics card's depth buffer and filling the card's frame buffer entirely with the lighting source's predefined color values. In addition, the system renders the graphical scene description into depth and stencil buffers, then it clears the frame buffer's stenciled areas and copies the result into the memory block allocated for the intermediate texture image.

If the graphics card provides a render-to-texture option, the read-back operation from the frame buffer into texture memory can be bypassed. The system renders the final illumination image from the projector by drawing the holographic emulsion into the frame buffer and texturing its geometry with the intermediate texture image. The projector then beams the illumination image onto the holographic emulsion.

Next, the system generates a rendering image from the viewer over the off-axis of the graphical display by rendering the content's depth information into the depth buffer and the graphical content's scene description into the depth and frame buffers. The graphical display then shows the rendering image.

LIGHT INTERACTION

The reconstructed object wave's amplitude is proportional to the reference wave's intensity. Besides using an incomplete reference wave for reconstructing a fraction of the hologram, intensity variations of the projected light permit local modification of the recorded object wave's amplitude.

Practically, this means that to create the illumination image, the system uses shading and shadowing techniques to render the holographic content instead of rendering it with a uniform intensity. To do this, the shading effects caused by the real light sources used for illumination during hologram recording, as well as the physical lighting effects caused by the video projector on the holographic plate, must both be neutralized. Next, the influence of a synthetic illumination must be simulated. This can also be done with conventional graphics hardware, as Figure 3a shows. Three intensity images must be rendered.

For the first image, the system renders the holographic content's depth information from the desired relationship between the viewer and the emulsion, using a white diffuse material factor and graphical light sources that generate approximately the same shading and shadow effects as the real light sources used during the holographic recording process. This results in an intermediate texture. The system generates the first image by rendering the holographic emulsion from the perspective of the video projector and texturing it with the intermediate texture. This image simulates the intensity of the recorded object wave.

The system repeats the process to create the second image, this time using graphical light sources to shade the holographic content under the new, virtual-lighting situation. The ratio of the second image to

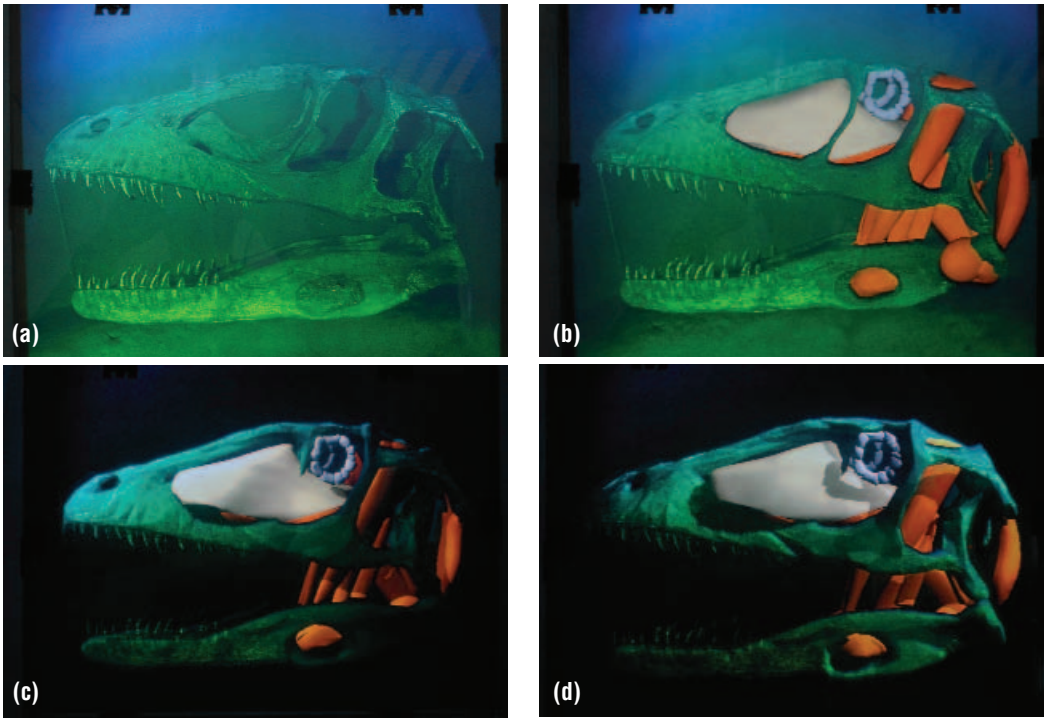


Figure 4. Rainbow hologram of a dinosaur skull combined with 3D graphical elements and synthetic shading effects. (a) The holographic plate illuminated with a uniform light. (b) The plate illuminated only at the portions not occluded by graphical elements such as muscles and other soft tissue. (c) The virtual light source located at the top-left corner, in front of the display. (d) The virtual light source placed at the top-right corner, in front of the display.

the first image represents the required intensity of the reference wave for the holographic plate's emulsion.

For the third image, the system renders the geometry of the holographic emulsion from the projector with a white diffuse material factor and a virtual point light source located at the projector's position. This intensity image represents the geometric relationship between the holographic plate and the video projector as a physical point light source. This third image contains form factor components, such as the square-distance attenuation and angular correlation of the projected light onto the holographic plate, and it neutralizes the physical effects of the projector itself.

The final illumination image can be computed in real time by dividing the second image by the first image and then by the third image via pixel shades. The projection of the resulting image onto the holographic emulsion will neutralize the physical and recorded illumination effects as much as possible and create new shadings and shadows based on the virtual illumination. Again, the system must stencil out the appearance of the graphical content in the final image before displaying it.

During all illumination and rendering steps, the system uses hardware-accelerated shadow-mapping techniques to simulate real and virtual shadow effects on the holographic content's depth information and the graphical scene description. Finally, the system can cast synthetic shadows correctly from all holographic and graphical elements onto all other elements.

This technique's capabilities are clearly limited. It produces acceptable results if the recorded scenery has been illuminated well while making the hologram. However, it cannot neutralize recorded shadows and extreme shading differences. Further, it cannot cancel out recorded color, reflections, and higher-order optical effects.

PROVING THE CONCEPT

The autostereoscopic prototype in Figure 3b and the stereoscopic desktop prototype in Figure 3c were built to validate the proposed techniques. These prototypes consist entirely of off-the-shelf components, including a lenticular-lens-sheet display with integrated head-finder for wireless user tracking or a conventional CRT screen with active stereo glasses, wireless infrared tracking, and a touch screen for interaction. Both prototypes use digital light projectors. A single PC with a dual-output graphics card renders the graphical content on the screen and illuminates the holographic plate on the video projector.

In both cases, the screen additionally holds the front layers—glass protection, holographic emulsion, and optional mirror beam splitter. The remaining two layers shown in Figure 1—the lenticular-lens-sheet and LCD array—already form part of the autostereoscopic display. The display shown in Figure 3c does not need them because the system uses shutter glasses to separate the images for the left and right eye.

A rainbow transmission hologram and a reflection hologram of a dinosaur skull were recorded with a 527.5-nm green laser. Figure 4a shows a photograph of the entire reconstructed hologram, illuminated with a projected uniform light. The system generates illumination and stereoscopic images so that the graphical and holographic content can

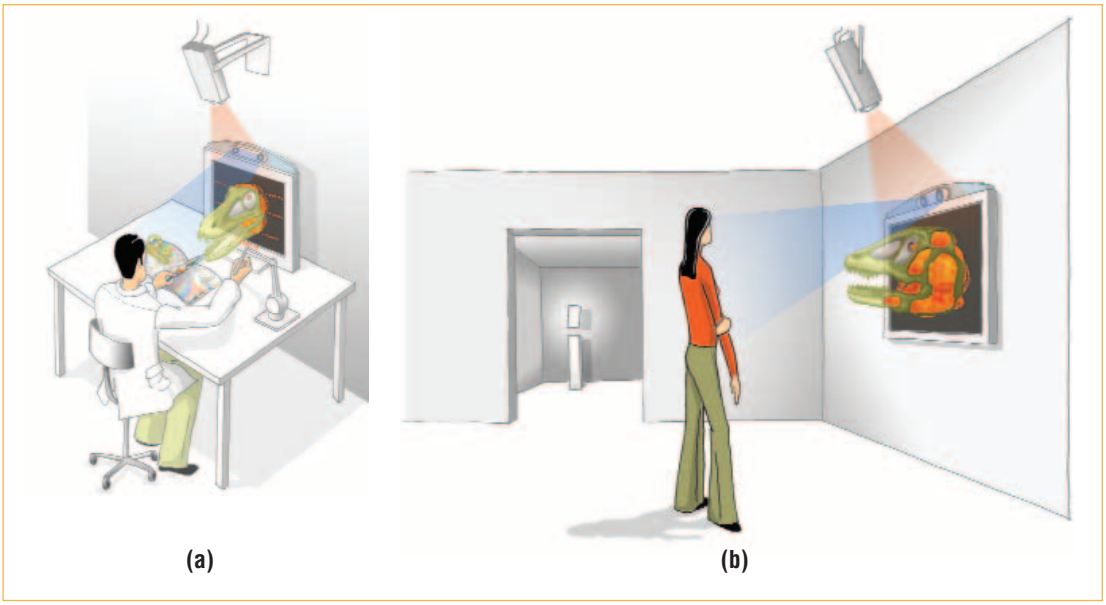


Figure 5. Display variations. (a) Desktop display that can be used in a light-box fashion, with a special input device allowing interaction and providing haptic feedback of holographic and graphical content. (b) Wall-mounted display in a museum environment, with a ceiling-mounted video projector replacing conventional spotlights. An integrated head-finder alerts the display when observers stand in front of it.

be merged within the same space, as Figure 4b shows. Projecting an intensity image that contains new shading and shadow effects instead of a uniform illumination lets the system neutralize most of the diffuse shading recorded in the hologram and produced by the projector. The system can then consistently illuminate the holographic and graphical content, creating matching shading and shadow effects, under novel lighting.

Figures 4c and 4d show the synthetic shading effects caused by a virtual light source. In addition, the image shows virtual shadows cast correctly between hologram and graphical elements. Note that none of the photographs have been retouched. When capturing these images, the system rendered the graphics monoscopically. Although Figure 4 shows the results with a monochrome transmission hologram, reflection and color holograms can achieve the same effects.

Given that hardware-accelerated consumer graphics cards support all these rendering techniques, including shadow mapping and shading, using this method can easily achieve interactive frame rates.

INDUSTRY, SCIENCE, AND EDUCATION

Optical holograms can store a massive amount of information on a thin holographic emulsion. This technology can record and reconstruct a 3D scene with almost no loss in quality. For example, the monochrome hologram shown in Figure 4 comprises a theoretical information content of approximately 96.42 Gbytes. Higher-quality holograms can store a fringe pattern that exceeds one terabyte. Reaching the same quality and performance with computer graphics would require rendering the entire data set in real time on a 1.08 terapixel display—equivalent to 9 million points per square millimeter on a 40 × 30 cm panel. For a color hologram of the same size, the information content multiplies. Moore's law—which asserts that computing power doubles every 18 months—must be applied many times for graphical or electroholographic rendering techniques and displays to reach this quality at interactive frame rates.

A combination of interactive computer graphics and high-quality optical holograms represents an alternative that can be realized today with off-the-shelf consumer hardware. Several commercial online services already offer uncomplicated and inexpensive ways to create color holograms from a set of images or video clips. With this technology, users can create holograms with almost any content—even outdoor scenes.

Archaeologists, for example, already use optical holograms to archive and investigate ancient artifacts.[8,9] Scientists can use hologram copies to perform their research without having access to the original artifacts or settling for inaccurate replicas. They can combine these holograms with interactive computer graphics to integrate real-time simulation data or perform experiments that require direct user interaction, such as packing reconstructed soft tissue into a fossilized dinosaur skull hologram. In addition, specialized interaction devices can simulate haptic feedback of holographic and graphical content while scientists are performing these interactive tasks. An entire collection of artifacts will fit into a single album of holographic recordings, while a light-box display such as that used for viewing x-rays can be used for visualization and interaction, as Figure 5a shows.

This approach has the potential for wide application in other industries. In the automotive industry, for example, complex computer models of cars and components often lack realism or interactivity. Instead of attempting to achieve high visual quality and interactive frame rates for the entire model, designers could decompose the model into sets of

interactive and static elements. The system could record physical counterparts of static elements in a hologram with maximum realism and release computational resources to render the interactive elements with a higher quality and increased frame rate. Multiplexing the holographic content also lets users observe and interact with the entire model from multiple perspectives.

Augmenting optical holograms in museums with animated multimedia content lets exhibitors communicate information about the artifact more excitingly and effectively than with text labels. Such displays can also respond to user interaction. Because wall-mounted variations (Figure 5b) require little space, museums can display a larger number of artifacts.

Clearly, holograms or other replicas cannot substitute for original artifacts because viewing those originals is the main reason patrons visit a museum. If, however, a unique artifact is unavailable or too fragile to be displayed, a hologram offers an enticing alternative by showing the artifact as a high-quality, 3D image that, combined with computer graphics, lets users experience it interactively.

We can use this concept to develop a pallette of display variations. For example, with only minor changes to the presented rendering techniques, arbitrarily curved holograms such as the cylindrical shapes used for multiplex holograms can be supported instead of only simple planar plates. Even without graphical augmentations, projector-based illumination alone has several potential applications. In combination with optical or digital holograms, it can be used to create visual effects. Certain portions of a hologram, for example, can be made temporarily invisible while others can be highlighted.

Emerging large-scale autostereoscopic displays and existing stereoscopic projection screens will let designers upscale this proposed concept. Not only the display, but also the holograms themselves, can be composed from multiple smaller tiles to reach large dimensions and high resolutions.

The foundations of future 3D displays may have been laid in the late 1940s by Dennis Gabor, who, 20 years later, received the Nobel Prize in physics for his invention of holography. Large interactive electroholographic displays with high resolution and full color would provide the ultimate displays—realizing what today is only possible in science fiction. Technological advances will pave the road toward making this fantasy a reality. Intermediate solutions, such as merging optical holograms with 3D graphical elements, represent the first steps along this path. ■

Acknowledgments

Thanks to Tim Frieb and Holowood Holografiezentrum Bamberg for their technical support.

References

1. M. Yamaguchi, N. Ohyama, and T. Honda, "Holographic 3-D Printer," *Proc. SPIE*, vol. 1212, Springer, 1990, p. 84.
2. M.A. Klug, "Scalable Digital Holographic Displays," *Image Processing, Image Quality, Image Capture Systems Conf.* (IS&T PICS 2001), 2001, pp. 26-32.
3. J.S. Kollin, S.A. Benton, and M.L. Jepsen, "Real-Time Display of 3-D Computed Holograms by Scanning the Image of an Acousto-Optic Modulator," *Proc. SPIE*, vol. 1136, *Holographic Optics II: Principles and Applications*, Springer, 1989, pp. 178-185.
4. M. Lucente, "Interactive Three-Dimensional Holographic Displays: Seeing the Future in Depth," *Siggraph Computer Graphics*, special issue on Current, New, and Emerging Display Systems, ACM Press, 1997, pp. 63-67.
5. M. Lucente and A. Tinsley, "Rendering Interactive Images," *Proc. Siggraph 95*, ACM Press, 1995, pp. 387-394.
6. M. Lucente, "Interactive Computation of Holograms Using a Look-Up Table," *J. Electronic Imaging*, vol. 2, no. 1, 1993, pp. 28-34.
7. C. Petz and M. Magnor, "Fast Hologram Synthesis for 3D Geometry Models Using Graphics Hardware," *Proc. SPIE 03, Practical Holography XVII and Holographic Materials IX*, vol. 5005, Springer, 2003, pp. 266-275.
8. F. Dreesen and G. von Bally, "Color Holography in a Single Layer for Documentation and Analysis of Cultural Heritage," *Optics within Life Sciences* (OWLS IV), Springer, 1997, pp. 79-82.
9. F. Dreesen, H. Deleré, and G. von Bally, "High-Resolution Color Holography for Archaeological and Medical Applications," *Optics within Life Sciences* (OWLS V), Springer, 2000, pp. 349-352.

Oliver Bimber *is a junior professor for augmented reality at Bauhaus University, Weimar, Germany. His research interests include display technologies, computer graphics, and human-computer interaction. Bimber received a PhD in engineering from Darmstadt University of Technology, Germany. Contact him at bimber@ieee.org.*

The Virtual Showcase

Oliver Bimber
Fraunhofer Institute for Computer Graphics, Germany

Bernd Fröhlich
Bauhaus University Weimar, Germany

Dieter Schmalstieg
Vienna University of Technology, Austria

L. Miguel Encarnação
Fraunhofer Center for Research in Computer Graphics, USA

We present the Virtual Showcase, a new multiviewer augmented reality display device that has the same form factor as a real showcase traditionally used for museum exhibits.

Intuitive access to information in habitual real-world environments is a challenge for information technology. An important question is how can we enhance established and well-functioning everyday environments rather than replace them by virtual environments (VEs)? Augmented reality (AR) technology has a lot of potential in this respect because it augments real-world environments with computer-generated imagery. Today, most AR systems use see-through head-mounted displays, which share most of the disadvantages of other head-attached display devices.

In this article, we introduce a new projection-based AR display system—the Virtual Showcase (see Figure 1). The Virtual Showcase has the same form factor as a real showcase, making it compatible with traditional museum displays. Real scientific and cultural artifacts are placed inside the Virtual Showcase allowing their 3D graphical augmentation. Inside the Virtual Showcase, virtual representations and real artifacts share the same space providing new ways of merging and exploring real and virtual content. Solely virtual exhibits may also be displayed. The virtual part of the showcase can react in various ways to a visitor, which provides the possibility for intuitive interaction with the displayed content.

Another interesting aspect of our system is its support for two to four simultaneously tracked users looking at the Virtual Showcase from different sides. This feature allows the collaborative exploration of artifacts shown in the Virtual Showcase. These interactive showcases contribute to ambient intelligent landscapes, where the computer acts as an intelligent server in the background and visitors can focus on exploring the exhibited content rather than on operating computers.

As Figure 1 shows, the Virtual Showcase consists of two main parts (the numbers in parentheses correspond to the numbers in the figure): a convex assembly of half-silvered mirrors (1) and a graphics display (2). So far, we've built Virtual Showcases with two different mirror configurations. Our first prototype consists of four half-silvered mirrors assembled as a truncated pyramid (see Figure 1a). Our second prototype uses a single mirror sheet to form a truncated cone (see Figure 1b). We placed these mirror assemblies on top of a projection screen (2). Users can see real objects inside the showcase through the half-silvered mirrors merged with the graphics displayed on the projection screen. We illuminated the showcases' contents with a controllable light source (3) while presenting view-dependent stereoscopic graphics to the observer. For our current prototypes, we use standard shutter glasses (5) controlled by infrared emitters (4). Head tracking is accomplished using an electromagnetic tracking device (6).

Our pyramid-shaped prototype supports up to four viewers simultaneously looking at the showcase from four different sides. Our cone-shaped prototype provides a seamless surround view onto the displayed artifact.

We renderend the scenes in this article on a 500-MHz Pentium III with nVidia Quadro2 graphics. Note that we haven't touched up the photographs. They appear as seen from the viewer's perspective and we rendered them as monoscopic images, although the rendering algorithms normally produce stereoscopic images.

Background and related work

Here we discuss two main areas related to our approach: display technology for AR systems and mirror display technology. In addition, we outline our approach from a geometric optics point of view.

Display technology for augmented reality

Optical or video see-through head-mounted displays (HMDs) are currently the display devices mainly used for AR. However, we can attribute several disadvantages to HMDs:

- deficient resolution and field of view (FOV),
- ergonomic drawbacks due to heavy and cumbersome devices, and

0272-1716/01/$10.00 © 2001 IEEE

- visual perception issues that result from constrained focal lengths.

These disadvantages call for the development of alternative display concepts.

Spatially AR[1] represents such an alternative. Front-projection devices seamlessly project images directly on physical objects' surfaces instead of displaying them somewhere else in the viewer's visual field. On the one hand, this overcomes some of the drawbacks related to HMDs. On the other hand, Spatially AR introduces new problems such as

- shadow-casting of the physical objects and interacting users that results from using front projection and
- restrictions on the display area, which is constrained to the physical objects' size, shape, and color.

Mirror displays

Besides optical see-through HMDs, a few other display systems exist that apply full or half-silvered mirrors to reflect screens. We can categorize them into the following classes: Pepper's Ghosts configurations,[2] reach-in systems,[3-5] real-image displays,[6,7] and varifocal mirror systems.[7,8] All these systems differ from our Virtual Showcase approach.

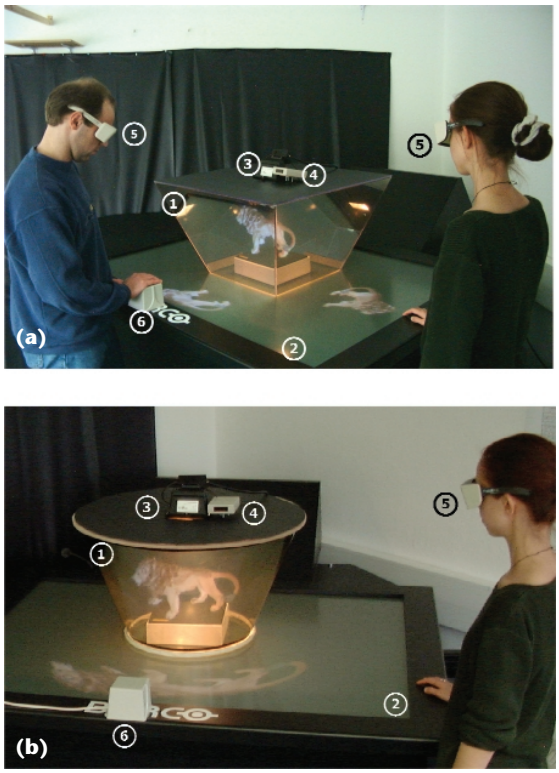
For such systems, the graphics must be transformed before being displayed to ensure that the viewers don't see mirrored or distorted images. For Pepper's Ghosts configurations and reach-in systems, this transformation is trivial (for example, a simple mirror transformation of the frame-buffer content[3] or of the world-coordinate axes[4,5]) because they constrain viewing to restricted areas and benefit from a static mechanical mirror-screen alignment. Some approaches[4] combine this transformation with the device-to-world transformation of the input device by computing a composition map during a calibration procedure and multiplying it with the device-coordinates during the application. Other approaches[5] determine the projection of virtual points on the reflected image plane via ray tracing and then map it to the corresponding frame-buffer location by reversing one coordinate component. Mirror displays that apply curved mirrors, such as real-image displays and varifocal mirror systems, generally don't predistort the graphics before they're displayed. Yet some systems apply additional optics, such as lenses, to stretch the reflected image.[7]

However, if a view-dependent rendering is required or if the mirror optics are more complex and don't require a strict mechanical alignment, these transformations become more complicated. Note that both issues apply to Virtual Showcases.

The challenge for the Virtual Showcase is developing appropriate real-time rendering methods and image deformation techniques that support a view-dependent image presentation for single or multiple users and for planar and curved mirror optics.

Our approach

We refer to the real area in front of a mirror as the *object space* and call the virtual area behind a mirror (that's perceived while looking at it) the *image space*.



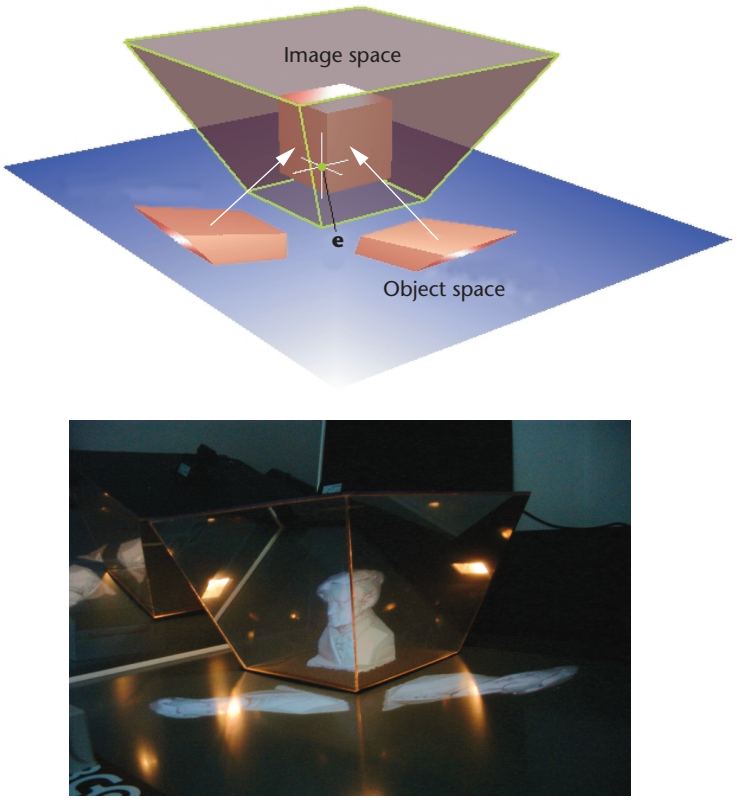
1 The Virtual Showcase prototypes. (a) The pyramid-shaped prototype that supports up to four viewers, and (b) the cone-shaped prototype that supports multiple users and provides a seamless surround-view.

Note that these definitions meet more of the conventions of geometric optics than of computer graphics, where the object space is usually the 3D world-coordinate system and the image space is the 2D projection. Depending on the mirror geometry (for example, planar or curved), the reflection of the object space maps differently into the image space. While the image space of planar mirrors is a stigmatic and affine map of the object space, the image space of curved mirrors is curvilinearly transformed. However, users don't expect to see a mirror while looking at the Virtual Showcase device. Rather, it must appear transparent—like a traditional showcase. Reflections on its surface must be seamlessly combined with the enclosed real objects.

In the case of half-silvered mirrors, the image space unites the reflected image of the object space in front of the mirror and the transmitted image of the real environment behind the mirror.

The starting point for the following rendering techniques requires a description of the scene geometry that's virtually located inside the showcase. Since, from an optics point of view, this geometry is spatially located within the image space, we call it the *image space geometry*.

We aim to transform the image space geometry into the object space in such a way that the reflection of the displayed object space optically results in the expected image space. Therefore, the transformed image space geometry is displayed on a projection plane that's located within the object space (that is, in front of the mirror). Thus, the mirror's reflection neutralizes the image space geometry's transformation. The optically formed image appears orthoscopic, and stereoscopically and perspectively correct and undistorted to an observer.



2 Multiple reflections merged into a single, consistent image space.

Because virtual and real objects coexist in conjunction within the image space, the appearance of the entire image space is known for every given viewpoint.

In the following sections, we describe rendering techniques that can be applied for Virtual Showcases built from planar mirror sections and for showcases built from a single curved mirror piece. We assume that a single planar display device (such as a rear-projection system) is used and that the display device and the mirror optics are defined within the same world-coordinate system. In the upcoming examples, the projection plane matches the world-coordinate system's x/y plane. Note that we illustrate the following methods in an OpenGL context.

Virtual showcases built from planar sections

To transform the known image space geometry appropriately into the object space, we can apply different, slightly modified transformation pipelines for each planar mirror section. With known plane parameters ($\mathbf{n}_r = [a_r, b_r, c_r]$, δ_r) for each mirror, we have to integrate two modifications into the common model-view transformation:[9]

- We apply an additional model transformation between scene transformation $\boldsymbol{M}$ (that is, the accumulation of glTranslate, glRotate, glScale, and so on) and viewpoint transformation $\boldsymbol{V}$ (such as gluLookAt). We do this by multiplying the reflection matrix $\boldsymbol{R}$ with the current transformation matrix—before viewpoint transformation and after scene transformation.
- We apply the common viewpoint transformation matrix $\boldsymbol{V}$ with the reflected viewpoint $\mathbf{e}'$ instead of the actual viewpoint $\mathbf{e}$. We compute the reflected viewpoint by transforming the actual viewpoint over the specific mirror-plane: $\mathbf{e}' = \boldsymbol{R} \cdot \mathbf{e}$.

The reflection matrix is given by

$$\boldsymbol{R} = \begin{bmatrix} 1-2a_r^2 & -2a_r b_r & -2a_r c_r & -2a_r \delta_r \\ -2a_r b_r & 1-2b_r^2 & -2b_r c_r & -2b_r \delta_r \\ -2a_r c_r & -2b_r c_r & 1-2c_r^2 & -2c_r \delta_r \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note that the inverse of $\boldsymbol{R}$ is equivalent to $\boldsymbol{R}$ if $\mathbf{n}_r = [a_r, b_r, c_r]$ is normalized. We can then write the accumulated transformation matrix as $\boldsymbol{P} \cdot \boldsymbol{V} \cdot \boldsymbol{R} \cdot \boldsymbol{M}$, where $\boldsymbol{P}$ denotes the transformation matrix of the applied off-axis perspective projection (for example, generated by glFrustum).

Since an individual reflection matrix exists for each mirror plane, we must apply a modified model-view transformation (with individual $\boldsymbol{R}$ and $\mathbf{e}'$) for each front-facing mirror, respectively. Thus, for a given viewpoint $\mathbf{e}$, the image-space geometry is transformed and rendered multiple times (for each front-facing mirror individually). The example in Figure 2 illustrates this for a truncated pyramid-like Virtual Showcase, assuming that a single observer sees through two mirrors simultaneously.

Because the application of $\boldsymbol{R}$ also reverses the polygon order (which influences front-face determination, lighting, back-face culling, and so on), we have to reverse the polygon order explicitly between transformation and rendering.[9]

A single object displayed in the object space appears exactly once in the image space because convex mirror assemblies unequivocally subdivide the object space into individual reflection zones for each mirror, which don't intersect or overlap. Consequently, convex mirror assemblies provide a definite one-to-one mapping between the object and image spaces.

Observed from $\mathbf{e}$, the different images optically merge into a single consistent image space by reflecting the projection plane, whereby this image space visually equals the image of the untransformed image-space geometry.

The views in Figure 3 can be seen by a single viewer while moving around the showcase, or by two individual viewers while looking at different mirrors simultaneously.

Figure 4 shows an example of an augmented real exhibit, displayed within a Virtual Showcase. We textured and partially projected a virtual representation of the Buddha onto the real statue to demonstrate the precise superimposition of the two environments. We visualized additional multimedia information such as images, movies, and text annotations to complement the exhibit.

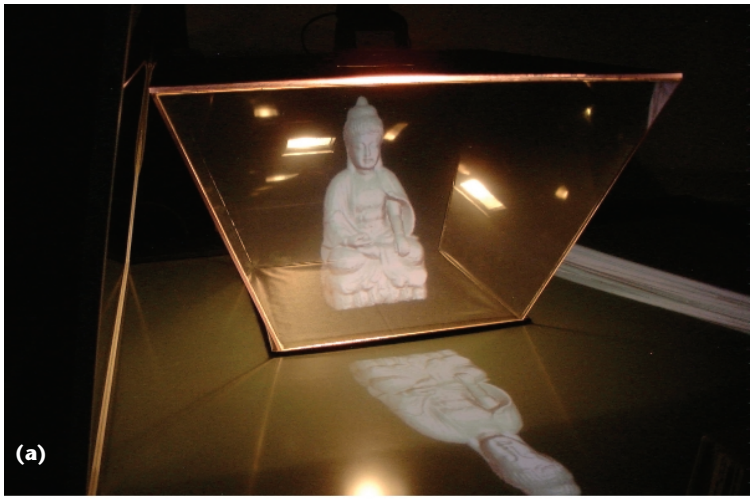
In terms of generating stereoscopic images, we must apply all transformation and rendering steps individually for each eye position of each viewer. This means that to serve four viewers simultaneously, for instance, we must

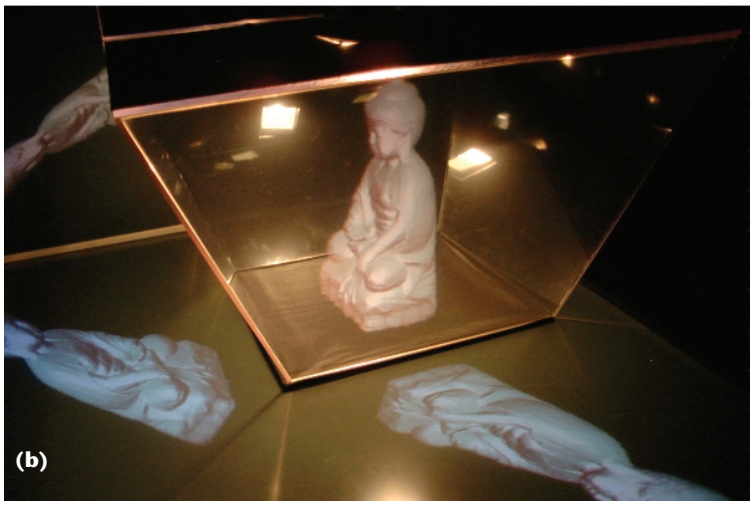
split the transformation pipeline into four subpipes after a common scene transformation $\boldsymbol{M}$. Following the application of the mirror-specific reflection transformations $\boldsymbol{R}$, we can split the subpipelines again to generate the different stereoscopic images for each eye. The subsequent eight subpipelines use different viewpoint transformations with individually reflected viewpoints $\mathbf{e}_i'$, corresponding to each eye position $\mathbf{e}_i$.

Diverging from the example described, we can present individual scenes (that is, different image spaces) to each viewer. In this case, we must apply an individual scene transformation $\boldsymbol{M}$ within each subpipe.

Because of the independence among the transformation subpipes, we can apply parallel rendering techniques (such as using multipipeline architectures). Since $\boldsymbol{R}$ is affine, the modified transformation pipeline doesn't require an explicit access to the image-space geometry. Thus, we can realize it completely independent of the application and even implement it in hardware.

To provide a single, consistent image space (consistent regardless of the viewpoint), we must register the mirror planes precisely. Slight registration errors optically result in gaps between the reflections and therefore in multiple inconsistent image spaces. This problem is comparable to misregistered multiplane projection devices or tiled displays, although for mirror planes the optical aberrations are view dependent.





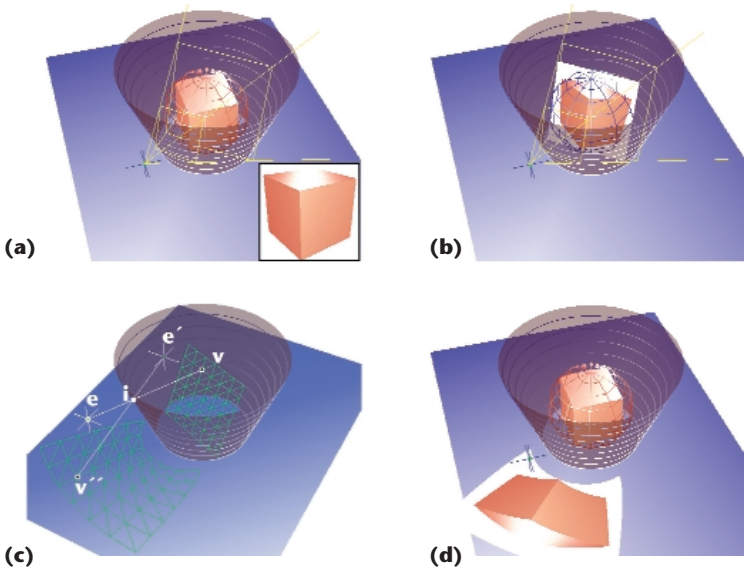
3 Two individual views into the same image space from different perspectives.

4 Augmented real exhibit displayed within a Virtual Showcase.

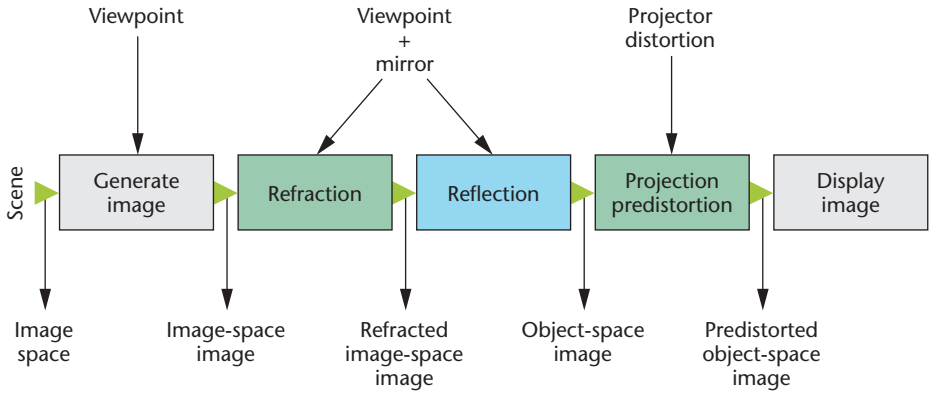
Convexly curved virtual showcases

Building Virtual Showcases from a single sheet instead of using multiple planar sections reduces the calibration problem to a single registration step and consequently decreases the error sources. In addition, the edges of adjacent mirror sections, which can be annoying in some applications, disappear.

However, using curved mirrors introduces a new problem: The transformation of the image space geometry into the object space isn't affine but curvilinear. To map the image-space geometry appropriately into the object space, curved mirrors require per-vertex viewpoint and model transformations. Therefore, we can apply a similar accumulation of transformation matrices as we do for planar mirrors. However, the matrices' parameters differ for each vertex. We've developed several nonaffine geometry transformation techniques for curved mirrors. However, we can transform only highly tessellated image-space geometries with such methods to provide an acceptable curvilinearity deformation behavior.



5 Image-based multipass method for curved optics. (a) Image generation during the first rendering pass, (b, c) image deformation, and (d) image rendering during the second pass.



6 Overview of rendering (blue) and image deformation (green) steps, expressed as a pipeline.

The following two-pass rendering method (see Figure 5) avoids direct access to the image-space geometry and consequently prevents the time-costly transformations of many scene vertices. Our method applies a sequence of intermediate nonaffine image deformations, which we currently consider most efficient for curved mirror displays. It represents a mixture between the extended camera concept[10] and projective textures.[11] While projective textures use a perspective texture matrix to map projection-surface vertices into texture coordinates of those pixels that project onto these vertices, our method projects image vertices directly on the projection surface while the texture coordinate of each image vertex remains constant. This is necessary because curved mirrors yield a different projection (that is, a different projection origin or viewpoint) for each pixel.

The main difference between our approach and the extended camera concept is that the extended camera concept generates a deformed image via ray tracing—that is, each pixel is generated from a modified primary ray. Our method deforms an existing image by projecting it individually for each pixel.

Figure 6 gives an overview of the rendering and image-deformation steps in the form of a rendering pipeline. We'll describe the first and second rendering pass, as well as the reflection transformation, in detail within the following sections. In addition to these components, we built further steps that correct optical distortion caused by in–out refraction and miscalibrated projection systems based on geometric models of these sources of error into this pipeline.

Image generation

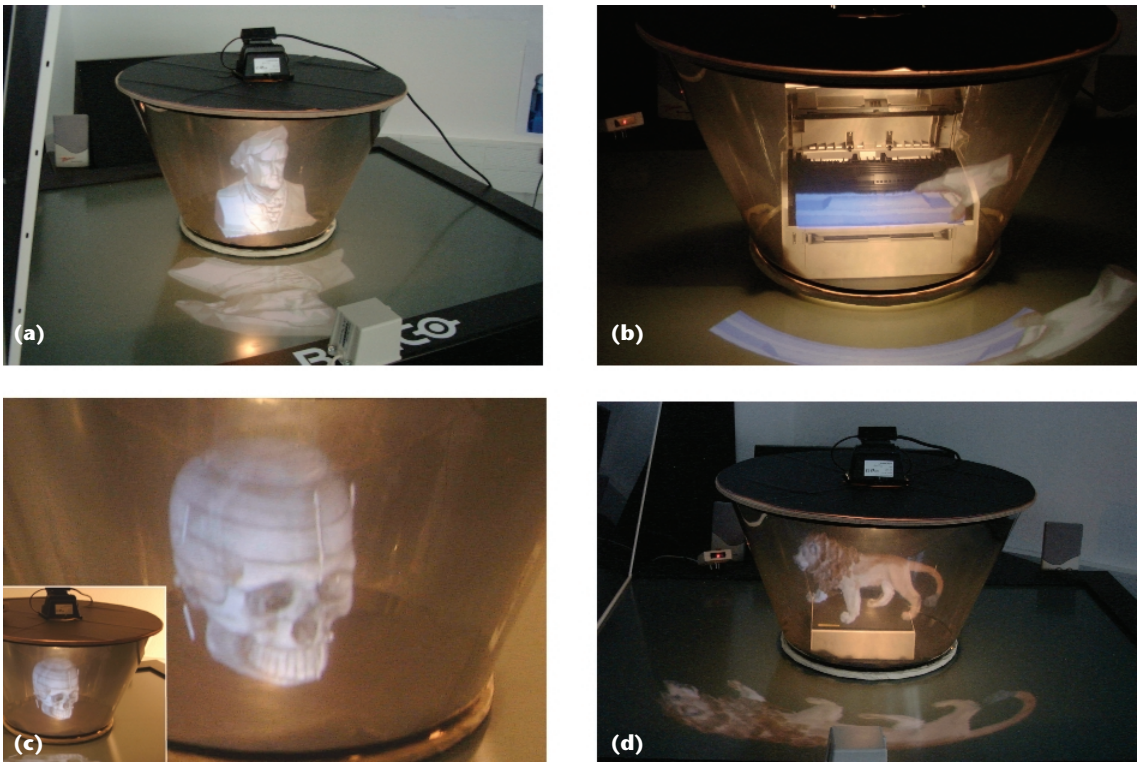
The first rendering pass creates a picture of the image space and renders it into the texture buffer rather than into the frame buffer. To generate this image, we perform an on-axis projection. We determine the size of the projection's viewing frustrum from the image space's bounding sphere. Figure 5a illustrates this for a truncated cone-like Virtual Showcase.

Note that we can exchange the first-pass rendering method. Instead of using a geometric renderer, we can use other techniques (such as image-based and nonphotorealistic rendering, interactive ray tracing, volume rendering, and so on) to generate the image space picture. Besides an ordinary geometric renderer that we used to generate the images in Figures 7a and 7b, Figure 7c shows the application of a volumetric renderer. For the image displayed in Figure 7d, however, we chose a progressive point-based renderer.

Figures 7a to 7d show some results of our rendering approach for curved Virtual Showcases, applying different image-generation methods during the first pass. While Figures 7a and 7c show exclusive virtual exhibits, Figures 7b and 7d illustrate hybrid exhibits (a virtual hand places a virtual cartridge into a real printer and a virtual lion on top of a real base, respectively).

Image geometry and reflection transformation

We now have to transform the image that we generated during the first rendering pass so that its reflection is perceived undistorted within the mirror. To support the subsequent image deformations, we must generate a geometric representation of the image plane. This image geometry consists of a uniformly tessellated grid (represented by an indexed triangle mesh) that's transformed into the current viewing frustum inside the image space so that, if the image is mapped onto the grid, each line of sight intersects its corresponding pixel (see Figure 5b). Therefore, the image is perpendicular to the optical axis and centered with the image space geometry. Finally, we transform each grid point with



7 Examples for different image generation methods: (a, b) geometric rendering, (c) volumetric rendering, and (d) point-based rendering.

respect to the mirror geometry, the current viewpoint, and the projection plane and texture each grid point with the image that we generated during the first rendering pass (see Figure 5c).

For the reflection transformation, we make sure that only visible triangles (that is, the ones with three visible vertices) are rendered during the second rendering pass. Therefore, we set a marker flag for each vertex.

For all grid vertices $\mathbf{v}$, we compute the intersection $\mathbf{i}$ of the geometric line of sight with the mirror geometry (that is, the ray that's spanned by the eye $\mathbf{e}$ and the vertex $\mathbf{v}$). Next, we determine the normal vector $\mathbf{n}$ at the intersection. The intersection point, together with the normal vector, gives the tangential plane at the intersection. Thus, they deliver the plane parameters for the per-vertex reflection transformation that can also be represented by the reflection matrix $\boldsymbol{R}$. We then use the reflection matrix to compute the reflected viewpoint $\mathbf{e}' = \boldsymbol{R} \cdot \mathbf{e}$ and the reflected vertex $\mathbf{v}' = \boldsymbol{R} \cdot \mathbf{v}$. Note that an intersection isn't given if $\mathbf{e}$ and $\mathbf{v}$ are located on the same side of a tangential plane.

Next, we generate a projection matrix $\boldsymbol{P}$ that, given a projection origin and plane parameters, projects a 3D vertex onto an arbitrary plane. In contrast to the projection for planar mirrors, only the beam that projects a single reflected vertex onto the projection plane is of interest. Thus, generating and applying a perspective projection defined by an entire viewing frustum in combination with the corresponding viewpoint transformation (such as glFrustum and gluLookAt) would implicate too much computational overhead. Consequently, this would slow down the image-deformation process. In addition, the viewpoint's reflection becomes superfluous. Since $\mathbf{e}'$, the intersection $\mathbf{i}$, and the final projection of $\mathbf{v}'$ lie on the same beam, we can use $\mathbf{i}$ as the projection origin instead of $\mathbf{e}'$. In doing this, we save the matrix multiplication for the viewpoint transformation and the determination of the viewing-frustum.

The projection matrix is given by

$$\boldsymbol{P} = \begin{bmatrix} \kappa - a_p x & -b_p x & -c_p x & -\delta_p x \\ -a_p y & \kappa - b_p y & -c_p y & -\delta_p y \\ -a_p z & -b_p z & \kappa - c_p z & -\delta_p z \\ -a_p & -b_p & -c_p & -\delta_p \end{bmatrix}$$

where $(\mathbf{n}_p = [a_p, b_p, c_p], \delta_p)$ are parameters of the projection plane, $[x, y, z, 1]$ are the coordinates of the projection center, and $\kappa = [a_p, b_p, c_p, \delta_p] \cdot [x, y, z, 1]$.

Finally, $\mathbf{v}'$ is projected with $\mathbf{v}'' = \boldsymbol{P} \cdot \mathbf{v}'$. Since $\boldsymbol{P}$ is a perspective projection, we must perform a perspective division to produce the correct device coordinates. We can summarize the entire vertex-individual transformation with $\mathbf{v}'' = \boldsymbol{P} \cdot \boldsymbol{R} \cdot \boldsymbol{M} \cdot \mathbf{v}$, whereby $\boldsymbol{M}$ denotes a possible scene transformation. Doing this for all image vertices results in the projected image within the object space (see Figure 5c).

Note that standard graphics pipelines (such as the one implemented within the OpenGL package) only support primitive-based transformations and not per-vertex transformations. Thus, we explicitly reimplemented the transformation pipeline for this approach, bypassing the OpenGL pipeline. Note also that in contrast to the transformation of scene geometry, depth handling isn't required for the image geometry's transformation

Having a geometric representation to approximate the showcase's shape (such as a triangle mesh) provides a

flexible way of describing the showcase's dimensions. However, the computational cost of the per-vertex transformations increases with a higher resolution showcase geometry. For triangle meshes, a fast ray-triangle intersection method[12] is required that also delivers the barycentric coordinates of the intersection within a triangle. We can then use the barycentric coordinates to interpolate between a triangle's three vertex normals and to approximate the normal vector at the intersection.

A more efficient way of describing the showcase's dimensions is to apply an explicit function. We can use this function to calculate the intersections and normal vectors (using its first-order derivatives) with an unlimited resolution. However, we can't express all showcase shapes by explicit functions. Since cones are simple second-order surfaces, we can use an explicit function and its first-order derivative to describe the extensions of our curved showcase. After we transform a geometric line of sight from the world-coordinate into the cone-coordinate system, we can intersect it easily with the cone by solving a quadratic equation created by inserting a parametric ray representation into the cone equation. We compute the normals by inserting the intersection points into the first-order derivative.

Image rendering

During the second rendering pass, the transformed image geometry is finally displayed within the object space, mapping the outcome of the first rendering pass as texture onto its surface (see Figure 5d). Note that we only rendered triangles that provide three visible vertices.

Since the reflection transformations of the previous step delivers device coordinates, and we've defined the projection device and the mirror optics within our world-coordinate system, a second projection transformation (such as glFrustum) and the corresponding perspective divisions and viewpoint transformation (such as gluLookAt) aren't required. If we use a plane projection device, a simple scale transformation suffices to normalize the device coordinates—for example, glScale(1/device_width/2),1/device_height/2,1). A subsequent view-port transformation finally upscales them into the window-coordinate system—for example, glViewport(0,0,window_ width, window_height).

Time-consuming rendering operations that aren't required to display the 2D image (such as illumination computations, back-face culling, depth buffering, and so on) should be disabled to increase the rendering performance. In this case, the polygon order doesn't need to be reversed before rendering, as we noted previously.

Obviously, we have the choice between a numerical and an analytical approach to intersect rays with simple mirror surfaces. Higher order curved mirrors require numerical approximations. In addition, the grid resolution that's required for the image geometry also depends on the mirror's shape. Pixels between the triangles of the deformed image mesh are linearly approximated during rasterization (that is, after the second rendering pass). Thus, some image portions stretch the texture while others compress it. This results in different regional image resolutions. However, our experiments showed that because of the symmetry of our mirror setups, a regular grid resolution and a uniform image resolution achieve acceptable image quality. Since a primitive-based (or fragment-based) antialiasing doesn't apply in deformed texture cases, we can use bilinear or trilinear texture filters instead. As with antialiasing, texture filtering is usually supported by the graphics hardware.

Note that the image's background and the empty area on the projection plane must be rendered in black, because black doesn't emit light and therefore won't be reflected into the image space. For cases in which multiple images must be composed to support multiuser applications, the black background must be color-blended appropriately.

Discussion and future work

Compared to HMDs, using Virtual Showcases for AR tasks provides high and scalable resolution and improved ergonomics because of the lightweight glasses used. In addition, since the reflected graphics appear near their real-world locations, Virtual Showcases support an easier eye accommodation.

The techniques we presented are flexible enough to be smoothly integrated into existing software frameworks and they're general enough to support different hardware setups (that is, projection devices and mirror configurations). Additionally, they take advantage of hardware-implemented rendering pipelines as much as possible. While the rendering passes and per-primitive transformations can be completely executed by graphics accelerators of today's graphics adapters, intermediate per-vertex transformations aren't supported by prevalent rendering pipelines (such as the one implemented in OpenGL). Consequently, they can't benefit from current graphics acceleration hardware. Presently, we've realized these transformations in software—that is, they tax the main CPU and memory bandwidth. Next-generation graphics engines support programmable per-vertex operations, which will allow hardware acceleration of the required per-vertex transformations.

Rather than using a uniform image tessellation, adaptive sampling approaches (such as those derived from view-dependent progressive meshes with frame-to-frame coherence) might result in additional performance resources. We plan to evaluate these techniques and their adaptation to our problem.

We also plan to evaluate other first-pass rendering techniques that generate realistic images of complex scenes and animations at interactive rates. Besides the already integrated volume rendering and splatting algorithms, image-based methods are of particular interest, because they allow realistic display of virtual representations of real artifacts.

Our current prototype uses a dimmable light source for illuminating real artifacts placed inside the showcase. This illumination causes interreflections, which become visible from the outside in some cases. We can solve this problem with upside-down configurations of Virtual Showcases (for example, using a ceiling projection). Additionally, this prevents the observers from directly seeing the projection plane. Furthermore, using a video-projector-based illumination instead of a simple light source (similar to Raskar et al.[1]) allows con-

trolled illumination on a per-pixel basis.

We need to develop interaction techniques and input devices for Virtual Showcases that are unobtrusive and work well for novice users. Since direct interaction is physically restricted by the mirror surfaces, indirect techniques and passive, real-world props seem more appropriate. These interface issues are the most challenging problems for integrating the Virtual Showcase technology into museums and other everyday environments. ■

Acknowledgments

We thank Steve Feiner and David Zeltzer for fruitful discussions and for pointing out additional related work. Thanks to Marc Levoy and Peter Neugebauer for providing and scanning the 3D models. This project is supported by the European Union IST-2001-28610, action line III.1.6 (virtual representation of cultural and scientific objects). Patent pending.

References

1. R. Raskar, G. Welch, and H. Fuchs, "Spatially Augmented Reality," *Augmented Reality: Placing Artificial Objects in Real Scenes* (Proc. First IEEE Workshop Augmented Reality, IWAR 98), R. Behringer, G. Klinker, and D. Mizell, eds., A.K. Peters, Natick, Mass., 1998, pp. 64-71.
2. M. Walker, *Ghostmasters: A Look Back at America's Midnight Spook Shows*, Cool Hand Communications, Boca Raton, Fla., 1994.
3. K.C. Knowlton, "Computer Displays Optically Superimpose on Input Devices," *Bell Systems Tech. J.*, vol. 53, no. 3, March 1977, pp. 36-383.
4. T. Poston and L. Serra, "The Virtual Workbench: Dextrous VR," *Proc. Virtual Reality Software and Technology* (VRST 94), IEEE CS Press, Los Alamitos, Calif., 1994, pp. 111-121.
5. T.E. Weigand, D.W. von Schloerb, and W.L. Sachtler, "Virtual Workbench: Near-Field Virtual Environment System with Applications," *Presence*, vol. 8, no. 5, Oct. 1999, pp. 492-519.
6. C. Chinnock, "Holographic 3D images Float in Free Space," *Laser Focus World*, vol. 31, no. 6, June 1995, pp. 22-24.
7. S. McKay et al., "Membrane Mirror Based Display For Viewing 2D and 3D Images," *Proc. SPIE 3634—Projection Displays V*, vol. 3634, SPIE Press, Bellingham, Wash., 1999, pp. 144-155.
8. H. Fuchs et al., "Adding a True 3-D Display to a Raster Graphics System," *IEEE Computer Graphics and Applications*, vol. 2, no. 7, Sept. 1982, pp. 73-78.
9. O. Bimber, L.M. Encarnação, and D. Schmalstieg, "Augmented Reality with Back-Projection Systems using Transflective Surfaces," *Computer Graphics Forum* (Proc. Eurographics 2000), vol. 19, no. 3, Blackwell Publishers, Oxford, UK, 2000, pp. 161-168.
10. H. Löffelmann and E. Gröller, "Ray Tracing with Extended Cameras," *J. Visualization and Computer Animation*, vol. 7, no. 4, Oct. 1996, pp. 211-228.
11. M. Segal et al., "Fast Shadows and Lighting Effects Using Texture Mapping," *Computer Graphics* (Proc. Siggraph 92), ACM Press, New York, 1992, pp. 249-252.
12. T. Möller and B. Trumbore, "Fast, Minimum Storage Ray-Triangle Intersection," *J. Graphics Tools*, vol. 2, no. 1, Jan. 1997, pp. 21-28.

Oliver Bimber *is a scientist at the Fraunhofer Institute for Computer Graphics and is a PhD candidate at the Darmstadt University of Technology, Germany. His research interests are in display technologies, rendering, and human–computer interaction for mixed realities. He received a BS in commercial computing from the Dundalk Institute of Technology, Ireland and a Dipl.Inform (FH) in scientific computing from the University of Applied Science Giessen, Germany.*

Bernd Fröhlich *is a professor for Virtual Reality Systems with the media faculty at the Bauhaus University Weimar in Germany. His recent work focuses on input devices, interaction techniques, display systems, and support for tight collaboration in local and distributed virtual environments. He received his MS and PhD in computer science in 1988 and 1992, respectively, from the Technical University of Braunschweig, Germany.*

Dieter Schmalstieg *is an assistant professor at the Interactive Media Systems group at Vienna University of Technology, Austria, where he directs the Studierstube augmented reality research project. His research interests include virtual environments, augmented reality, 3D user interfaces, and real-time graphics. He holds a PhD and Habilitation degree from Vienna University of Technology.*

L. Miguel Encarnação *is the head of the Human Media Technologies Department of the Fraunhofer Center for Research in Computer Graphics in Providence, Rhode Island, and is an adjunct professor in computer science at the University of Rhode Island. His research interests include next-generation user interfaces, mixed reality technologies, and advanced distributive learning environments. He received his PhD in computer science in 1997 from the University of Tübingen, Germany.*

Readers may contact Bimber at Fraunhofer Center for Research in Computer Graphics (CRCG), 321 S. Main St., Providence, RI 02903, email obimber@crcg.edu.

For further information on this or any other computing topic, please visit our Digital Library at http://computer.org/publications/dlib.

Superimposing Pictorial Artwork with Projected Imagery

Oliver Bimber, Franz Coriand, Alexander Kleppe, Erich Bruns, Stefanie Zollmann, and Tobias Langlotz
Bauhaus University Weimar

We present a novel approach for using pictorial artwork as information displays and show how to combine almost any kind of computer-generated visual information directly with the painted content.

Working high above the floor of the Sistine Chapel in the Vatican of Rome, between 1509 and 1512 Michelangelo Buonarroti painted some of the finest pictorial images of all time. On the ceiling of the papal chapel, he created a masterpiece fresco that includes nine scenes from the book of Genesis. Among them is the famous *Creation of Adam* scene—showing God touching Adam's hand. In 1510, an initial study led Michelangelo to draw the Adam figure as a sanguine on a piece of paper. Today, this early drawing is displayed at the London British Museum.

Around 1518 Jacopo Pontormo painted yet another scene from the book of Genesis, called *Joseph and Jacob in Egypt*. It decorated the bedchamber of Pier Francesco Borgherini for many years and is now being displayed at the London National Gallery. This oil painting made headlines after art historians discovered incredible underdrawings beneath the top paint layer. The underdrawings show that the artist had decided to flip the background design after starting work and simply overpainted his initial approach.

Such underdrawings have been found in many Renaissance and Baroque paintings. In 1634, for example, Rembrandt van Rijn painted a self-portrait that was later retouched by one of his students to feature a Russian nobleman. The original portrait was hidden under layers of paint for more than 300 years, until it was uncovered recently by art restorers. It sold for nearly seven million British pounds.

Pictorial artwork, such as these examples, can tell interesting stories. The capabilities of museums to communicate this and other information, however, are clearly limited. Text legends and audio guides can mediate facts, but offer little potential for presenting visual content such as embedded illustrations, pictures, animations, and interactive elements.

Also because of the difficult economic situation, edutainment is becoming an important factor for museums. By applying new media technologies—such as computer graphics, virtual reality, and augmented reality—exhibit-oriented information might be communicated more effectively, but certainly in a more exciting way.

In this article we describe a novel technological approach, a mathematical model, a real-time rendering algorithm, and examples of presentation techniques for integrating almost any kind of visual information directly into pictorial artwork. Our system displays such information while keeping the observers' attention on the original artifact and doesn't require additional screens.

Technical approach

A seamless and space-efficient way for integrating visual information directly into pictorial artwork is to use the artwork as an information display (see Figure 1). The display can serve as a diffuse projection screen, and we can apply conventional video projectors to show computer graphics together with the painted content. To perceive the projected imagery in the correct colors and intensities, however, requires that the influence of the underlying physical color pigments is neutralized. In most situations, this isn't possible if untreated images are simply projected directly onto arbitrarily colored surfaces. The problem is that the projected light interacts with the color pigments on the canvas and is partially absorbed if the pigment's color isn't fully white.

To solve this problem, we use a new film material that has two properties: first, it's completely transparent and second, it diffuses a fraction of the light projected onto it. The film consists of an even deposition of fine particles on both sides of a polyester base with no visible artifacts. It was used for creating special effects in Hollywood movies such as *Minority Report* and *Paycheck* and sells for $350 per square foot. Initial

1070-986X/05/$20.00 © 2005 IEEE

Published by the IEEE Computer Society

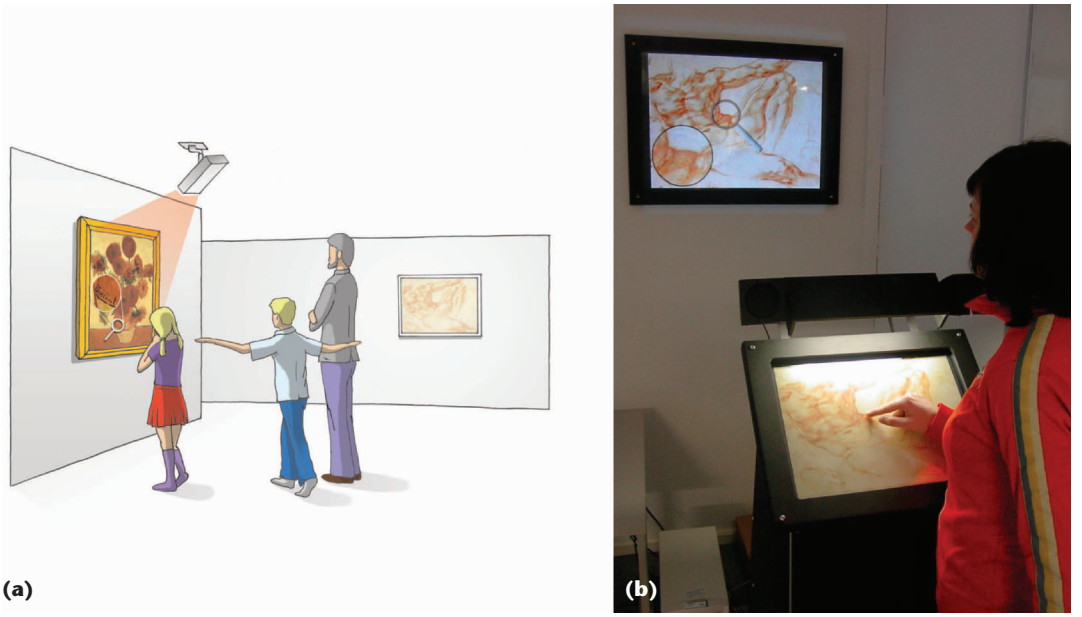


Figure 1. Using pictorial artwork as information displays: (a) concept sketch and (b) first prototype.

measurements have revealed that on average 20 percent (± 1 percent) of the light that strikes the film is diffused while the remaining fraction is transmitted toward the canvas. This 0.1-mm thin, transparent film can be seamlessly overlaid on the canvas (with or without direct contact) by integrating it into the frame that holds the artwork. We use BenQ 7765PA 1,100 ANSI lumens extended graphics array (XGA) digital light projectors to display images on film and canvas.

Mathematical model

In this section we describe how light interacts with a textured surface through the film. If a light beam with incident radiance L is projected onto the transparent film material located on top of the original artwork, a portion d of L is directly diffused from the film while the remaining portion t of L is transmitted through the film. The transmitted light tL interacts with the underlying pigment's diffuse reflectance M on the canvas, and a color-blended light fraction tLM is diffused. The portion $tLMt$ is then transmitted through the film, while the remaining part $tLMd$ is reflected back toward the canvas, where it is color blended and diffused from the same pigment again. This ping-pong effect between film material and canvas is repeated infinitely while for every pass a continuously decreasing amount of light is transmitted through the film that contributes to the resulting radiance R. Mathematically, we can express this as an infinite geometric series that converges toward a finite value. The same is true for the environment light with incident radiance E emitted from uncontrollable light sources. Because these light sources also illuminate the canvas and the film material, we must also consider the environment light's contribution to R.

Figure 2a (next page) shows this process as a sequence diagram. Note that in contrast to this conceptual illustration, normally no physical gap exists between the film material and canvas, and the light interaction occurs at the same spot.

If all parameters (L, E, M, t, and d) are known, we can compute the resulting radiance R that's visible to an observer in front of the canvas:

$$R = (L+E)d + (L+E)t^2 M \sum_{i=0}^{\infty} (Md)^i$$
$$= (L+E)\left(d + \frac{t^2 M}{1-Md}\right) \quad (1)$$

Now that we know R,which we expect to see, we need to solve Equation 1 for L:

$$L = \frac{R}{\left(d + \frac{t^2 M}{1-Md}\right)} - E \quad (2)$$

Equation 2 allows computing the incident radiance L that needs to be projected onto the film and the canvas to create the known result R. The radiant intensity I of the projector to create L is related to a discretized pixel value and is given by

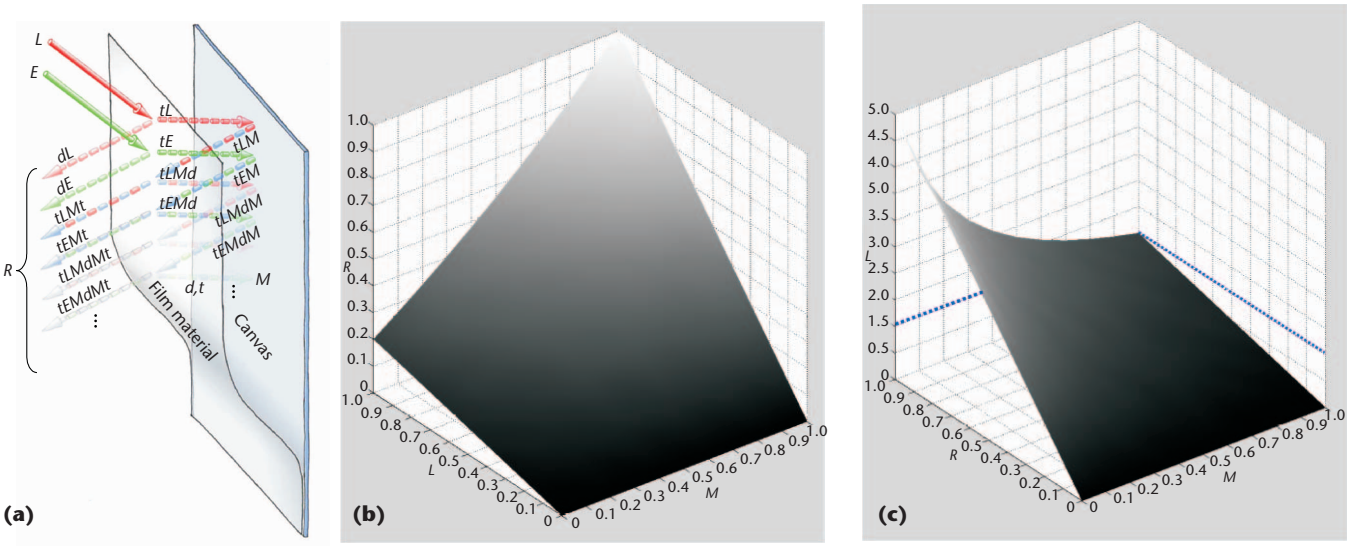


Figure 2. Properties of the film material. (a) Interaction of projected light and environment light with the canvas and the film material (sequence diagram). Visualization of (b) Equation 1 and (c) Equation 2 over R, L, and M (without E, t = 80 percent and d = 20 percent).

$$I = L\frac{r^2}{\cos\alpha}s \qquad (3)$$

$$L = \sum_{i=1}^{n} L_i, \quad I_i = L_i\frac{r^2}{\cos\alpha_i}s_i \qquad (4)$$

where $r^2/\cos\alpha$ are the form factor components: square distance attenuation and angular correlation of the projected light onto the canvas. The additional factor s allows scaling the intensity to avoid clipping and to consider the simultaneous contributions of multiple projectors.

Our approach has clear limitations, which are illustrated in Figures 2b and 2c. Not all radiances R can be produced under every condition. If M is dark, most of L and E are absorbed. In an extreme case, the corresponding pigment is black ($M = 0$). In this case the right term of Equation 1 is canceled out. The remaining left term—which depends on the diffusion factor d of the film material—sets the boundaries of the final result that can be produced. The intersection of the surface with the RL-plane in Figure 2b illustrates these limitations. Consequently, in the worst case of our example, only 20 percent of R can be generated. This situation is also reflected in Figure 2c as the intersection of the surface with the LR-plane. Here we want to assume that $sr^2/\cos\alpha = 1$, which results in $L = I$. For a single video projector, the projected radiance L and consequently the radiant intensity I cannot exceed the normalized intensity value of 1 (dotted line). But for creating most of the resulting radiance values, L and I must be larger. This situation worsens for $r^2/\cos\alpha > 1$ and for $E \to 1$ or $M \to 0$.

However, the contributions of multiple (n) projectors allow displacing this boundary with If all projectors provide linear transfer functions (for example, after gamma correction) and identical brightness, $s_i = 1/n$ balances the load among them equally. However, s_i might be decreased further to avoid clipping and to consider differently aged bulbs.

For Figures 2b and 2c, we don't consider the environment light E and set it to zero. Additional environment light would simply shift the surface in Figure 2b up on the R-axis, and the surface in Figure 2c down on the L-axis. Note that our mathematical model must be applied to all color channels (such as red, green, and blue for projected graphical images) separately.

Real-time color correction

We implemented Equations 2 through 4 as a pixel shader to support real-time color correction. We used Nvidia's Cg framework for fragment processing on an MSI GeForce FX 5600 TD graphics board. Figure 3 illustrates the rendering process based on an example of Michelangelo's *Creation of Adam*. Although we'll use this example to explain the process, it's universal and can be applied with arbitrary background images.

In our example, a copy of the original Adam drawing serves as a background image. Our goal is to overlay it entirely with a registered view on the actual ceiling fresco of the Sistine Chapel.

The first step of the rendering process is to create an input image I_i, which needs to be overlaid.

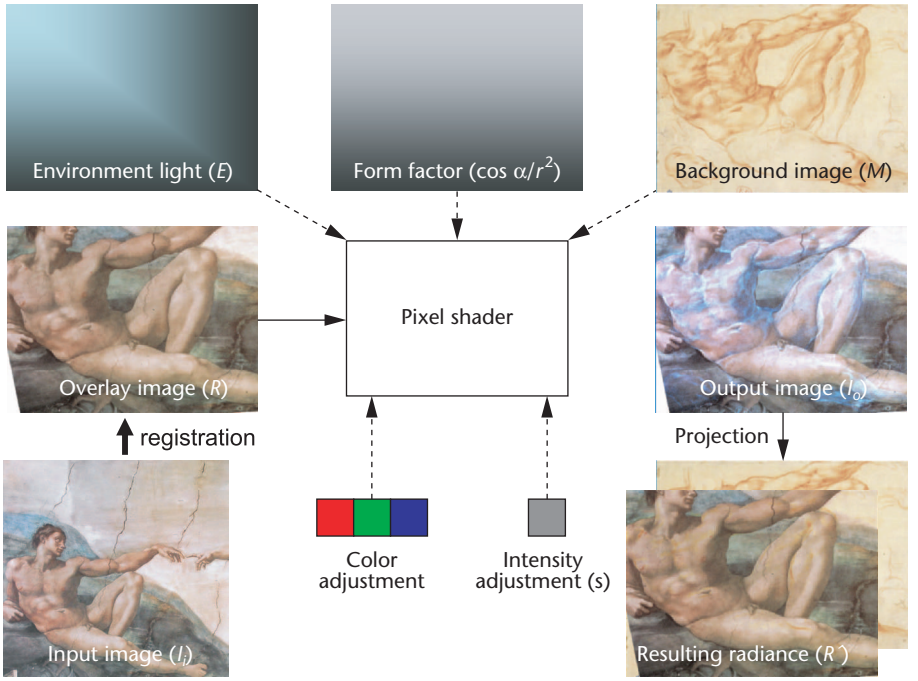


Figure 3. Real-time color-correction process.

This image can be dynamically rendered (as part of a real-time animation or an interactive experience), played back (frames of a prerecorded movie), or static (a photograph of the corresponding ceiling portion—as is the case in our example).

The input image must be registered to the physical drawing. Registration is achieved by texture mapping I_i onto a predistorted image geometry that's precisely aligned with the physical drawing. The amount of distortion depends on the geometric relation between the video projector and the canvas. The distortion can be simple keystone deformations to more complex curvilinear warping effects (for example, if lens distortion of the projector has to be neutralized).

Several automatic approaches apply video cameras and image analysis to align multiple images of tiled projection screens.[1,2] A structured light registration benefits from controllable features, such as projected grid edges or Gaussian matchpoints that can be easily detected in the camera views with a subpixel precision.

In our case, however, we have to register a digital representation of the artistic content against its physical representation on the canvas, rather than registering one projected structured light image against another. To detect nonstructured artistic features (such as fine lines) in the artwork and register them automatically against the corresponding features in the digital content represents an important task of computer vision—especially if projected pixels and physical pigments must align precisely on the canvas. We're critical about the feasibility and precision of an automatic method for our problem and have decided to solve it with a manual registration process that benefits from the resolution of the human visual system. Because the following steps have to be performed only once, we believe that they represent an acceptable solution.

Our manual registration process lets users interactively identify 2D correspondences between artistic features in the background image *M* within the image space—that is, an image of the drawing displayed on a control screen—and the physical drawing on the wall. This is done using a 2D input device, such as a conventional mouse whose pointer is visible on the control screen and as projection on the canvas.

We use dual output graphics card and an additional video signal splitter to drive the control screen and one or two projectors. The result of this feature matching is a set of 2D vertex fiducials with their corresponding texture coordinates within the image space. The fiducials are Delauny triangulated, and the system uses texture coordinates to map the correct image portions of *I* onto the image geometry. This results in the overlay image *R*. Here we should stress again that a precise correspondence between *R* and *M* is important to achieve qualitatively good results.

In our experiments, the measurement of 50 to

70 fiducials proved sufficient. In contrast to uniform grid methods normally applied for projector alignment, this general geometric registration allows correlating arbitrary artistic features in the physical drawing with the corresponding pixels of M in the image space.

Thus, it provides an accurate matching that can be regionally improved further if linear interpolation within single-grid triangles fails to be precise enough. The registration results don't change if the projector and background image are fixed. Before we can render R, we must enable the color-correction pixel-shader. Five parameters are passed to the pixel shader to ensure that Equations 2 through 4 can be computed.

The first parameter is the environment light E in the form of intensity texture that has the same size as R. It contains intensity values that represent the uncontrollable lighting situation on the canvas. We can determine the intensity values by measuring the irradiance of the environment light with a light meter for a discrete number of sample spots on the canvas' surface—resulting in the lux (lx) values E'. These values have to be normalized to an intensity space that ranges from 0 to 1 so the shader can process them. To do this, we measure the same spots again, but this time the highest intensity possible (for example, a white image) is projected onto the light meter, which is measured in addition to the environment light. These measurements are equivalent to the total irradiance $T' = L' + E'$, and also carry the unit lux.

Because we know that $L' = T' - E'$ is equivalent to the scaled intensity value $\cos\alpha/r^2$, we can convert the measured radiance of the environment light from lux into the normalized intensity space with $E = E'/(T' - E')\cos\alpha/r^2$. To approximate the intensity values for the entire image area in E, all the measured spots are mapped onto the image space and are Delauny triangulated. The values for the remaining pixels are linearly interpolated by the graphics card while rendering the Delauny mesh. Note that E is constant if the environment light doesn't change. For reasons that we describe next, we can assume that $\cos\alpha/r^2$ is constant and equals 1.

The second parameter is the form factor that represents the geometric relation between the video projector as a point light source and the canvas. Because it doesn't change for a fixed relation between projector and canvas, we can precompute it and pass it to the pixel shader in the form of an intensity texture with the same dimensions as E and R.

Like Raskar's Shader Lamps,[3] the graphics pipeline can produce this texture—a geometric model of the canvas is rendered with a white diffuse reflectance from the viewpoint of the projector. Attaching a virtual-point light source (also with a white diffuse light component) to the position of the projector and enabling square distance attenuation produces intensities proportional to $\cos\alpha/r^2$. The pixel shader can compute the required reciprocal. Practically (that is, for normal-sized canvases and nonextreme projector configurations), we found that the form factor is constant over all pixels. It's then contained by the intensity adjustment parameter s.

The third parameter is the background image M. It also has the same dimensions as E and R. This image can be generated by, for example, scanning the color values or taking a photograph of the original drawing under uniform illumination.

The fourth and fifth parameters contain color and intensity adjustment values that let users fine-tune the video projector's individual color response and prevent intensity clipping. These parameters also help users adopt color drifts that they can introduce while capturing the background image and allow users to consider the contributions of multiple projectors. Note that gamma correction must be applied in advance. This mostly occurs with projectors with nonlinear transfer functions as well as projectors with linear transfer functions that apply a de-gamma mapping on the video input. Gamma correction is usually supported by the graphics hardware and the video driver, but the pixel shader can also carry it out. We adjust these values manually, but the support of automated methods for color matching multiple projectors[4] is imaginable.

The output image I_o is the final result of this rendering process and will be displayed by the video projector. If projected geometrically correctly onto the drawing, the result R' will be visible. Both images—R and R'—are mostly identical, except for slight artifacts that appear because of previously discussed limitations. Figure 4 shows the results of our example projected onto a real drawing with a single video projector.

Apparently, the underlying drawing can be made partially or completely invisible to display the graphical overlay in the correct colors on top of it. Figures 4e through 4h show close ups in which diverging body parts (such as belly and knee) are overdrawn and displaced by the projection.

Some intensities and colors that are required

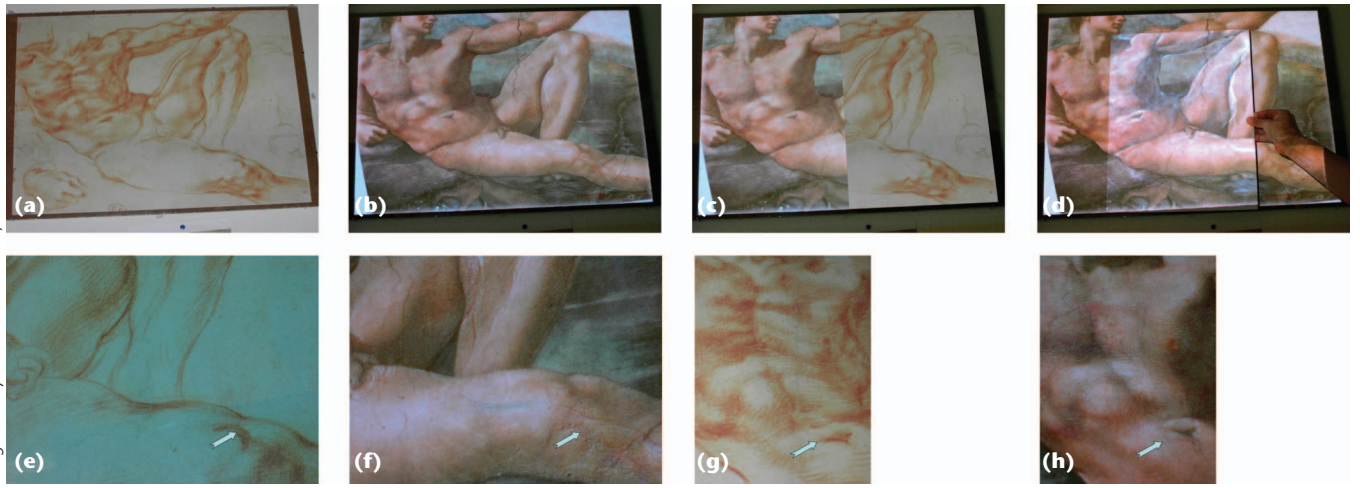


Image courtesy of the British Museum, London

Figure 4. Results of the color-correction process with a single projector on a real drawing. (a) Real drawing (64 × 48 cm) under environment light. (b) Output image emitted onto drawing. (c) Partially augmented drawing. (d) Output image on a white piece of paper. (e–h) Close ups. While the upper body coincides with the drawing and painting, Michelangelo modified the lower body. The arrows indicate the displaced knee and belly sections. They point at the same spot on the drawing.

to neutralize the underlying color pigments can't be achieved by a single video projector. The worst case is to turn a black pigment on the canvas into a white color spot. Figures 2b and 2c illustrate that in such a case the required intensity can easily exceed the boundary of 1 in our normalized intensity space. The pixel shader clips these values to 1, which results in visible artifacts.

The simultaneous contributions of multiple projectors can reduce or even eliminate these effects. Figure 5 shows the extreme case of an input image that has no geometric correspondences to the underlying background image. In addition, both projections together create bright colors (the sky) on top of dark color pigments on the canvas. In Figure 5a, a single projector is used. Intensity values that are too large are clipped and result in visible artifacts. Balancing the intensity load between two projectors reduces these artifacts clearly (see Figure 5b). Figures 6 through 8 (next page) show more results for other cases.

Because of the hardware acceleration of today's graphics cards, we can easily perform the color-correction process in real time. Note that none of the photographs in this article have been retouched. Slight variations in color and brightness are because of different camera responses.

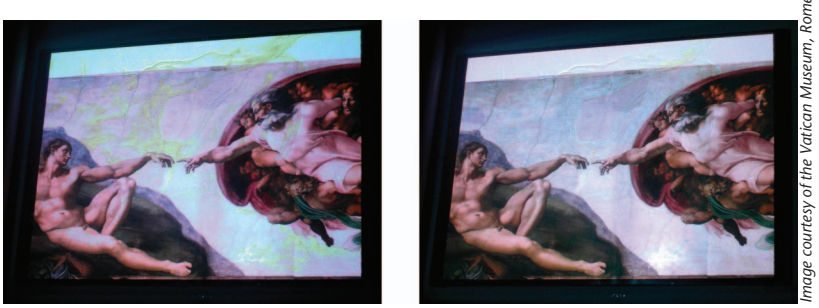


Image courtesy of the Vatican Museum, Rome

Figure 5. Results of color-correction process with two projectors. (a) Limited intensity capabilities of a single projector result in visible artifacts. (b) The contribution of a second projector reduces these effects.

Content creation, authoring, and presentation

The basic color-correction process allows visualizing all sorts of information and effects by rendering the desired result as input image I_i into the pixel shader. This opens the potential for applying a wide range of established or new presentation tools and techniques. We want to categorize them into six main classes:

- inlay objects (textual information, images, videos, and arrow annotations);
- lens effects (magnification, x-ray, toolglasses, and magic lenses[5]);
- focus effects (blurring, decolorization, hiding, and highlighting);
- 3D effects (walk- or flythrough and object observation);
- modification effects (drawing style modifica-

Image courtesy of the Vatican Museum, Rome

Figure 6. Sample scenes of the papal chapel's ceiling projected onto the Adam drawing (Figure 4a) during an interactive slide presentation. The underlying drawing is barely visible.

Image courtesy of the National Gallery, London

Figure 7. Pontormo's Joseph and Jacob in Egypt *(65 × 49 cm). (a) Copy of original painting illuminated under environment light, (b) modification of painting style from oil on wood to watercolor on paper via a 2D artistic filter, (c) re-illumination and lens flare, and (d) registered visualization of underdrawings (infrared recordings are black and white).*

tion, re-illumination, and color restoration[6]); and

- audio effects (verbal narrations, music, and sound that are synchronized to the visual effects).

Figure 9 illustrates a variety of examples. Embedded information windows (see Figure 9a), such as the semitransparent text panel or the opaque image and video panel are—in contrast to simple physical text labels—dynamic and can be directly linked to corresponding portions of the artwork (via arrow annotations, for example). The magnifying glass in Figure 9b allows zooming into interesting areas to identify brush strokes and hatching styles of the artist. Usually museum visitors are restricted from getting close enough to recognize such details. To draw the observers' attention to a certain image portion, it can be brought into the foreground while the remaining part is eclipsed. In Figure 9c, the focus area is highlighted while the rest is decolorized.

This is technically achieved by projecting the inverse colors of the background image onto *M*. In this case, the colors of the pigments on the canvas are physically canceled out and *M* appears in grayscale. Figure 9d shows a 3D flythrough in the Sistine chapel—building a spatial correlation to surrounding paintings and the environment. This allows relocating the observer virtually to remote places.

Stereoscopic rendering (and optional head-tracking technology) allows an immersive experience. Although the transparent film preserves the polarization of light, the underlying canvas doesn't. Consequently, we can only apply active or anaglyphic stereo solutions. We use a mechanical (half opaque and half transparent) shutter wheel rotating with 120 Hz in front of two video projectors. A light sensor measures the wheel's position and triggers the shutter glasses' infrared synchronization signal. In combination with conventional LCD projectors, this is a simple and—compared to CRT projectors—cost-efficient alternative for active stereo.

Figure 7 illustrates further presentation examples, such as modifications of the painting style via 2D artistic filters, visualization of the underdrawings, and scene modification through re-illumination.

For noninteractive presentations, it's possible to pregenerate the entire content. In this case, we can apply well-established content creation and authoring tools (such as digital imaging, 3D modeling, video editing, and audio-mixing packages). These tools already provide techniques such as animations, image filtering, rollover effects, and so on, as well as professional graphical user interfaces. For interactive presentations, however, the generated content must be man-

aged by the presentation software linked to an interaction framework.

If no user interaction is required, we apply an embedded Audio Video Interleaved (AVI) player to map video frames onto input images I_i on the fly. This represents a direct interface to our own player framework that comprises rendering and color correction, and allows the user or content creator to benefit from the capabilities of established content creation and authoring packages, such as Adobe Photoshop, Discreet 3ds max, Alias Maya, or Adobe Premiere. Our content interface for 3D interactive and stereoscopic/head-tracked presentations is Nvidia's NVB format, which can be exported from Discreet's 3ds max and displayed by our player.

To see how others have approached incorporating projectors, see the sidebar on the next page, "Previous and Related Work."

Image courtesy of the Museum het Rembrandthuis, Amsterdam

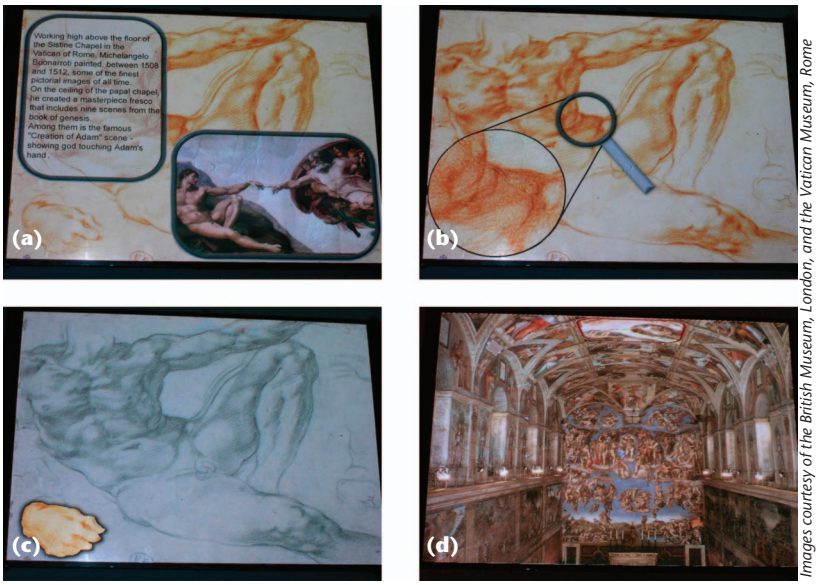
Figure 8. Rembrandt's self-portrait (48 × 56 cm). (a) Copy of original painting as it looks today (illuminated under environment light). Various cleaning stages to remove the overpainted layers from (b) 1935, (c) 1950, and (d) 1980 are projected onto (a). Only black-and-white photographs of these stages are available. The high black level of the video projectors prevents us from creating a totally black color on the canvas. Extreme regions, such as overlaid hair and a hat can't appear completely black for this reason.

Future work

Using pictorial artwork as information displays opens another door to embedded multimedia content in museums and art galleries. Our proposed method is simple, seamless, cost and space efficient, robust, and compatible with off-the-shelf hardware and software. These are all important factors for museum operators, content creators, and museum visitors. The presented techniques let us think of a wide variety of presentation and interaction tools that we didn't explore in this article. Dynamic view management and label placement are only two examples. As for other areas, the development of efficient interaction techniques and devices for such displays will be an interesting challenge.

As we previously discussed, our method has limitations that are mainly defined by the capabilities of the applied hardware. For example, the restricted resolution, brightness, contrast, minimum focus, distance, and black level of video projectors are issues that will certainly be improved by next-generation projectors. The XGA resolution and the brightness of 1,100 ANSI lumens of our low-cost projectors were appropriate for small- and medium-sized paintings in a normally lit environment. An upscaling is possible by using more projectors, but downscaling would either result in a loss of effective resolution or in focus problems.

Black, for instance, is a color that can't be projected. Instead the environment light together with the black level of the projectors illuminates areas that need to appear black. We found that



Images courtesy of the British Museum, London, and the Vatican Museum, Rome

Figure 9. Examples of presentation techniques: (a) inlay text and image, (b) magnification, (c) focus through highlighting and decolorization, and (d) 3D flythrough in the Sistine chapel. Note that in images (a–c) the Adam drawing itself isn't projected.

Previous and Related Work

With increasing capabilities and decreasing costs, video projectors have become widespread and established presentation tools. Being able to generate images that are larger than the actual display device virtually anywhere is an interesting feature for many applications that can't be provided by desktop screens. Several research groups have discovered this potential by applying projectors in unconventional ways to develop new and innovative information displays that go beyond simple screen presentations.

The Luminous Room,[1] for instance, describes an early concept for providing graphical display and interaction at each interior architecture space's surface. Co-located two-way optical transducers—called *I/O bulbs*—that consist of projector-camera pairs capture the user interactions and display the corresponding output. With the Everywhere Displays Projector,[2] Pinhanez has extended this concept technically by allowing a steerable projection using a pan/tilt mirror. Later, Raskar et al.[3] demonstrate how context-aware handheld projectors—so-called *iLamps*—can be used as mobile information displays and interaction devices.

Raskar et al. also use multiple projectors for their Shader Lamps[4] approach to lift the visual properties of neutral diffuse objects that serve as a projection screen. The computed radiance at a point of a nontrivial physical surface is mimicked by changing the bidirectional reflectance distribution function (BRDF) and illuminating the point appropriately with projector pixels. Animating the projected images lets us create the perception of motion without physical displacement of the real object.[5] This type of spatial augmentation is also possible for large, human-sized environments, as demonstrated by Low et al.[6]

Projector-based illumination has become an effective technique in augmented reality to achieve consistent occlusion[7,8] and illumination[9] effects between real artifacts and optically overlaid graphics. Video projectors instead of simple light bulbs are used to illuminate physical objects with arbitrary diffuse reflectance. The per-pixel illumination is controllable and can be synchronized with the rendering of the graphical overlays. It also makes the combination of high-quality optical holograms with interactive graphical elements possible.[10] Using a video projector to produce a controlled reference wave lets us reconstruct a hologram's object wave partially—but not at those portions that are overlaid by integrated graphical elements.

Yoshida et al.[11] describe a virtual retouching technique that applies video projectors for reconstructing and enhancing the faded colors of paintings by projecting light directly onto the canvas. An affine correlation between projection and the result captured by a camera is established manually for each pixel. Users can then retouch the original painting interactively via a desktop GUI.

In the context of our own approach, we can divide these methods into four main groups:

- information displays that project arbitrary images onto surfaces with arbitrary reflectance, but don't consider the effect of color blending;[1-3]
- techniques that project colored images onto surfaces with a neutral white reflectance to avoid color blending;[4-6]
- methods that project a uniformly colored illumination onto surfaces with arbitrary reflectance and texture; [7-10] and
- approaches that project colored images onto surfaces under consideration of their reflectance.[11]

As with Yoshida's work, our approach belongs to the fourth group. Although there are basic conceptual similarities between both attempts, they differ in their general technological realization, aimed applications, as well as in their mathematical model and rendering techniques. We want to know whether a direct projection of colored light onto surfaces with arbitrary reflectance will work effectively under real-world conditions. The limitations of such an approach will be reached quickly in cases where an adequately large portion of the color spectrum is absorbed by the underlying surface and can consequently not be reflected.

References

1. J. Underkoffler, B. Ullmer, and H. Ishii, "Emancipated Pixels: Real-World Graphics in the Luminous Room," *Proc. ACM Siggraph*, 1999, ACM Press, pp. 385-392.

even in our case the human vision system adjusts well to local contrast effects—which makes these areas appear much darker than they actually are. Even with little environment light, however, the high black level of video projectors causes this illusion to fail in extreme situations, such as the one shown in Figure 9. The development of video projectors indicates that we can expect a decrease of the black level and an increase of the contrast ratio in the future.

Light can damage the artwork. Ultraviolet (UV) and infrared (IR) radiation produced by the lamps of video projectors is critical. Commercially available UV/IR blocking filters can be mounted in front of the projectors' lenses to remove most of these unwanted rays while transmitting visible wavelengths.

For the remaining visible light portion, a general rule of thumb in the museum field advises to illuminate valuable and delicate pieces perma-

2. C. Pinhanez, "The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces," *Proc. Ubiquitous Computing*, Springer, 2001, pp. 315-331.
3. R. Raskar et al., "iLamps: Geometrically Aware and Self-Configuring Projectors," *Proc. ACM Siggraph*, ACM Press, 2003, pp. 809-818.
4. R. Raskar et al., "Shader Lamps: Animating Real Objects with Image-Based Illumination," *Proc. Eurographics Rendering Workshop,* Eurographics Assoc., 2001, pp. 89-102.
5. R. Raskar, R. Ziegler, and T. Willwacher, "Cartoon Dioramas in Motion," *Proc. Int'l Symp. Nonphotorealistic Animation and Rendering*, ACM Press, 2002, pp. 7-ff.
6. K. Low et al., "Life-Sized Projector-Based Dioramas," *Proc. Symp. Virtual Reality Software and Technology*, IEEE CS Press, 2001, pp. 93-101.
7. O. Bimber and B. Fröhlich, "Occlusion Shadows: Using Projected Light to Generate Realistic Occlusion Effects for View-Dependent Optical See-Through Displays," *Proc. IEEE/ACM Int'l Symp. Mixed and Augmented Reality*, IEEE CS Press, 2002, pp. 186-195.
8. S. Noda et al., "An Optical See-Through Mixed Reality Display with Real-Time Rangefinder and an Active Pattern Light Source," *Trans. Virtual Reality Soc. of Japan*, Virtual Reality Soc. of Japan, vol. 4, no. 4, 1999, pp. 665-670.
9. O. Bimber et al., "Consistent Illumination within Optical See-Through Augmented Environments," *Proc. IEEE/ACM Int'l Symp. Mixed and Augmented Reality*, IEEE CS Press, 2003, pp. 198-207.
10. O. Bimber, "Combining Optical Holograms with Interactive Computer Graphics," *Computer*, vol. 37, no. 1, Jan. 2004, pp. 85-91.
11. T. Yoshida, C. Horii, and K. Sato, "A Virtual Color Reconstruction System for Real Heritage with Light Projection," *Proc. Virtual Systems and Multimedia*, IEEE CS Press, 2003, pp. 158-164.

nently with no more than 100 lx–150 lx. The potential damage caused by light is cumulative (for example, 1 hour with 1,000 lx equals 1,000 hour with 1 lx). The damage also depends on the material and color of the painting and the wavelength (color) of the light. A temporary illumination of a higher light intensity isn't critical.

To give rough indications of the light levels that were reached in our experiments, we measured an ~300 lx maximum for the results shown in Figure 4 (single projector), an ~400 lx maximum for the results shown in Figure 6 (two projectors), and an ~700 lx maximum for the results shown in Figure 5 (also two projectors).

During a 2–3 minute presentation, this increased lighting is only temporary and such highlight situations usually only appear (if at all) for a short period of time (a few seconds) at varying locations on the canvas. Nevertheless, using the intensity adjustment we've described, the maximum light level can be constrained to be below an upper threshold. However, this might cause visible artifacts depending on the presented content, the painting, and the environment light (as described in Figure 2). Thus, it's important to reach a good balance between total illumination (projected and environment light) over time and creating convincing presentation effects.

We currently consider only the intensity of the environment light E. This is adequate for regular white light sources but will result in artifacts if visible color shading is created on the canvas by the environment illumination. Without modifying the mathematical model or rendering process, the environment light's color can be compensated by determining it with the aid of colorimeters, encoding this information in E, and passing it to the pixel shader.

Using cameras in combination with projected structured light samples might help us automate our registration, color, and intensity adjustment steps. This, however, requires that an acceptable level of precision can be achieved.

The presented concept and techniques are applicable in combination with diffuse pictorial artwork, such as watercolor paintings, pen or ink drawings, sanguine sketches, or matte oil paintings. Extreme light and view-dependent effects, such as non-Lambertian specular reflections; self-shadows; subsurface scattering and interreflections that are created by brush strokes, paint material; or canvas textures can't be handled yet. We'll investigate these ideas in our future research. However, some of these cases will require components—such as head tracking—that might not be effectively integrated into museum environments.

We plan to evaluate our approach in museums and art galleries. We'll have to conduct user studies and surveys on the efficiency and acceptability of this technology in the future. The feedback from visitors, curators, and staff members will lead to improvements and new ideas for the interactive presentation of pictorial artwork. **MM**

Acknowledgments

We thank the Museum het Rembrandthuis, Amsterdam, for providing digital photographs used for the experiment shown in Figure 8, and Matthias Hanzlik for the illustrations in Figures 1 and 2.

References

1. Y. Chen et al., "Automatic Alignment of High Resolution Multi-Projector Displays Using an Uncalibrated Camera," *Proc. IEEE Visualization*, IEEE CS Press, 2000, pp. 125-130.
2. R. Yang et al., "PixelFlex: A Reconfigurable Multi-Projector Display System," *Proc. IEEE Visualization*, IEEE CS Press, 2001, pp. 21-26.
3. R. Raskar et al., "Shader Lamps: Animating Real Objects with Image-Based Illumination," *Proc. Eurographics Rendering Workshop,* Eurographics Assoc., 2001, pp. 89-102.
4. A. Majumder et al., "Achieving Color Uniformity Across Multi-Projector Displays," *Proc. IEEE Visualization*, IEEE CS Press, 2000, pp. 117-124.
5. E. Bier et al., "Toolglasses and Magic Lenses: The See-Through Interface," *Proc. ACM Siggraph*, ACM Press, 1993, pp. 73-80.
6. T. Yoshida, C. Horii, and K. Sato, "A Virtual Color Reconstruction System for Real Heritage with Light Projection," *Proc. Virtual Systems and Multimedia*, IEEE CS Press, 2003, pp. 158-164.

Oliver Bimber is a junior professor for augmented reality at the Bauhaus University Weimar in Germany. His research interests include future display technologies, real-time rendering, optics, and advanced human–computer interfaces. He holds a PhD in computer science from the Technical University of Darmstadt.

Franz Coriand is a graduate student of media systems at the Bauhaus University Weimar. His research interests include augmented reality and virtual reality, as well as software engineering and Unix-based operating systems.

Alexander Kleppe is a graduate student of media systems at the Bauhaus University Weimar. His research interests include augmented reality and computer graphics, as well as input devices and interaction techniques for virtual environments.

Erich Bruns is a graduate student of media systems at the Bauhaus University Weimar. His research interests include computer graphics, augmented reality, and the psychology of perception, as well as the relation between anthropology and computer science.

Stefanie Zollmann is a graduate student of media systems at the Bauhaus University Weimar. Her research interests include virtual reality and augmented reality, as well as input and manipulation techniques for 3D environments.

Tobias Langlotz is a graduate student of media systems at the Bauhaus University Weimar. His research interests include augmented reality, virtual reality, and multiuser game engines.

Readers may contact Oliver Bimber at obimber@computer.org.

For further information on this or any other computing topic, please visit our Digital Library at http://www.computer.org/publications/dlib.

RFIG Lamps: Interacting with a Self-Describing World via Photosensing Wireless Tags and Projectors

Ramesh Raskar* Paul Beardsley Jeroen van Baar Yao Wang Paul Dietz Johnny Lee
Darren Leigh Thomas Willwacher

Mitsubishi Electric Research Labs, Cambridge MA†

Abstract

This paper describes how to instrument the physical world so that objects become self-describing, communicating their identity, geometry, and other information such as history or user annotation. The enabling technology is a wireless tag which acts as a radio frequency identity and geometry (RFIG) transponder. We show how addition of a photo-sensor to a wireless tag significantly extends its functionality to allow *geometric* operations - such as finding the 3D position of a tag, or detecting change in the shape of a tagged object. Tag data is presented to the user by direct projection using a handheld locale-aware mobile projector. We introduce a novel technique that we call interactive projection to allow a user to interact with projected information e.g. to navigate or update the projected information.

The ideas are demonstrated using objects with active radio frequency (RF) tags. But the work was motivated by the advent of unpowered passive-RFID, a technology that promises to have significant impact in real-world applications. We discuss how our current prototypes could evolve to passive-RFID in the future.

CR Categories: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Interaction styles; I.4.1 [Image Processing and Computer Vision]: Digitization and Image Capture—Imaging geometry;

Keywords: human-machine communication, augmented reality, projector, radio frequency identification, stucture from motion, image stabilization

1 Introduction

As computational devices shrink and their power requirements and cost decline, it becomes feasible to envisage an everyday world whose surfaces are embedded with computational power and locally-held data. This paper describes the tagging of physical objects with radio frequency (RF) tags. Our experimental work is based on active, battery-powered radio frequency tags. However, our goal has been to develop methods that can be used with passive, unpowered radio frequency identification (RFID) tags. It is the cost of custom-design that prevented us from actually doing experiments with passive-RFID. Passive tags are just beginning to enjoy widespread deployment in real-world applications and the ability to implement our techniques on passive tags is fundamental to the usefulness of the ideas.

We augment each tag with a photo-sensor to significantly extend the current functionality and support radio frequency identity and geometry (RFIG) discovery. The ability to address and wirelessly access distributed photosensors creates a unique opportunity. We recover geometric information, such as 3D location of tags or shape history of tagged objects, and exploit the associated geometric operations to bring the RF tags into the realm of computer vision and computer graphics.

We present a complete solution for retrieving, viewing, navigating, and updating tag data. We build on our work with iLAMPS in [Raskar et al. 2003] using handheld locale-aware mobile projectors (LAMPs) to do direct projection onto tagged objects. Further our work goes beyond passive projection to show a truly interactive system in which projected information can be navigated and updated.

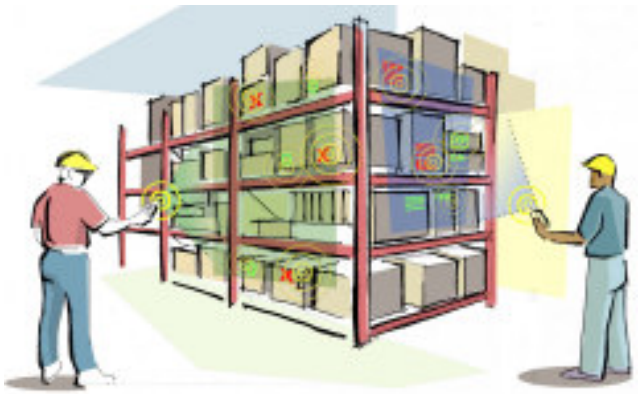


Figure 1: Warehouse scenario. A user directs a handheld projector at tagged inventory, with communication mediated by two channels – RF and photo-sensing on the tags. The user sees a projection of the retrieved tag information collocated with the physical objects, and performs a desktop-like interaction with the projection. A second user performs similar operations, without conflict in the interaction because the projector beams do not overlap.

1.1 Contributions

At the conceptual level, we show how to extend current tag-readers, which operate in broadcast mode with no concept of a 3D coordinate frame, to **allow selection of individual tags,** and create **a 3D coordinate frame** for the tags. This new functionality is achieved

*email:[raskar,pab,jeroen,dietz,leigh]@merl.com
†web:http://www.merl.com/projects/Projector/

by combining the tag-reader with a projector, and by augmenting each tag with a photo-sensor. In fact, we show that the system of projector and photo-sensing tag offers a new medium for many of the results from the area of structure-from-motion in computer vision, with projector and tags replacing camera and image interest points.

We also introduce **interactive projection,** allowing a user with a handheld projector to do mouse-style interaction with projected information. This is achieved by treating a projection as conceptually having two parts – a stabilized component that is static in the display surface, and a cursor that follows any user pointing motion of the projector – effectively allowing the user to track a cursor across a projection. Accompanying mouse buttons are used to do selection. Interactive projection is not specific to a tagged environment, but the technologies meld well.

1.2 A Self-Describing World

Our work in this paper involves experiments on the individual functions of active, photo-sensing RFID tags. However we motivate the work by suggesting a real-world application that could utilize the ideas. The first large-scale use of passive-RFID tags is expected to be for inventory control as part of logistics (a US$900 billion industry), so we turn to a scenario in a warehouse - locating objects with a required property and annotating them. For example, a warehouse employee identifies food products that are close to expiry date and annotates an instruction to trash them. If these products were items on a computer desktop, this could be done with a few clicks. Our goal is to craft a scheme in the physical world that maintains the simplicity of the computer environment. In the computer, the items are files, and the interface is via keyboard, mouse, and display. In the physical world, the items are tagged objects, and the interface uses a handheld projector and user interaction directly through the projected information. Figure 1 illustrates a handheld system being used in the warehouse scenario - selecting tags, collecting data, projecting imagery onto the objects to present the information to the user, and interacting with the projection. The term 'handheld' is used in this paper to cover both continuous handheld motion of the projector, and motion interspersed with static but opportunistic placement, as might occur during say augmented reality task-guidance.

We describe a number of innovations at the technical level. We show how to address the two hard requirements of augmented reality – object recognition, and recovery of 3D pose – without employing visually obtrusive markers. We extend passive display of augmented reality with an easy way to navigate through and update augmentation information. We show geometry-based functionality like finding the geometric configuration or geometric deformation of a set of tags. We show image-based functionality such as capturing imagery of the surface texture of a tagged object, with correction of perspective effects - this image data provides a suitable basis for creating the desired augmentation that is to be projected onto the surface. In the specific area of projection, we show a way to deal with the problem of projecting augmentation onto dark or cluttered surfaces, by capturing an image of the surface, then projecting the distortion-free image plus its augmentation onto a clean display surface.

The primary reason for using a projector in our work is that it underpins the technique for selecting tags and for setting up a 3D coordinate frame. This technique involves light communication that is not for human viewing. But it is convenient to make dual-use of the projector - using it also to project augmented reality data onto tagged surfaces, for human viewing. Projection has some advantages over other augmented reality approaches - augmented reality on a handheld display requires a context-switch as the user visually matches what is on the display with the physical object. And eye- or head-mounted displays need a real-time, accurate computation of the relationship between the user's coordinate frame and the world.

The limitations of our system are that a projector must have line-of-sight when interacting with a photo-sensing tag. The viewing quality of projected augmentation is affected by ambient lighting, low albedo surfaces, and shadows. Range for tag communication with our current setup is 2-3m. Within these constraints, the work still suggests many possibilities.

For applications, we focus on augmented reality, not just passive display, but demonstrating user interaction with the augmentation. Augmented reality faces challenges in doing reliable object recognition and computation of object position. Current solutions that use visual markers or user trackers are unwieldy. In contrast, a tag-based system can provide object identification and (as we show) object position in an unobtrusive way. Tags also allow direct storage of augmentation data with tagged objects and this sometimes has advantages over keeping an object's data in a remote database - say for mislaid objects, or for transient annotations.

1.3 Relationship of our Work to Passive RFID

This section describes why we assign importance to being able to support passive RFID tags, and what would be involved in evolving from the current work to a passive RFID-based system.

Powered radio-frequency tags currently use a battery that is about the size of a watch-battery, have a lifetime of a few years, and have a cost of a few dollars. In contrast, passive RFID tags are unpowered, can be as small as a grain of rice, and cost tens of cents [Want 2003]. Prices of both are dropping but the price differential will remain. The size and cost properties are such that RFID is showing signs of being adopted as a mass-deployment technology. Current commercial applications including embedding of RFID tags in packaging for inventory control, non-contact access control, and ear tags for livestock. Despite the interest in RFID, the available functionality is very limited – an RF-reader broadcasts a request, and in-range tags (collect energy from the RF pulse and) reply. Our work is motivated by the observation that RFID is showing the potential to be a ground-breaking, pervasive technology, yet current functionality is limited.

The key issue in evolving our active tag system to passive tags would be power. In the work in this paper, we only allowed ourselves computation and sensing consistent with the size and power levels we felt were achievable on a passive RFID system. For example, (a) tags are not photo-sensing or computing until woken up by the RF reader and (b) we do not have a light emitting diode (LED) on the tag as a visual beacon to a human or camera-based system because it would be power-hungry. Also note that our tags incorporate a photo-sensor, so a passive version could draw power not just from the RF channel, but also from the incident light. Of course, there would be significant engineering challenges in moving from active to passive RFID. Still, this has been the motivating vision for the work, and this has determined our design choices for the current prototype.

1.4 Related Work

Smart Objects: Multiple groups have looked at adding intelligence to objects and, in some cases, building human interactions around them. Smart-Its provides interconnected smart devices that can be attached to everyday items [Holmquist et al. 2001]. Visual feedback is not considered. SenseTable tracks and augments the area around sensing tablets on a tabletop using a projector [Patten et al. 2001]. Intelligent furniture has also been explored [Omojola et al. 2000]. Some systems use active RF tags that respond to laser pointers. The FindIT flashlight uses a one-way interaction and an indicator light on the tag to signal that the desired object has been found

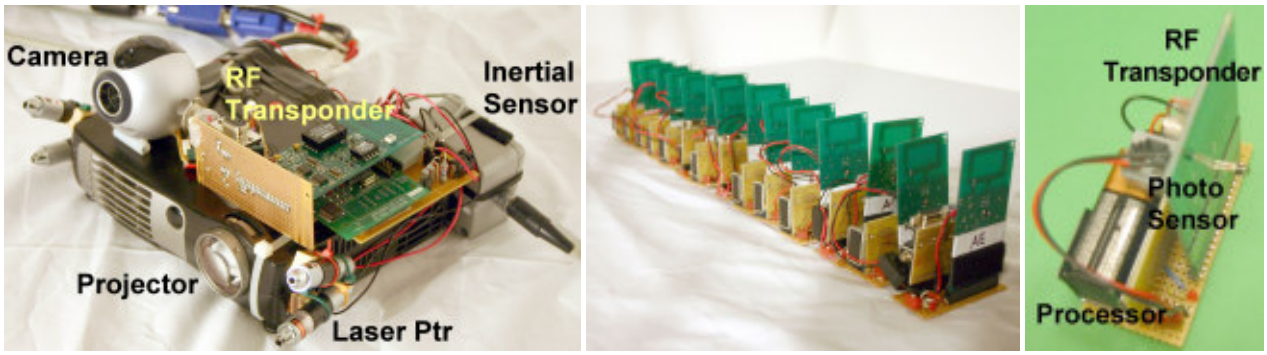


Figure 2: (a) Handheld projector system, (b) Fourteen of the tags and (c) Details of a tag.

[Ma and Paradiso 2002]. Other systems use a two-way interaction, where the tag responds back to the PDA using a power-hungry protocol like 802.11 or X10 [Patel and Abowd 2003] and [Ringwald 2002]. CoolTown [The CoolTown Project 2001] uses beacons that actively transmit devices references but without the ability to point and without visual feedback.

Interaction and Augmentation: Interaction with laser pointers for large display screens is common. A collection of sensing and interaction techniques for mobile devices is described in [Hinckley et al. 2000]. Augmentation of physical world objects has been primarily achieved via eye-worn or head-mounted displays [Azuma et al. 2001] or handheld screens. Screen-based augmentation using PDA, camera, and RFID tags is described in [Want et al. 1999; Rekimoto et al. 2001]. Projector-based augmentation has been explored by [Raskar et al. 1998; Underkoffler et al. 1999; Pinhanez 2001; Raskar et al. 2001; Bimber et al. 2001; Verlinden et al. 2003]. Image warping for a handheld projector to support shape and object adaptive projection is described in [Raskar et al. 2003]. The Cricket project [Teller et al. 2003] recovers pose of a handheld device using installed RF- and ultrasound- beacons, and does projected augmentation. A handheld projector as virtual flashlight, creating real shadows of virtual objects on real surfaces and supported by camera and optical tracking of fiducials on the wall, is described in [Foxlin and Naimark 2002]. Adding distinctive visual markers for tracking is not always practical, and unlike previous real-time AR applications, we do not add obtrusive environmental infrastructure to obtain a 3D context.

Other: Location sensing systems such as the Olivetti Active Badge [Want et al. 1992] and Xerox PARCtab [Want et al. 1995] recover location, but have typically been used for passive tracking, not an interactive system. Our work is also influenced by efforts in wireless sensor networks, pervasive computing and amorphous computing [Abelson et al. 2000], systems that use autonomous units to propagate information. Our passive tags lack the power to communicate with neighbors, but we use the handheld projector to provide an information conduit between tags.

A distinguishing factor between our work and related systems is in the parallel nature of the interaction. Laser-pointer systems require a user to identify a target object and direct the pointer at it to initiate interaction. But accurate pointing is difficult when the tags become visually imperceptible. And multiple tags can only be dealt with serially. In contrast, we use a casually-directed projector to interact with all tagged objects within the projector beam, and can select all objects or a subset for further interaction. Further, our tags are rich data sources, with the interactive projection technique providing a flexible user interface to the data.

1.5 Paper Organization

Section 2 describes the hardware. Section 3 describes tag selection and creation of a 3D coordinate frame. Section 4 describes interactive projection. Section 5 puts the pieces together in several examples of use. Section 6 describes results. Important issues that are not discussed because they are outside the scope of the paper are hardware design (miniaturization, cost, industry standards), RF communication (power, range, interference) and social aspects (privacy, security).

2 Physical Components

2.1 Handheld Projector System

The components of the prototype handheld projector system in Figure 2a are:

- a Plus V-1080 projector with 1024x768 resolution, 60Hz frame-rate, dimensions $7\times5\times1.5$ inches and weight 1kg,
- a rigidly attached 640x480 pixel webcam,
- a rigidly attached AC&C VS123 6DOF inertial sensor (accelerometer and gyroscope),
- four rigidly attached laser pens,
- a Parallax board for RF communication at 433.92 MHz with data transfer rate of 2400 baud,
- two click buttons for simple IO,
- a computer.

The camera is used to compute pose (position and orientation) of the device, as well as supporting user interactions that capture surface texture. The inertial sensor supports the computation of pose by providing high-frequency estimates (160Hz) of change in pose. The four laser pens, which could be invisible IR, are used to project distinctive points on the projection surface to support computation of pose, rather than constraining the form of the projector's own image data for this purpose. Euclidean calibration between all elements is done offline, and in the remaining discussion we assume projector and camera intrinsics are known and have been factored out. Much of the calibration uses established camera calibration techniques (modified for a projector) [Hartley and Zisserman 2000].

2.2 Tags

The prototype active tag, Figure 2b, has a microprocessor, memory, Parallax board for RF communication, and a Panasonic PN121PS-ND photo-sensor for sensing projector illumination.

2.3 Projector-Tag Communication

The handheld device has two ways to communicate with a tag. RF communication is omni-directional with a typical range in meters. Projected light provides directional communication via the photo-sensor. Figure 3 is a schematic of the devices and interaction.

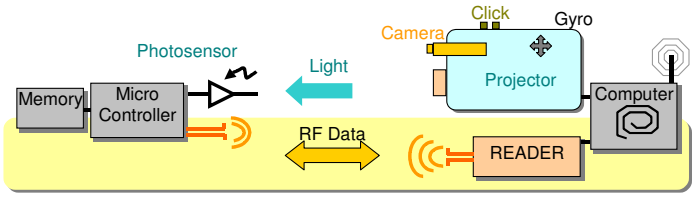


Figure 3: Communication between the photosensing tag (left) and handheld projector system (right).

Radio-frequency and photo-sensing are complementary approaches for communication. Traditional tags do not use visible or infrared light communication because occlusion and ambient illumination make light unsuitable for data transfer. However, light is suitable for directional methods such as directional selection of tags, and for recovering geometry by employing light rays.

3 Processing Tags

We describe novel functionality that becomes available when a tag-reader is combined with a projector, and a tag is augmented with a photo-sensor. Section 3.1 describes how this allows a user to select individual tags for interaction instead of working in broadcast mode, and Section 3.2 describes how to recover a 3D coordinate frame for the tag system.

3.1 Selecting Tags

Conventional tag communication works by broadcast from an RF-reader, with response from all in-range tags. Limiting the communication to a required tag is traditionally achieved using a short-range tag-reader and close physical placement with the tag. In contrast, we can select tags for interaction at long-range using projected light, while ignoring unwanted in-range tags. The handheld device in Section 2 first transmits an RF broadcast. Each in-range tag is awoken by the signal, and its photo-sensor takes a reading of ambient light, to be used as a zero for subsequent illumination measurements. The projector illumination is turned on. Each tag that detects an increase in illumination sends a response to indicate that it is in the beam of the projector, and is ready for interaction. The user can control the selection by varying the projector beam size, or by shaped projection. A more powerful mechanism for tag selection is given in Section 4.2 using interactive projection.

Figure 4: (a) A tagged surface, (b) One frame during Gray-code projection, (c) Correctly-placed projected augmentation.

3.2 Setting up 3D Coordinate Frame for Tags

Existing tag systems have no functionality for creating a 3D coordinate frame. There are directional RF devices that could be used to provide direction between RF-reader and tag, but they are not appropriate for handheld use, requiring meter-wide antennas, or multiple RF readers distributed in space. While computer vision could potentially be used, it would require visually-distinctive tags, and our goal is to move in the direction of small, indiscernible tags. Instead we propose the following approach.

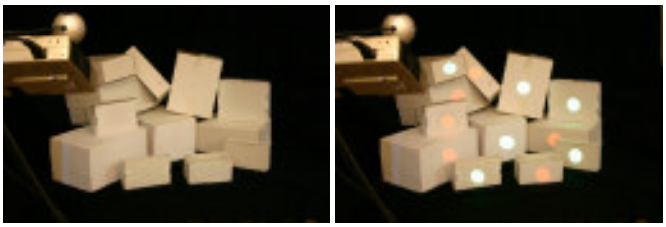


Figure 5: The projector beam reveals the presence of tagged objects

The handheld device is first aimed casually in the direction of a tagged surface like that in Figure 4a. The handheld device sends an RF signal to synchronize the tags, followed by illumination with a sequence of binary patterns – see one example frame in Figure 4b - such that each projector pixel emits a unique temporal Gray-code. The tag records the Gray-code that is incident on its photo-sensor, and then makes an RF transmission of its identity plus the recorded Gray-code back to the RF-reader.

The handheld device's camera also observes the projected light and determines where each Gray-code appears in the camera's view of the surface. Thus, via the Gray-code, we know the following information for the tag – what projector pixel **p** is illuminating it, and what camera pixel **c** is viewing it. Since this is a calibrated projector-camera system, this reduces to a triangulation problem in which we intersect the 3D rays associated with **p** and **c** to find the 3D position of the tag. It is then straightforward to create correctly positioned augmentation on the tagged surface as shown in Figure 4c and Figure 5.

4 Handheld Projector for Interaction and Display

Our other main contribution is to provide a novel user interface that allows a user to interact with a projection. Section 4.1 describes how we attain a stable projection from a handheld projector under hand-jitter, and Section 4.2 describes our method of interactive projection. Although the focus of this paper is on projection in tagged environments, note that interactive projection can work with any display surface, and does not require a tagged surface.

4.1 Stabilized Projection

In creating a handheld projector, an immediate problem that arises is that hand-jitter results in jitter in the projection. The core requirement to deal with this problem is to compute the pose of the projector relative to the display surface. There are multiple ways to approach this. Section 3.2 already introduced a way to find the 3D coordinates of tags on a tagged surface (and hence to find projector pose relative to the surface). However, the computation of projector pose via projected Gray-codes is too slow for correction of hand-jitter.

We have used two approaches to stabilize a projection on a planar display surface:

• The first approach, which we call absolute stabilization, works by recovering full projector pose - location and orientation - relative to the display surface, by utilizing the camera's view of four or more points of known coordinates in general position on the surface (or the equivalent, such as four known lines). Hence, we keep the projection static on the display surface, factoring out motion of the projector completely,

• The second approach, which we call quasi-stabilization, works by recovering projector pose relative to the display surface up to an unknown translation in the plane. It utilizes the camera's view of four projected points on the display surface e.g. the four vertices of the projector image (or the equivalent, such as the four boundary lines of the projector image). This gives projector pose to an unknown translation in the plane, and an unknown rotation around the plane normal. We fix the rotation around the plane normal by using a tilt-sensor to determine the world vertical, and hence to define a unique upright direction for any non-horizontal display surface. This allows us to preserve the projection size and orientation, but the projection translates in the display plane if there is a dominant projector motion parallel to the plane.

Laser Pens – The descriptions below involve the handheld device's camera detecting the four vertices of the current projector image. In fact, we detect the projected laser points from the four laser pens. This improves reliability of detection, because laser points are bright, and avoids a constraint on the form of the projector image. This involves an extra layer of processing because the laser rays are not concurrent with the projector pinhole [Beardsley et al. 2004b], but conceptually the situation is unaltered, so we omit this detail from the algorithms below.

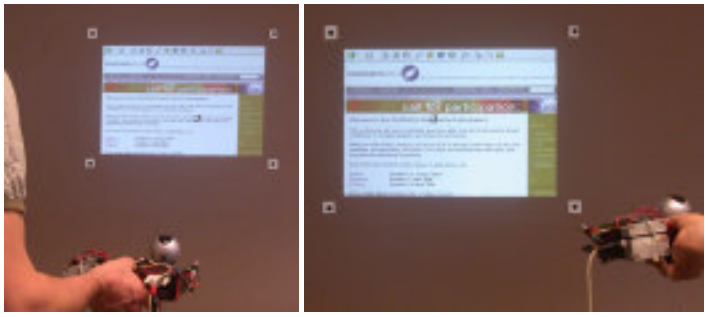


Figure 6: Absolute stabilization projects to a fixed location, regardless of projector position.

Absolute-Stabilized Projection Absolute stabilization, Figure 6, is possible when the camera is viewing four or more fixed surface points in general position, or the equivalent, on a planar display surface. Our goal is to find a homography H_{PS} between the projector image plane and a fixed coordinate frame on the surface. Homography H_{PS} specifies the mapping between each pixel on the projector image plane and the fixed surface coordinate frame. Hence we can use its inverse H_{SP} to transform from a desired (fixed) projection on the surface to the projector image plane, thereby determining the required projector image to give the fixed projection. The steps to find H_{PS} are as follows.

- Take an image with the handheld device's camera. Detect camera image points $\mathbf{x}_i$ for four surface features of known surface coordinates $\mathbf{X}_i$. Compute the homography H_{CS} between the camera image plane and the surface coordinate frame, using the correspondences $(\mathbf{x}_i, \mathbf{X}_i)$.
- In the same camera image, identify the four camera image points $\mathbf{y}_i$, corresponding to the vertices of the projector image. Knowing the projector pixel coordinates for the four vertices $\mathbf{v}_i$, compute the homography H_{CP} induced by the surface between the camera image plane and the projector image plane using the correspondences $(\mathbf{y}_i, \mathbf{v}_i)$.
- Compute the required homography between the projector image plane and the coordinate frame on the display surface $H_{PS} = H_{CS} H_{CP}^{-1}$.

In conjunction with this, we use the inertial sensor on the handheld device to provide a fast update of change in pose of the handheld device. To preserve the projective nature of the algorithm above, the pose change is used to predict a new estimate of the camera image points $\mathbf{x}_i$, and the algorithm then proceeds exactly as above.

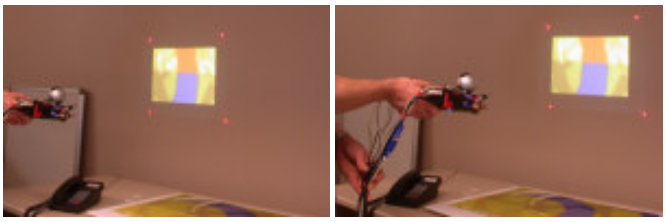


Figure 7: Quasi-stabilized projection translates with the user but maintains an upright rectangle with correct aspect ratio.

Quasi-Stabilized Projection Quasi-stabilization, Figure 7, preserves the form of the projection up to an unknown translation in the plane i.e. it preserves projection shape, size and orientation. But the projection translates in the display plane in accordance with projector motion parallel to the plane. The steps to achieve this are:

- Take an image with the handheld device's camera. Detect the camera image points $\mathbf{x}_i$ for the four vertices of the projector image boundary.
- Having a calibrated projector-camera system, knowing $\mathbf{x}_i$, and knowing the projector pixel coordinates for the four vertices $\mathbf{v}_i$, reconstruct the 3D plane of the display surface in the coordinate frame of the hadheld device. This provides projector pose relative to the display surface up to an unknown translation in the plane, and unknown rotation around the plane normal.
- Use the inertial sensor to get the world vertical, to fix the unknown rotation around the plane normal (for non-horizontal planes).
- Given projector pose up to an unknown translation in the plane, generate a projection of the desired shape, size and orientation on the plane.

The inertial sensor is incorporated into the scheme in a similar way to absolute-stabilization.

4.2 Interactive Projection

The term 'interactive projection' is used to refer to mouse-style interaction with projected data. Functionality to do pointing on a projection could be achieved with a touchpad on the handheld device, or with a separate laser pointer, but these are inconsistent with a compact device and one-handed interaction. We describe how to use simple one-handed direction of the projector to track a cursor across a stabilized projection. The handheld device has two click buttons for selection when the cursor is at a desired point. Interactive projection employs absolute stabilization described in Section 4.1, and therefore requires the display surface to have some texture.

To introduce the mechanism, first note that conceptually we have two types of projection. A conventional mode of projection might

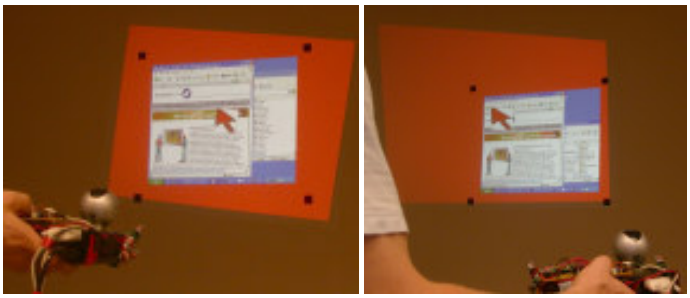


Figure 8: During interactive projection, a moving cursor can be tracked across a projection that is static on the display surface, to do mouse-like interaction with projected data. In these images, the background pixels are intentionally rendered in red to show the extent of projection. The pointer remains at the center of the projector field while displayed desktop remains stable within the surface features.

have a static image on the projector image plane and, as the projector is moved, the projection on the display surface moves in direct correspondence. In contrast, absolute stabilization, introduced in the previous section, has a dynamically-updated image on the projector image plane that results in a static projection on the display surface during projector motion.

Now note that both types of projection can be done simultaneously. Say we apply absolute stabilization to project some content such as a windows desktop environment. The user perceives a stabilized static windows desktop on the display surface. But in addition we always set a graphic for a cursor at the center of the projector image plane. Now as the projector moves, the user sees the stabilized projection of the windows desktop with the cursor tracking across it – see Figure 8. Furthermore the cursor motion follows the motion of the handheld projector, so it is a natural interaction for the user.

With the ability to point, projection becomes interactive. In fact all the standard mouse interactions from a WIMP interface are possible in this new setting. One difference is that in the projector domain, the cursor can be used to indicate either a selection in the projected content, or a selection of something in the physical world (with augmentation giving stabilized feedback about the selection e.g. a rectangular window on a surface). This opens up new types of interaction possibilities.

Interaction with projected content. Section 1 introduced a scenario for user interaction with warehouse inventory. This section describes how interactive projection supports the interaction. **(i) Option selection.** The user wishes to invoke a standard query like 'show inventory close to expiry date'. This can be done by having a menu or an action-button in the stabilized projection, and selecting with the cursor. **(ii) Tag selection.** Section 3.2 described how tags in the projector beam can be highlighted by projected augmentation to indicate their location to the user. To support tag selection, the augmentation could show a selection icon next to each tag - the cursor is tracked to the selection icon, with click to toggle its state. Alternatively the cursor can be used to select a rectangular region of interest by click-and-drag, with a stable view of the selected rectangle providing visual feedback about the selection, to select multiple tags simultaneously. **(iii) Controlling placement of augmentation.** The user can use the cursor to grab individual items in the projected augmentation and move them to a preferred layout.

Interaction with physical texture. The cursor can be used to indicate a point or a region selection for physical features in the world, to be captured by the camera and processed. We present a simple example - copy-paste of a selected part of a surface – in the next section. See further examples in [Beardsley et al. 2004a].

5 Functionality in a Tagged Environment

Section 3 and 4 proposed fundamental techniques that we build on in this Section. Here our contribution is to show practical functions for doing augmented reality and geometric processing in a tagged environment including (a) initializing tags by recovering and storing their 3D position on an object, (b) recovering images of a tagged object's surface texture to provide a basis for creating an augmentation overlay, (c) using copy-paste as a way to show a projected augmentation for surfaces that don't support direct projection.

5.1 Tag Geometry – Geometric Configuration

This section describes how to compute the 3D coordinates of a set of tags on an object. As motivation, assume a newly manufactured object with tags that have been embedded in approximately known positions (say the vertices of the object), but more accurate 3D coordinates are required for subsequent augmentation. Section 3 described how to compute the 3D position of an individual tag relative to the handheld device. But error in individual calculations makes this approach unsuitable to compute the positions of multiple tags in a common frame. We instead use an approach based on structure-from-motion [Hartley and Zisserman 2000].

- For a given placement of the handheld device relative to the object, compute the projector pixels $\mathbf{x}_i$, $i = 1..m$ that illuminate each of the m tags, $m >= 8$, using the method of Gray-code projection in Section 2.1.
- Repeat for N distinct placements of the handheld device. This reduces to the following computer vision problem – compute structure from Nviews of m matched features.
- Compute the essential matrix E between first two views. Initialize a Euclidean coordinate frame and the 3D positions of the tags.
- Process each subsequent view, using a Kalman Filter to update the 3D tag positions.
- Do bundle adjustment to refine the final result.
- Transform the coordinate frame to a world vertically aligned system, using the tilt of the inertial sensor.

Delaunay triangulation is used to create a 3D mesh of tag vertices and hence surface facets. All the computed geometry and connectivity is stored in the tags. A projector subsequently recovers its pose in the object coordinate frame as follows.

- Illuminate the object and recover the projector pixel $\mathbf{x}_i$ illuminating each tag by the method of projected Gray-codes.
- Retrieve the 3D coordinates $\mathbf{X}_i$ stored in each tag.
- This again reduces to a standard computer vision problem (requiring no modification for a projector, which is also a pinhole device) - compute camera pose given 2D-3D correspondences $(\mathbf{x}_i, \mathbf{X}_i)$. Given the projector internal parameters, use four correspondences to compute the projector pose [Zhang 1999].

5.2 Tag Geometry - Deformation

This section describes detection of non-rigid deformation in a set of tags as a way to detect change in the underlying object or surface. This can be useful to detect and highlight change, say detecting pressure damage during shipping, or monitoring deformations due to moisture or heat.

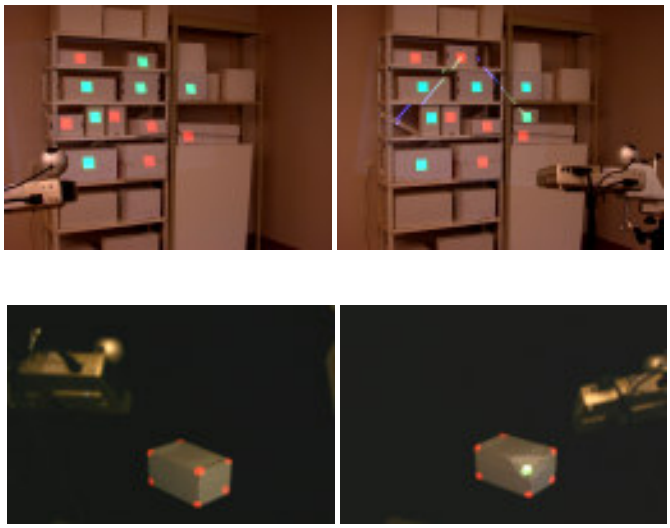


Figure 9: (a) Tagged boxes in original position, (b) Projection indicates change in box locations, (c) Box before deformation, (d) Projection indicates deformation of front vertex.

Each tag contains stored 3D coordinates $\mathbf{X}_i$ acquired pre-deformation, as in Section 5.1. The RANSAC-type procedure [Fischler and Bolles 1981] to detect deformation assumes that the majority of the tags are rigid, and a small number of tags have changed their relative position – we identify the tags that conform to rigidity, in order to obtain the tags that have undergone non-rigidity. The steps are:

- Find the tag positions $\mathbf{x}_i$ on the projector image plane using projected Gray-codes, and retrieve the 3D coordinates $\mathbf{X}_i$ by RF request, to obtain the correspondences $(\mathbf{x}_i, \mathbf{X}_i)$.
- Take a subsample of four correspondences from the full set $(\mathbf{x}_i, \mathbf{X}_i)$.
- Estimate projector pose Γ' from the subsample. Use the computed pose to project all the undistorted 3D tag coordinates $\mathbf{X}_i$ onto the projector image plane. Use the reprojection error between the actual tag positions $\mathbf{x}_i$ and the ideal projections to determine the inlying correspondences to this solution – those having reprojection error $|\mathbf{x}_i - \Gamma'\mathbf{X}_i|$ less than threshold d.
- Continue for n random subsamples, retaining the solution Γ with the largest number of inliers. Record the outliers to this solution as deformed tags.

5.3 Tag Imagery – Local Surface Texture

This section describes how to recover distortion-free image texture for the surface facets of an object (obtained in Section 5.1). As motivation, we might store image texture to provide input for an editor, for creating and positioning the augmentation graphics on a tagged surface. As a second motivation, assume the image texture is stored for subsequent use for change detection on a tagged surface like a control panel.

The steps are:

- Use the method in Section 5.1 to recover the projector pose and hence, by the known calibration, the camera pose relative to the object.
- Using the surface facet information retrieved from the tags, determine the projection of each surface facet on the camera image plane.

Figure 10: Projected augmentation on the box is attached to tags at the vertices, and so adapts automatically to the opening of the lid.

- Extract the image texture for each surface facet. Use a homography between the camera image coordinates of the surface facet, and coordinates for a fronto-parallel projection of the surface facet (obtainable from the known 3D coordinates) to rectify the texture to a fronto-parallel view.
- Store the distortion-free surface facet texture on the tags.

The motivating functions suggested above are now straightforward e.g. given the projector pose in the object coordinate frame by the method in Section 5.1, and the captured image texture with updates to incorporate desired augmentation graphics, it is straightforward to project the image texture back to the correct location on the object as shown in Figure 10.

5.4 Interactive Projection - Copy-Paste

This section describes copy-paste of image texture from a physical surface. We use the handheld device's attached camera to capture texture from a tagged surface, normalize to remove viewpoint dependence, and paste this texture onto a new surface without distortion. As a motivating example, assume we want to project augmentation onto a surface like that in Figure 11, which does not readily support projection. In fact, our experience suggests that even cluttered, dark surfaces can support simple bold projection. But fine detail projection requires another solution. The copy-paste operation is used to capture an image of the surface in Figure 11, simultaneously recording data from the tags embedded in the surface. A distortion-free view of the surface is then 'pasted' to a projection on a clean surface, showing both the image texture and the overlaid augmentation

There are multiple ways to select a physical region for copy, including the method in Section 4.2 for selecting a rectangular ROI, but we outline a simple method here. Use the handheld device to project a full-white projector image onto a planar surface. Due to perspective the rectangular projector image will project as a quadrilateral. Define the copy area as the largest vertically aligned inscribed rectangle inside the projected quadrilateral. The steps are:

- Find the four vertices of the projected quadrilateral in the camera view. Knowing the projector pixels for the vertices, and the camera pixels, and having a calibrated projector-camera pair, compute the quadrilateral in 3D.
- Find the largest inscribed rectangle, S, inside the projection quadrilateral, aligned with the world vertical (obtained from inertial sensor).
- Compute the projection of S to the camera image plane. Store the image texture for S, incorporating a warp to generate a fronto-parallel view of the captured texture.

For pasting at a new location:

- Find an inscribed rectangle on the display surface as above.
- Project the stored texture centered on this rectangle

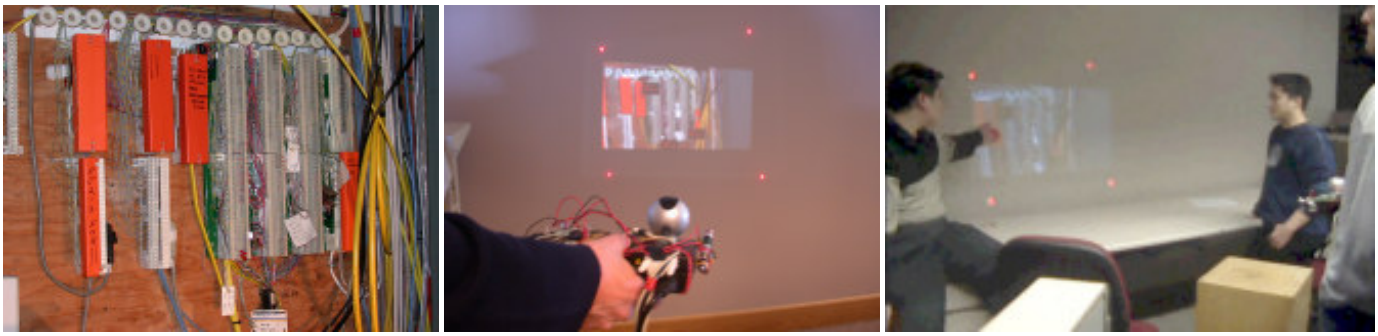


Figure 11: (Left) This surface, with embedded tags, does not readily support projected augmentation. To deal with this, during 'copy', an image is captured, and location and data on the tags is retrieved. (Middle and Right) During 'paste', perspective and photometric distortion is removed, and the image plus augmentation data is displayed on a clean projection surface.

Photometric Compensation Photometric compensation is used to produce a paste image that is as close as possible to the copied texture. There are two relevant factors – attenuation due to distance and orientation, and surface reflectance.

Projector illumination at a specific surface point is a function of projector-surface distance along the ray and incidence angle of the projector ray. The dimmest projector illumination falls on the surface point with the largest incidence angle. This therefore sets the brightness limit for the projection, and we apply a per-pixel attenuation map to the rest of the image to provide an even illumination. For a projector pose $\Gamma = [r_1\ r_2\ r_3\ t]$, with rotation $R = [r_1\ r_2\ r_3]$, the homography between the projector image plane and physical surface is $H_P = [r_1\ r_2\ t]$. A projector pixel x maps to 3D point $X = (X_x, X_y, 0)$ where $(X_x, X_y, 1) \cong H_P^{-1} x$. The angle between the projector ray and surface normal at X is

$$\theta = \cos^{-1}(V \bullet (0,0,1))$$

where, the projector ray is $V = (-R't - X)/|(-R't - X)|$

If the largest incidence angle is θ_m , then attenuation at every projector pixel x is $\cos(\theta_m)/V_z$, where V_z is the z components of the vector V given above. We achieve this attenuation with a simple pixel shader program.

For reflectance compensation, we avoid explicit albedo calculation [Nayar et al. 2003] by noting that the copied texture and the view of the illuminated surface after paste operation are both captured by the same camera. We take an image of the paste surface for two different illuminations from the projector. Given camera intensities I_1 and I_2, for a given camera pixel under the two illuminations L_1 and L_2, and a target paste intensity of I_T, the required linearly interpolated projector illumination is $L = L_1 + k(L_2 - L_1)$ where $k = (I_T - I_1)/(I_2 - I_1)$. For color images, we apply the correction independently in each channel.

6 Results

6.1 Qualitative and Quantitative Results

How well does our implementation work? Typical range between the handheld device and tags in our experiments was about 2m. The method for identifying which tags are within the projector beam, in Section 3.1, works reliably all the time. It would likely have problems in environments where the ambient light undergoes sudden change. The method for computing 3D tag position using Gray-code projection, in Section 3.2, takes about 2 seconds to do the projection. It works about 95% of the time, with the failures being due to noise in the Gray-code detection at the tag.

The method of stabilized projection in Section 4.1 results in a projection that is visibly more stable than raw projection under hand-jitter. However jitter is still present. We have experimented with interactive projection, in Section 4.2, by projecting a standard windows desktop environment onto a wall, invoking a browser, and doing operations like menu selection, clicking on links, and scrolling. The interactions feel somewhat clumsy due to the remaining projection-jitter - a major reason is that we are putting the computed cursor motion directly into a standard mouse-handler, rather than a custom-handler, and mouse-click is often interpreted as mouse-drag due to jitter. Despite this, the standard mouse-type interactions can be successfully carried out on the projection of a desktop environment with the current prototype.

The four functions in Section 5 can all be done reliably.

Projection stabilization in the presence of hand-jitter works by counteractive displacement of the image on the projector image plane, and hence we need a buffer zone around the usable projection area. The pixels outside the usable area are black. This decreases the effective projector field of view. In our case, we used the central one third of the pixels in both dimensions, i.e. about 15 degree horizontal and vertical field of view.

6.2 Inventory Control

The warehouse scenario for inventory management was introduced in Section 1. Here we consider a specific interaction, with results illustrated in Figure 12. A warehouse manager uses the handheld device to annotate food products, and another employee subsequently retrieves the annotation. The manager first uses an RF broadcast to wake up the tags, then uses the handheld device to select and transmit the RF query *find-products* that will expire within two days. Selection is done by projected menu or a projected action-button as described in Section 4.2. A tag that detects it is in the projector beam as in Section 3.1, checks its expiry date and sends an RF transmission of its expiry status back to the handheld device. The projector system determines the 3D location of all illuminated tags as in Section 3.2, and beams green circles on those objects (tags) that are not about to expire, and red circles on others. The manager selects a subset of red-circled objects, by region selection as in Section 4.2, and the circle illumination changes to white. The red and green circles on the other tags remain stabilized on the physical surfaces throughout, Section 4.1. The manager selects and transmits the RF request *mark-selected-objects* for removal, and the selected tags record the request internally. A subsequent user, from a different projector view, selects and transmits the RF query *show-items-requiring-action*. The projector illuminates all annotated items with colored circles, and projected augmentation indicates the actual annotated instructions.

Figure 12: Warehouse scenario implementation. (a) Manager locates items about to expire (marked in red circles) and (b) annotates some of those items (marked in larger white circles) (b) Employee retrieves and views the same annotations from a different projector view.

7 Modes of Tag Deployment

The work has been motivated in terms of the commercially-important application of inventory control. But we believe that photo-sensing tags may have many innovative uses, and here we outline broad modes of deployment. Note these are speculative uses, not work done.

Single tags: A single tag per object is adequate for many situations e.g. consider installed projectors projecting product augmentation in a retail store, say on a wall of sports shoes. Each shoe transmits identity, and the projected display automatically adjusts for the spacing of the shoes, with dynamic update of the display as shoes are removed or repositioned.

Multiple tags on a single rigid object: Multiple tags allow computation of an object's 3D pose. Consider the packing task for objects that are tagged with their physical dimensions stored on tags. Since the system can compute the relative pose of each object, it can automatically suggest the next object to be packed and guide its placement by projecting markers, and track what the user is doing. It is typically difficult to determine what the user is doing with a vision-based system alone. As a second example, an environment can be tagged to support precise navigation and task guidance of an autonomous robot .

Ad-Hoc Distributed tags: Some scenarios can involve fairly ad-hoc distribution of tags. Consider a set of tags, each with a temperature sensor, on a surface - this sparse scalar field can be interpolated and projected back onto the surface to show temperature variation, with user selection of a specific point for a readout of the interpolated temperature value. The same tags can be used to find the deformation in the shape of that surface over time, without using a fixed camera or projector.

Interaction between tagged objects: with tags on multiple objects we can record history of placement among neighboring objects.

8 Discussion

Hardware Issues: Optical communication between projector and tag can be affected by ambient light. Wavelength division multiplexed communication is commonly used to solve this problem (e.g. TV remote and IR photo-sensor). The optical communication also gets noisier as projector-tag distance increases, and as the photo-sensor gets dirty.

Projected augmented reality can be adversely affected by ambient light, is poor on low albedo or specular surfaces, and it does not allow private display. Obtaining a visible projection with a small projector strains power requirements - a possible solution is for the projector to beam an image to a bistable display, like eInk, and then switch off while the bistable display continues to show the intended image.

Interface Issues: The handheld projector system lacks text input. Options include hybrid LCD panels for graffiti input, or a speech interface. New types of text-input solution are also appearing e.g. typing onto a projected laser pattern keyboard with finger placement detected by IR sensing [Canesta 2002].

Future Directions: Industry is already addressing range and power issues for passive-RFID. Photosensing tags provide a new type of imaging using distributed single pixel cameras and can solve a variety of computer vision problems. Tag photo-sensors can be modified in various ways – to do motion sensing, by replacement with 2D cameras, or by being made to be sensitive to a narrow band of wavelength so that the projector can transmit optical signals with wavelength division multiplexing. Light emission on passive tags is difficult due to power constraints, but active tags can use LEDs for optical feedback [Moore et al. 1999]. Active actuated surfaces could support scenarios which involve not just data update on tags, but also change in the physical position of objects.

The trend for projectors is miniaturization. Symbol Technologies has a small laser projector, with two tiny steering mirrors for vertical and horizontal deflection [Symbol 2002]. Siemens has a 'mini-beamer' attachment for mobile phones [Siemens 2002]. LEDs are replacing lamps [Lumileds 2003] and reflective displays are replacing transmissive displays (DLPs, LCOS). Both lead to improved efficiency requiring less power and less cooling. DLP and LCOS projectors can display images at extremely high frame rates, currently 180Hz and 540 Hz respectively, but lack video bandwidth. High frame rate will improve image stabilization and allow tracking of moving tags with rapid binary or colored patterns. At the same time, MEMS will miniaturize inertial sensors.

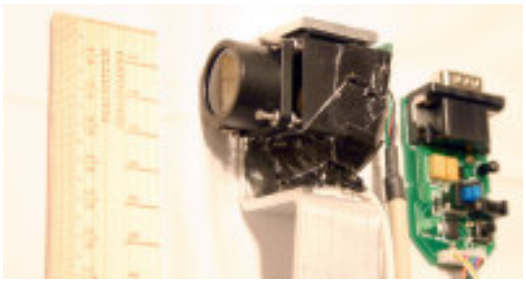


Figure 13: A micro-projector prototype we have built – to indicate the viability of designing a projector for truly handheld use. It is a 1-inch cube using an LED and LCOS imager to create a color animated projection at a distance of about 12-inches.

9 Conclusion

The embedding of computation power and data in everyday environments is now becoming feasible as cost and power problems diminish, and the prototypes in this paper have shown innovative ways to interact with such ambient intelligence. We demonstrated how a wireless RFIG tag system can support geometric operations. We showed a way to extend passive projection of augmented reality to an interactive system, with navigation and update of the augmentation data. The work indicates some of the possibilities for blurring the boundaries between the physical and digital worlds by making the everyday environment into a self-describing wireless data source, a display surface, and a medium for interaction.

Acknowledgements We thank the anonymous reviewers for useful comments and guidance. We thank Cliff Forlines, Rebecca Xiong, Dick Waters, Joe Marks and Kent Wittenburg for stimulating discussions, Mamuro Kato, Takashi Kan and Keiichi Shiotani for providing motivating applications, Debbi vanBaar and many members of MERL for help in reviewing the paper.

References

ABELSON, H., ALLEN, D., COORE, D., HANSON, C., HOMSY, G., KNIGHT, T., NAGPAL, R., RAUCH, E., SUSSMAN, G., AND WEISS., R. 2000. Amorphous computing. In *Communications of the ACM*, vol. 43(5), 74–82.

AZUMA, R., BAILLOT, Y., BEHRINGER, R., FEINER, S., JULIER, S., AND MACINTYRE, B. 2001. Recent Advances in Augmented Reality. In *IEEE Computer Graphics and Applications*, vol. 21(6), 34–47.

BEARDSLEY, P., RASKAR, R., FORLINES, C., AND VANBAAR, J. 2004. Interactive Projection. TR 2004/042, MERL.

BEARDSLEY, P., VANBAAR, J., AND RASKAR, R. 2004. Augmenting a Projector-Camera Device with Laser Pointers. TR 2004/035, MERL.

BIMBER, O., FROLICH, B., SCHMALSTIEG, D., AND ENCARNARO, L. M. 2001. The Virtual Showcase. *IEEE Comput. Graph. Appl. 21*, 6, 48–55.

CANESTA, 2002. Miniature Laser Projector Keyboard. http://www.canesta.com.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random Sample Consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach. vol. 24*, 381–95.

FOXLIN, E., AND NAIMARK, M. 2002. Shadow Effects of Virtual Objects on Real Surfaces with a Handheld Projector. *Unpublished*.

HARTLEY, R., AND ZISSERMAN, A. 2000. *Multiple View Geometry in Computer Vision*. Cambridge University Press.

HINCKLEY, K., PIERCE, J., SINCLAR, M., AND HORVITZ, E. 2000. Sensing Techniques for Mobile Interaction. In *ACM UIST CHI Letters*, vol. 2(2), 91–100.

HOLMQUIST, L. E., MATTERN, F., SCHIELE, B., ALAHUHTA, P., BEIGL, M., AND GELLERSEN, H.-W. 2001. Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. In *Ubicomp*, Springer-Verlag LNCS 2201, 273–291.

LUMILEDS, 2003. (Bright LEDs). http://lumileds.com.

MA, H., AND PARADISO, J. A. 2002. The FindIT Flashlight: Responsive Tagging Based on Optically Triggered Microprocessor Wakeup. In *Ubicomp*, 160–167.

MOORE, D. J., WANT, R., AND ET AL. 1999. Implementing Phicons: Combining Computer Vision with InfraRed Technology for Interactive Physical Icons. In *Proceedings of ACM UIST'99*, 67–68.

NAYAR, S. K., PERI, H., GROSSBERG, M. D., AND BELHUMEUR, P. N. 2003. A Projection System with Radiometric Compensation for Screen Imperfections. In *Proc. ICCV Workshop on Projector-Camera Systems (PROCAMS)*.

OMOJOLA, O., POST, E. R., HANCHER, M. D., MAGUIRE, Y., PAPPU, R., SCHONER, B., RUSSO, P. R., FLETCHER, R., AND GERSHENFELD, N. 2000. An installation of interactive furniture . In *IBM Systems Journal*, vol. 39(3,4).

PATEL, S. N., AND ABOWD, G. D. 2003. A 2-Way Laser-Assisted Selection Scheme for Handhelds in a Physical Environment. In *Ubicomp*, 200–207.

PATTEN, J., ISHII, H., AND PANGARO, G. 2001. Sensetable: A Wireless Object Tracking Platform for Tangible User Interfaces. In *Conference on Human Factors in Computing Systems (ACM CHI)*.

PINHANEZ, C. 2001. The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces. In *Ubiquitous Computing 2001 (Ubicomp"01)*.

RASKAR, R., WELCH, G., AND FUCHS, H. 1998. Spatially Augmented Reality. In *The First IEEE International Workshop on Augmented Reality (IWAR)*.

RASKAR, R., WELCH, G., LOW, K.-L., AND BANDYOPADHYAY, D. 2001. Shader Lamps: Animating Real Objects With Image-Based Illumination. In *Rendering Techniques 2001, Proceedings of the Eurographics Workshop in London, United Kingdom*.

RASKAR, R., VAN BAAR, J., BEARDSLEY, P., WILLWACHER, T., RAO, S., AND FORLINES, C. 2003. iLamps: Geometrically Aware and Self-configuring Projectors. *ACM Trans. Graph. (SIGGRAPH) 22*, 3, 809–818.

REKIMOTO, J., ULLMER, B., AND OBA, H. 2001. DataTiles: A Modular Platform for Mixed Physical and Graphical Interactions. In *CHI 2001*.

RINGWALD, M. 2002. Spontaneous Interaction with Everyday Devices Using a PDA Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings. In *UbiComp*.

SIEMENS, 2002. Siemens Mini Beamer. http://w4.siemens.de/en2/html/press//newsdesk_archive/2002/-foe02121b.html.

SYMBOL, 2002. Laser Projection Display. http://www.symbol.com/products/oem/lpd.html.

TELLER, S., CHEN, J., AND BALAKRISHNAN, H. 2003. Pervasive pose-aware applications and infrastructure. *IEEE Computer Graphics and Applications* (Jul).

THE COOLTOWN PROJECT, 2001. . http://www.cooltown.com/research/.

UNDERKOFFLER, J., ULLMER, B., AND ISHII, H. 1999. Emancipated pixels: Real-world graphics in the luminous room. In *Proc. Siggraph 99, ACM Press*, 385–392.

VERLINDEN, J. C., DE SMIT, A., PEETERS, A. W. J., AND VAN GELDEREN, M. H. 2003. Development of a Flexible Augmented Prototyping System. In *The 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2003*.

WANT, R., HOPPER, A., FALCO, V., AND GIBBONS, J. 1992. The Active Badge Location System. *ACM Trans. Inf. Syst. 10*, 1, 91–102.

WANT, R., SCHILIT, B. N., ADAMS, N. I., GOLD, R., PETERSEN, K., GOLDBERG, D., ELLIS, J. R., AND WEISER, M. 1995. An Overview of the ParcTab Ubiquitous Computing Experiment. In *IEEE Personal Communications*, 28–43.

WANT, R., HARRISON, B. L., FISHKIN, K., AND GUJAR, A. 1999. Bridging Physical and Virtual Worlds with Electronic Tags. In *ACM SIGCHI*, 370–377.

WANT, R. 2003. RFID, A Key to Automating Everything. In *Scientific American*, vol. 290(1), 56–65.

ZHANG, Z. 1999. A Flexible New Technique for Camera Calibration. *IEEE Pattern Analysis and Machine Intelligence 22*, 1330–1334.

Code A : Calibration of a Projector (or a camera)

Input: set of corresponding 3D points `points.{X,Y,Z}` and 2D pixels `pixels.{u,v}`
Output of step 3: Perspective 3 by 4 projection matrix, `ProjMV`
Output of step 3: OpenGL 4x4 matrices, `ProjMat, ModelViewMat`

```
void calculateMatrices()
{
```

// Step 1: Fill Matrix that covers the constraining equations

```
      Matrix lhs(2*nPointsFound, 12);        // lhs=LeftHandSide
      for (int i = 0; i < nPointsFound; i++) {
            // odd rows
            lhs(2*i+1, 1)= points[i].X;
            lhs(2*i+1, 2)= points[i].Y;
            lhs(2*i+1, 3)= points[i].Z;
            lhs(2*i+1, 4)=1;
            lhs(2*i+1, 5)=0; lhs(2*i+1, 6)=0;
                  lhs(2*i+1, 7)=0; lhs(2*i+1, 8)=0;
            lhs(2*i+1, 9)= -pixels[i].u * points[i].X;
            lhs(2*i+1, 10)= -pixels[i].u * points[i].Y;
            lhs(2*i+1, 11)= -pixels[i].u * points[i].Z;
            lhs(2*i+1, 12)= -pixels[i].u;

            // even rows
            lhs(2*i+2, 1)=0; lhs(2*i+2, 2)=0;
                  lhs(2*i+2, 3)=0; lhs(2*i+2, 4)=0;
            lhs(2*i+2, 5)= points[i].X;
            lhs(2*i+2, 6)= points[i].Y;
            lhs(2*i+2, 7)= points[i].Z;
            lhs(2*i+2, 8)= 1;
            lhs(2*i+2, 9)= -pixels[i].v * points[i].X;
            lhs(2*i+2, 10)= -pixels[i].v * points[i].Y;
            lhs(2*i+2, 11)= -pixels[i].v * points[i].Z;
            lhs(2*i+2, 12)= -pixels[i].v;
      }
```

// Step 2: Find u-vector corresponding to smallest singular value (S) (=Solution)

```
      DiagonalMatrix D(12);
      Matrix U(12,12);
      SVD(lhs.t()*lhs, D, U);      // lhs.t() denotes matrix transpose
      int smallestCol = 1;         // Column containing smallest sing. value
      // find smallest
      for (int j = 1; j < 13; j++)
            if ((D(smallestCol)*D(smallestCol)) > (D(j)*D(j)))
                  smallestCol = j;
      ColumnVector S = U.Column(smallestCol);
```

// Step 3: write 12x1-Vector S as 3x4 Matrix (row-wise)

```
      Matrix ProjMV(3, 4);
      for (int k = 0; k < 12; k++)
            ProjMV((k / 4)+1,(k%4)+1) = S(k+1);
```

// Step 4: decompose ProjMV in Proj- and ModelView-matrices

```
double scale=sqrt(ProjMV.SubMatrix(3,3,1, 3).SumSquare());
ProjMV /= scale;

//ProjMV /= ProjMV(3,4);
if (ProjMV(3,4) > 0) ProjMV *= -1;

ColumnVector Q1 = (ProjMV.SubMatrix(1,1,1,3)).t();
ColumnVector Q2 = (ProjMV.SubMatrix(2,2,1,3)).t();
ColumnVector Q3 = (ProjMV.SubMatrix(3,3,1,3)).t();

double q14 = ProjMV(1,4);
double q24 = ProjMV(2,4);
double q34 = ProjMV(3,4);

double tz = q34;
double tzeps = 1;
if (tz > 0) tzeps = -1;

tz = tzeps*q34;
RowVector r3 = tzeps*Q3.t();
double u0 = (Q1.t()*Q3).AsScalar();
double v0 = (Q2.t()*Q3).AsScalar();
double a  = crossNorm(Q1,Q3);
double b  = crossNorm(Q2,Q3);
RowVector r1 = tzeps*(Q1.t() - (u0*Q3.t()))/a;
RowVector r2 = tzeps*(Q2.t() - (v0*Q3.t()))/b;
double tx = tzeps*(q14 - u0*tz)/a;
double ty = tzeps*(q24 - v0*tz)/b;

// create Rotation Matrix and Translation Vector
Matrix RotMatrix(3,3);
RotMatrix = r1 & r2 & r3;
ColumnVector t(3);
t << tx << ty << tz;
```

// Step 5: Expand found matrices to 4x4 matrices

```
      // Projection
      Matrix IntMat(4,4);
IntMat(1,1)=-a;  IntMat(1,2)=0; IntMat(1,3)=-u0; IntMat(1,4)=0;

IntMat(2,1)=0; IntMat(2,2)=-b; IntMat(2,3)= -v0; IntMat(2,4)=0;

IntMat(3,1)=0;   IntMat(3,2)=0;
IntMat(3,3)=-(gfFarPlane+gfNearPlane)/(gfFarPlane-gfNearPlane);
IntMat(3,4)=-2*gfFarPlane*gfNearPlane/(gfFarPlane - gfNearPlane);

IntMat(4,1)=0;   IntMat(4,2)=0;   IntMat(4,3)=-1;  IntMat(4,4)=0;

      // Rotation&Translation
      Matrix ExtMat(4,4);
      RowVector nulleins(4);
      nulleins << 0.0 << 0.0 << 0.0 << 1.0;
      ExtMat = ((r1 & r2 & r3) | t) & nulleins;
```

// Step 6: Set matrices as Current MV/Proj-matrices

```
      for (int l = 0; l < 16; l++) {
           MVMat[l] = ExtMat((l%4)+1, (l/4)+1);
           ProjMat[l] = IntMat((l%4)+1, (l/4)+1);
      }

      // ProjMat has to be multiplied by VP^-1
      mat16dMult( invViewport, ProjMat, ProjMat );
```

// Step 7: Save matrices to file ("IntMat0.dat"/"ExtMat0.dat")

```
      ofstream lfInt, lfExt;
      lfInt.open("IntMat0.dat", ios::out);
      lfExt.open("ExtMat0.dat", ios::out);
      for (i = 0; i < 16; i++) {
        lfInt << IntMat(i%4 + 1, i/4 + 1) << ((i%4-3)?"  ":"\n");
        lfExt << ExtMat(i%4 + 1, i/4 + 1) << ((i%4-3)?"  ":"\n");
      }
}
```

Code B
Vertex Shader for Quadric Transfer in Cg

```
vertout main(      appin IN,
                   uniform float4x4 modelViewProj,
                   uniform float4 constColor,
                   uniform float3x3 A,  uniform float3x3 E,    uniform float3 e)
{
    vertout OUT;
    float4 m1 = float4(IN.position.x, IN.position.y, IN.position.z,  1.0f );
    float4 m, mi ;       float3 m2,mp;     float scale;

    m = mul( modelViewProj, m1);
    m2.x = m.x/m.w;  m2.y = m.y/m.w;       m2.z = 1;
    scale = mul(m2, mul(E,m2));
    mp = mul(A,m2) + sqrt(scale)*e;
    mi.x = m.w * (mp.x)/(mp.z);
    mi.y = m.w * (mp.y)/(mp.z);
    mi.zw = m.zw;

    OUT.position = mi;
    OUT.color0 = IN.color0; // Use the original per-vertex color specified

    return OUT;
}
```

Note describing computing projection matrix from homography

Perspective Warping and One-Pass Rendering used in Office-of-the-Future Stereo Project Pair

Wei-Chao Chen
Original: Nov. 1999
Revised: Aug. 2000
Revised (2): Feb 2002
ciao@cs.unc.edu

This short document describes how to calculate the homography (collineation) between two quadrilaterals and how to extend the 3x3 homography matrix into the 4x4 matrix compatible with conventional graphics pipeline. Homography is commonly known in computer vision literature. Extending 3x3 matrix into 4x4 is due to Raskar[1].

To calculate homography, we need at least four corresponding points between the two quadrilaterals. A convenient way is to use the matching corners of the quadrilaterals.

Let
Coordinates $[u_i,v_i,w_i]$ (i = 0..3) correspond to coordinates $[x_i', y_i', w_i']$ (i=0..3)

We want to solve for homography (collineation) matrix A_{3x3} where:

$$x = A_{3x3}u = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} u$$

General Case:

Let a_{33}=1. Since we are using homogeneous coordinate system, using a_{33} !=1 will have the same result anyway.

Since
x = x'/w'
y = y'/w'

We have

$x = a_{11}u + a_{12}v + a_{13} / a_{31}u + a_{32}v + 1$
$y = a_{21}u + a_{22}v + a_{23} / a_{31}u + a_{32}v + 1$

=>

$x = a_{11}u + a_{12}v + a_{13} - a_{31}ux - a_{32}vx$
$y = a_{21}u + a_{22}v + a_{23} - a_{31}uy - a_{32}vy$

Now, we can write the above equations into the following linear system and solve for A_{3x3} using Gaussian elimination:

$$\begin{bmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -v_ox_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & -u_oy_0 & -v_0y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1y_1 & -v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2y_2 & -v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3y_3 & -v_3y_3 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{31} \\ a_{32} \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Naturally, we can use more than four points and find a least-square solution for A_{3x3}. Given *n* corresponding points between two quadrilaterals $[u_i,v_i,w_i]$ (i = 1..*n*) and $[x_i', y_i', w_i']$ (i=1..*n*), we can write the above equation as followed:

$$CA_v = P$$

where A_v is $[a_{11}\ a_{12}\ \ldots.\ a_{32}]^T$ and C, P are stacked versions of the following matrices:

$$C_i = \begin{bmatrix} u_i & v_i & 1 & 0 & 0 & 0 & -u_ix_i & -v_ix_i \\ 0 & 0 & 0 & u_i & v_i & 1 & -u_iy_i & -v_iy_i \end{bmatrix}$$

$$P_i = \begin{bmatrix} x_i & y_i \end{bmatrix}^T$$

$$C = \begin{bmatrix} C_1 \\ C_2 \\ \ldots \\ C_n \end{bmatrix}$$

and $[x_i, y_i]$ is obtained by perspective division $x_i=x_i'/w_i$, $y_i=y_i'/w_i$. A_v is then solved by:

$$A_v = (C^TC)^{-1}C^TP$$

Projection Matrix for One-Pass Warp

In practice, $[u_i,v_i,w_i]$ can be thought of as the corners of frame buffer(in pixels), and $[x_i', y_i', w_i']$ as the corners of the "desired screen region" (pixels cast onto the corners of the physical screen). We can produce a one-pass rendering using the following approach by using a tracker to measure the coordinates on the corners of the screen.

1. Construct a rectangular screen in real world. Measure the four corners with tracker and find their pixel coordinates on projectors.
2. Establish the world-coordinate (i.e., rendering coordinate) at the center of the screen with x-y plane as on the screen. Find the transformation matrix from tracker to the world coordinate.
3. Calculate the homography matrix A_{3x3}.
4. Left-Multiply the GL_PROJECTION matrix with the following matrix:

$$A'_{4x4} = \begin{bmatrix} a_{11} & a_{12} & 0 & a_{13} \\ a_{21} & a_{22} & 0 & a_{23} \\ 0 & 0 & 1-|a_{31}|-|a_{32}| & 0 \\ a_{31} & a_{32} & 0 & 1 \end{bmatrix}$$

5. The viewing frustrum of the user should be set using the user's location and four "desired" corners of the screen, all in world-coordinate.
6. Translate the model with *–uloc*, where *uloc* is the user location respected to world-coordinate system. Then, call the display function.

Notes: A'_{33} is devised so that the near/far Z clipping plane specified by the user will not be affected too much. In practice, Z clipping plane is less precise after A' transformation if the projectors are heavily keystoned.

References:

[1] Ramesh Raskar, "Immersive Planar Display using Roughly Aligned Projectors", IEEE VR2000.

Bibliography for
Course: 'Spatial Augmented Reality'

(This will be updated as we revise our course material)

References

1. Agrawala, M., Beers, A. C., Fröhlich, B., McDowall, I., Hanrahan, P., and Bolas, M. The Two-User Responsive Workbench: Support for Collaboration Through Individual Views of a Shared Space. Computer Graphics (proceedings of SIGGRAPH'97), pp. 327-332, 1997.
2. Arthur, K., Preston, T., Taylor, R., Brooks, F., Whitton, M., and Wright, W. Designing and Building the PIT: a Head-Tracked Stereo Workspace for Two Users. In proceedings of 2nd International Immersive Projection Tecnology Workshop (IPT'98), 1998.
3. Azuma, R. T., A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments, vol. 6, no. 4, pp. 355-385, 1997.
4. Bajura, M., Fuchs, H., and Ohbuchi, R. Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient. Computer Graphics (proceedings of the ACM SIGGRAPH '92), Annual Conference Series, pp.203-210.
5. Barco, Inc. Baron. Retrieved from the World Wide Web:

   http://www.barco.com/projection_systems/products/product.asp?GenNr=324, 2001.
6. Barco, Inc. Consul. Retrieved from the World Wide Web:

   http://www.barco.com/projection_systems/products/product.asp?GenNr=325, 2001.
7. Bennett, D. Elumens. Retrieved from the World Wide Web: http://www.virtual-reality.com, 2001.
8. Bimber, O., Encarnação, L.M., and Schmalstieg, D. Augmented Reality with Back-Projection Systems using Transflective Surfaces. Computer Graphics Forum (proceedings of EUROGRAPHICS 2000 - EG'2000), vol. 19, no. 3, pp.161-168, 2000.
9. Bimber, O., Stork, A., and Branco, P. Projection-based Augmented Engineering. Human-Computer Interaction International (HCII'01), vol. 1, pp. 787-791, 2001.
10. Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. The Virtual Showcase. IEEE Computer Graphics & Applications, vol. 21, no.6, pp. 48-55, 2001.
11. Bimber, O., Encarnação, L.M. and Branco, P. The Extended Virtual Table: A Optical Extension for Table-Like Projection Systems. Presence: Teleoperators and Virtual Environments, vol.10, no. 6, pp. 613-631, 2001.
12. Bimber, O. and Fröhlich, B. Occlusion Shadows: Using Projected Light to Generate Realistic Occlusion Effects for View-Dependent Optical See-Through Displays. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 186-195, 2002.
13. Bimber, O., Gatesy, S.M., Witmer, L.M., Raskar, R. and Encarnação, L.M. Merging Fossil Specimens with Computer-Generated Information. IEEE Computer, September, pp. 45-50, 2002.
14. Bimber, O. Interactive rendering for Projection-Based Augmented Reality Displays. Ph.D. Dissertation, University of Technology Darmstadt, http://elib.tu-darmstadt.de/diss/000270/, 2002.
15. Bimber, O., Encarnação, L.M., and Schmalstieg, D. The Virtual Showcase as a new Platform for Augmented Reality Digital Storytelling. In proceedings of IPT/EGVE 2003 workshop, pp. 87-95 2003.
16. Bimber, O., Fröhlich, B., Schmalstieg, D., and Encarnação, L.M. Real-Time View-Dependent Image Warping to correct Non-Linear Distortion for Curved Virtual Showcase Displays. To appear in Computers and Graphics - The international Journal of Systems and Applications in Computer Graphics, vol. 27, no. 4, 2003.
17. Bimber, O., Grundhöfer, A., Wetzstein, G., and Knödel, S. Consistent Illumination within Optical See-Through Augmented Environments. To appear in proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'03), 2003.
18. Blundell, B. and Schwarz, A. Volumetric Three-dimmensional Display Systems. New York: Wiley-Interscience, 2000.
19. Bandyopadhyay, D., Raskar, R., State, A., Fuchs, H., Dynamic Spatially Augmented 3D Painting. UNC Chapel Hill Tech Report TR01-006, 2001.

20. Bryson, S., Zeltzer, D., Bolas, M.T., de La Chapelle, and Bennett, D. The Future of Virtual Reality: Head-Mounted Displays Versus Spatially Immersive Displays. Computer Graphics (proceedings of SIGGRAPH'97), pp. 485-486, 1997.

21. Burroughes, J.H. Light Emitting Polymers. Nature, pp. 347, 1990.

22. Canesta. Miniature Laser Projector Keyboard. Retrieved from the World Wide Web: http://www.canesta.com. 2002.

23. Chen, S. E. and Williams, L. View Interpolation from Image Synthesis. SIGGRAPH '93, pp. 279-288, July 1993.

24. Chinnock, C. Holographic 3-D images float in free space. Laser Focus World, vol. 31, no. 6, pp. 22-24, 1995.

25. Cruz-Neira, C., Sandin, D., and DeFanti, T. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. Computer Graphics (proceedings of SIGGRAPH'93), pp. 135-142, 1993.

26. Debevec, P., Taylor, C. J., and Malik, J. Modeling and Rendering Architecture from Photographs. SIGGRAPH '96, August 1996.

27. Debevec, P., Yu, Y., and Borshukov, G. Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping. Proc. of 9th Eurographics Workshop on Rendering, June 1998.

28. Dimensional Media Associates, Inc., Retrieved from the World Wide Web: http://www.3dmedia.com/, 2001.

29. Dorsey, J., Sillion, F. X., and Greenberg, D. Design and Simulation of Opera Lighting and Projection Effects. SIGGRAPH '91, August 1991.

30. Downing, E.A., Hesselink, L., Ralston, J. and Macfarlane, R. A three-color, solid-state three-dimensional display. Science, vol. 273, pp. 1185-1189, 1996.

31. Elings, V. B. and Landry, C. J. Optical display device. U.S. Patent, No. 3,647,284, 1972.

32. Fakespace Labs, Inc. BOOM 3C. Retrieved from the World Wide Web: http://www.fakespacelabs.com/products/boom3c.html, 2001.

33. Fakespace Systems, Inc. ImmersaDesk Series. Retrieved from the World Wide Web: http://www.fakespacesystems.com/products.html, 2001.

34. Faugeras, O. Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, Cambridge, Massachusetts, 1993.

35. Favalora, G.E. Multiplanar autostereoscopic imaging system. U.S. Patent, No.5,936,767, 1999.

36. Favalora, G.E., Hall, D.M., Giovinco, M., Napoli, J., and Dorval, R.K. A Multi-Megavoxel Volumetric 3-D Display System for Distributed Collaboration. In proceedings of IEEE GlobeCom 2000 Workshop, Applications of Virtual Reality Technologies for future Telecommunication Systems, San Francisco, 2000.

37. Feiner, S., MacIntyre, B., et al. Windows on the World: 2D Windows for 3D Augmented Reality. In proceedings of ACM Symposium on User Interface Software and Technology, pp. 145-155, 1993.

38. Figl, M., Birkfellner, W., Ede, C., Hummel, J., Hanel, R. Watzinger, F., Wanschitz, F., Ewers, R., and Bergmann H. The Control Unit for a Head Mounted Operating Microscope used for Augmented Reality Visualization in Computer Aided Surgery. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 69-76, 2002.

39. Fuchs, H., Pizer, S. M., Tsai, L. C., and Bloombreg, S. H. Adding a True 3-D Display to a Raster Graphics System. IEEE Computer Graphics and Applications, vol. 2, no. 7, pp. 73-78, 1982.

40. Funkhouser, T. and Kai, L. (editors). Large Format Displays. Computer Graphics and Applications, Special Issue on Large Displays, vol. 20, no. 4, 2000.

41. Gatesy, S.M., Middleton, K.M., Jenkins, F.A., Jr. and Shubin, N.H. Three-dimensional preservation of foot movement in Triassic theropod dinosaurs. Nature, vol. 399, pp. 141-144, 1999.

42. Gortler, S. J., Grzeszczuk, R., Szeliski, R., and Cohen, M. F. The Lumigraph. SIGGRAPH '96, August 1996.

43. Halle, M. Autostereoscopic displays and computer graphics. Computer Graphics (proceedings of SIGGRAPH'97), vol. 31, no. 2, pp. 58-62, 1997.

44. Heckbert, P. and Hanrahan, P. Beam tracing polygonal objects. Compute Graphics (proceedings of SIGGRAPH'84), vol. 18, no. 3, pp. 119-127, 1984.

45. Hirose, M., Ogi, T., Ishiwata, S., and Yamada, T. A Development of Immersive Multiscreen Displays (CABIN). In proceedings of the VRSJ 2nd conference, pp. 137-140, 1997.

46. HowardModels.com, 7944 Central Avenue, Toledo, OH 43617. http://www.howardweb.com/model/.

47. Howard, W. E. Organic Light Emitting Diodes (OLED). OLED Technology Primer, Retrieved from the World Wide Web: http://www.emagin.com/oledpri.htm, 2001.

48. Hua, H., Gao, C., Brown, L., Ahuja, N., and Rolland, J.P. Using a head-mounted projective display in interactive augmented environments. In Proceedings of IEEE and ACM International Symposium on Augmented Reality 2001, pp. 217-223, 2001.

49. The HyperMask project. http://www.csl.sony.co.jp/person/nielsen/HYPERMASK/.

50. Inami, M., Kawakami, N., Sekiguchi, D., Yanagida, Y., Maeda, T. and Tachi, S., Visuo-Haptic Display Using Head-Mounted Projector, Proceedings of IEEE Virtual Reality 2000, pp.233-240, 2000.

51. Jenoptik. Laser Projector. Retrieved from the World Wide Web: http://www.jenoptik.com, 2002.

52. Kaiser Electro-optics, Inc. ProView SIMEYE XL100. Retrieved from the World Wide Web: http://www.cts.com/browse/keo/index.html, 2001.

53. Kajiya, J. T. The Rendering Equation. Computer Graphics 20(4) (1986), pp. 143-151.

54. Kawakami, N., Inami, M., Sekiguchi, D., Yanagida, Y., Maeda, T. and Tachi, S. Object-Oriented Displays: A New Type of Display Systems -From immersive display to Object-Oriented Displays-, IEEE-SMC '99 Abstracts, Tokyo, Japan, p. 493, October 1999.

55. Kersten D, Mamassian P, Knill D. Moving cast shadows induce apparent motion in depth. Perception, vol. 26, pp 171-192, 1997.

56. Kijima, R. and Ojika, T. Transition between virtual environment and workstation environment with projective head-mounted display, In proceedings of IEEE Virtual Reality Annual International Symposium, pp.130-137, 1997.

57. Kitamura, Y., Konishi, T., Yamamoto, S., Kishino, F. Interactive Stereoscopic Display for Three or More Users. Computer Graphics (proceedings of SIGGRAPH'01), 2001, pp. 231-239.

58. Kiyokawa, K., Kurata, Y. and Ohno, H. An Optical See-through Display for Mutual Occlusion of Real and Virtual Environments. In proceedings of IEEE & ACM ISAR 2000, pp. 60-67, 2000.

59. Klug, M.A., Scalable digital holographic displays, IS&T PICS 2001: Image Processing, Image Quality, Image Capture Systems Conference, Montreal, Quebec, Canada, pp. 26-32, 2001.

60. Knoll, W. and Hechinger, M. Architectural Models: Construction Techniques. McGraw-Hill Publishing Company, ISBN 0-07-071543-2.

61. Knowlton, K. C. Computer Displays Optically Superimpose on Input Devices. Bell Systems Technical Journal, vol. 53, no. 3, pp. 36-383, 1977.

62. Kollin, J. A Retinal Display For Virtual-Environment Applications. In proceedings of SID International Symposium, Digest Of Technical Papers, pp. 827, 1993.

63. Krueger, W., and Fröhlich, B. The responsive workbench. IEEE Computer Graphics and Applications, pp. 12-15, 1994.

64. Krueger, W., Bohn, C. A., Fröhlich, B., Schüth, H., Strauss, W., and Wesche, G. The responsive workbench: A virtual work environment. IEEE Computer, vol. 28, no. 7, pp. 42-48, 1995.

65. Langhans, K., Guill, C., Rieper, E. Oltmann, K., and Bahr, D. Solid FELIX: A static 3D-Laser Display, In Proceedings of Symposium on Electronic Imagining: Science and Technology (IS&T/SPIE), vol. 5006, 2003.

66. Levoy, M. and Hanrahan, P. Light Field Rendering. SIGGRAPH '96, August 1996.

67. Liljegren, G. E. and Foster, E. L. Figure with Back Projected Image Using Fiber Optics. US Patent # 4,978.216, Walt Disney Company, USA, December 1990.

68. Low, K., Welch, G., Lastra, A., and Fuchs, H. Life-Sized Projector-Based Dioramas. Symposium on Virtual Reality Software and Technology, 2001.

69. Lucente, M. Interactive three-dimensional holographic displays: seeing the future in depth. SIGGRAPH Computer Graphics, special issue on "Current, New, and Emerging Display Systems", 1997.

70. Mather G, Verstraten F, and Anstis S The Motion Aftereffect: a Modern Perspective. Cambridge, Mass: MIT Press, 1998.

71. Majumder, A., He, Z., Towles, H., and Welch, G. Color Calibration of Projectors for Large Tiled Displays, IEEE Visualization 2000.

72. Massie, T. H., and Salisbury, J. K. The PHANTOM Haptic Interface: A Device for Probing Virtual Objects. In proceedings of the ASME Dynamic Systems and Control Division, 55(1), 295-301, 1994.

73. McKay, S., Mason, S., Mair, L. S., Waddell, P., and Fraser, M. Membrane Mirror Based Display For Viewing 2D and 3D Images. In proceedings of SPIE, vol. 3634, pp. 144-155, 1999.

74. McKay, S., Mason, S., Mair, L. S., Waddell, P., and Fraser, M. Stereoscopic Display using a 1.2-M Diameter Stretchable Membrane Mirror. In proceedings of SPIE, vol. 3639, pp. 122-131, 1999.

75. McMillan, L. and Bishop, G. Plenoptic Modeling. SIGGRAPH '95, August 1995.

76. Melzer, J. and Moffitt, K. Head-Mounted Displays: Designing for the User. Optical and Electro-Optical Engineering Series, Fisher and Smith (eds.), McGraw Hill, ISBN 0070418195, 1996.

77. Milgram, P. and Kishino, F. A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information Systems E77-D, vol.12, pp. 1321-1329, 1994.

78. Milgram, P., Takemura, H., et al. Augmented Reality: A Class of Displays on the Reality-Virutality Continuum. In proceedings of SPIE: Telemanipulator and Telepresence Technologies, H. Das, SPIE. 2351, pp. 282-292, 1994

79. Mizuno, G. Display device. U.S. Patent, No. 4,776,118, 1988.

80. Naemura, T., Nitta, T., Mimura, A., Harashima, H. Virtual Shadows – Enhanced Interaction in Mixed Reality Environments. In proceedings of IEEE Virtual Reality (IEEE VR'02), pp. 293-294, 2002.

81. Naimark, M. Displacements. An exhibit at the San Francisco Museum of Modern Art, San Francisco, CA (USA), 1984.

82. Newman, J., Ingram, D., and Hopper, A. Augmented Reality in a Wide Area Sentient Environment, In proceedings of IEEE/ACM International Symposium on Augmented Reality (ISMAR'01), , 2001, pp. 77-86.

83. NuVision3D Technologies, Inc. 60GX. Retrieved from the World Wide Web: http://www.nuvision3d.com/the60gx.html, 2001.

84. Ogi, T., Yamada, T., Yamamoto, K. and Hirose, M. Invisible Interface for Immersive Virtual World. In proceedings of the Immersive Projection Technology Workshop (IPT'01), pp. 237-246, Stuttgart, Germany, 2001.

85. O'Rourke, J. Art Gallery Theorems and Algorithms. Oxford University Press, New York, 1987.

86. Panoram Technology. http://www.panoramtech.com

87. Parsons, J., and Rolland, J.P., A non-intrusive display technique for providing real-time data within a surgeons critical area of interest, In proceedings of Medicine Meets Virtual Reality'98, pp. 246-251, 1998.

88. Patrick, E., Cosgrove, D., Slavkovic, A., Rode, J.A., Verratti, T., and Chiselko, G. Using a Large Projection Screen as an Alternative to Head-Mounted Displays for Virtual Environments. In proceedings of CHI' 2000, vol. 2, no. 1, pp. 479-485, 2000.

89. Perlin, K., Paxia, S., and Kollin, J.S. An autostereoscopic display. Computer Graphics (proceedings of SIGGRAPH'00), pp. 319-326, 2000.

90. Pinhanez, C. (2001). The everywhere displays projector: A device to create ubiquitous graphical interfaces. In Proc. of Ubiquitous Computing 2001 (Ubicomp'01), Atlanta, Georgia.

91. Poston, T. and Serra, L. The Virtual Workbench: Dextrous VR. In proceedings of Virtual Reality Software an Technology (VRST'94), pp. 111-121, 1994.

92. G+B Pronova GmbH. HoloPro. Retrieved from the World Wide Web: http://www.holopro.html, 2001.

93. Pryor, Homer L., Furness, Thomas A. and Viirre, E. The Virtual Retinal Display: A New Display Technology Using Scanned Laser Light. In proceedings of Human Factors and Ergonomics Society, 42nd Annual Meeting, pp. 1570-1574, 1998.

94. Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L. and Fuchs, H. The Office of the Future : A Unified Approach to Image-Based Modeling and Spatially Immersive Displays. Computer Graphics (proceedings of SIGGRAPH'98), pp. 179-188, 1998.

95. Raskar, R., Cutts, M., Welch, G., and Stuerzlinger, W. Efficient Image Generation for Multiprojector and Multisurface Displays. In proceedings of 9th Eurographics Workshop on Rendering, pp. 139-144, 1998.

96. Raskar, R., Welch, G., and Fuchs, H. Spatially Augmented Reality. In proceedings of First IEEE Workshop on Augmented Reality (IWAR'98), San Francisco, CA, pp. 63-72, 1998.

97. Raskar, R., Welch, G., and Chen, W-C. Table-Top Spatially Augmented Reality: Bringing Physical Models to Life with Projected Imagery. In proceedings of Second International IEEE Workshop on Augmented Reality (IWAR'99), San Francisco, CA, pp. 64-71, 1999.

98. Raskar, R., Brown, M., Yang, R., Chen W., Welch, G., Towles, H., Seales, B., Fuchs, H.. Multi-Projector Displays Using Camera-Based Registration. IEEE Visualization 99, October 1999.

99. Raskar, R. Welch, G., Low, K.L., and Bandyopadhyay, D. Shader Lamps: Animating real objects with image-based illumination. In Proceedings of Eurographics Rendering Workshop (EGRW'01), 2001.

100. Raskar, R., Ziegler, R, Willwacher, T. Cartoon Dioramas in Motion. Int Symp on Non-photorealistic Animation and Rendering (NPAR), Annecy, France, 2002

101. Raskar, R. van Baar, J., Beardsly, P, Willwacher, T., Rao, S., and Forlines, C. iLamps: Geometrically Aware and Self-Configuring Projectors. To appear in Computer Graphics (proceedings of SIGGRAPH'03), 2003.

102. Reach-In Technologies. The Reach-In Display. Retrieved from the World Wide Web: http://www.reachin.se, 2001.

103. Rolland, J., Rich, H. and Fuchs, H. A Comparison of Optical and Video See-Through Head-Mounted Displays. In proceedings of SPIE: Telemanipulator and Telepresence Technologies, pp. 293-307, 1994.

104. Schmandt, C. Spatial Input/Display Correspondence in a Stereoscopic Computer Graphics Workstation. Computer Graphics (proceedings of SIGGRAPH'83), vol. 17, no. 3, pp. 253-261, 1983.

105. Schwald, B., Seibert, H., Weller, T. A Flexible Tracking Concept Applied to Medical Scenarios Using an AR Window. In proceedings of International Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 261-262, 2002.

106. Segal, M., Korobkin, C., van Widenfelt, R. Foran, J., and Haeberli, P. Fast Shaddows and Lighting Effects Using Texture Mapping. Computer Graphics (proceedings of SIGGRAPH'92), vol. 26, no. 2, pp. 249-252, 1992.

107. Siemens. Siemens Mini Beamer. Retrieved from the World Wide Web: http://w4.siemens.de/en2/html/press//newsdesk archive/2002/-foe02121b.html, Cited December 2002.

108. Silicon Graphics, Inc. Powerwall. Retrieved from the World Wide Web: http://www.stereographics.com/html/body_powerwall.html, 2001.

109. Siio, I., Virtual Glassboat: For looking under the Ground. Human-Computer Interaction International (HCII'01), 2001, pp. 683-687

110. Smith A, and Snowden R Visual Detection of Motion. London: Academic Press, 1994.

111. Starkey D. and Morant, R.B. A technique for making realistic three-dimensional images of objects. Behaviour Research Methods and Instrumentation, vol. 15, no. 4, pp. 420-423, 1983.

112. Stetten, G., Chib, V., Hildebrand, D., and Bursee, J. Real Time Tomographic Reflection: Phantoms for Calibration and Biopsy, In proceedings of IEEE/ACM International Symposium on Augmented Reality (ISMAR'01), pp. 11-19.

113. Stuerzlinger, W. Imaging all Visible Surfaces. Graphics Interface '99, pp. 115-122, June 1999.

114. Summer S. K., et. al. Device for the creation of three-dimensional images. U.S. Patent, No. 5,311,357, 1994.

115. Sutherland, I.E. The Ultimate Display. In proceedings of IFIP'65, 506-508, 1965.

116. Sutherland, I.E. A head mounted three dimensional display. In proceedings of the fall joint computer conference (AFIPS). 33(1), 757-764, 1968.

117. Symbol. Laser Projection Display. Retrieved from the World Wide Web: http://www.symbol.com/products/oem/lpd.html, December, 2002.

118. Szeliski, R. and Shum, H. Creating Full View Panoramic Mosaics and Environment Maps. SIGGRAPH '97, August 1997.

119. TAN Projectiontechnologies. Tanorama Cylinder. Retrieved from the World Wide Web: http://www.tan.de, 2001.

120. TAN Projectiontechnologies. Responsive Workbench. Retrieved from the World Wide Web: http://www.tan.de, 2001.

121. TAN Projectiontechnologies. Holobench. Retrieved from the World Wide Web: http://www.tan.de, 2001.

122. Traub, A.C. Stereoscopic Display Using Varifocal Mirror Oscillations. Applied Optics, vol 6, no. 6, pp. 1085-1087, 1967.

123. Travis, A., Payne, F., Zhong, J., and More, J. Flat panel display using projection within a wedge-shaped waveguide. Retrieved from the World Wide Web: http://ds.dial.pipex.com/cam3d/technology/technology01.html, 2002.

124. Underkoffler, J., Ullmer, B., and Ishii, H. Emancipated pixels: real-world graphics in the luminous room. SIGGRAPH '99, August 1999.

125. Walker, M. Ghostmasters: A Look Back at America's Midnight Spook Shows. Cool Hand Publ., ISBN 1-56790-146-8, 1994.

126. Ware, C., Arthur, K., et al. Fish Tank Virtual Reality. In proceedings of Inter CHI'93 Conference on Human Factors in Computing Systems, pp. 37-42, 1993.

127. Welck, S. A. Real image projection system with two curved reflectors of paraboloid of revolution shape having each vertex coincident with the focal point of the other. U.S. Patent, No. 4,802,750, 1989.

128. Wellner, B. Interacting with paper on the DigitalDesk. Communications of the ACM, 36(7):87–96, July 1993.

129. Wiegand, T. E., von Schloerb, D. W., and Sachtler, W. L. Virtual Workbench: Near-Field Virtual Environment System with Applications. Presence: Teleoperators and Virtual Environments, vol. 8, no. 5, pp. 492-519, 1999.

130. Witmer, L. M. The evolution of the antorbital cavity of archosaurs: a study in soft-tissue reconstruction in the fossil record with an analysis of the function of pneumaticity. Journal of Vertebrate Paleontology, vol. 17, supplement to no. 3, pp. 1-73, 1997.

131. Xerox PARC, Electronic Reusable Paper. Retrieved from the World Wide Web: http://www2.parc.com/dhl/projects/gyricon/ , 2003.