

ABSTRACT

Title of dissertation: SEMANTIC LIGHT:
BUILDING BLOCKS

Charles Lohr, Master of Science, 2010-2011

Dissertation directed by: Dr. Zary Segall
Department of Computer Science and
Electrical Engineering

The concept of *Semantic Light* is simply that lighting systems can be aware of what they are lighting. This offers a number of potential advantages over conventional lighting in quality and efficiency. Semantic Light requires fine grained control of the output of many lights and requires sensors to take in information about what is being lit. It uses this information to control the output lighting in great detail. By running various algorithms, Semantic Light can provide information to the user and has a number of applications including augmented reality.

Traditional lighting that is currently in wide use has limited control of quality and quantity of the light produced. Few lights for large-scale use are intended to control their output in any kind of detailed manner. Most area lighting only has a

switch that must be manually turned on or off. While there are many commercial systems that allow for more fine grained control, they are typically limited to remote control, motion control and extra manual controls. These systems can be wasteful, or they may provide inappropriate amounts of light, or they may be on when no one is using them.

While other Semantic Lighting systems may focus on “green” or power saving aspects, we concentrate instead on innovative roles Semantic Light could play as well as on the technology to make it possible to fill those roles. By emphasizing new utility and maximizing our speed to prototype, we have made several tradeoffs that will cause our system to be less efficient than it could be, even less efficient than traditional lighting systems. The ideas and concepts covered, however, could be adapted to different underlying technologies to produce a product that could provide considerable power saving over conventional lighting.

It is important to think of the many concepts covered as primary building blocks, rather than a complete commercial system. A number of refinements and extensions will be needed to produce a commercial viable product. We demonstrate all of the needed building blocks in a concise, prototyped system.

SEMANTIC LIGHT: BUILDING BLOCKS

by

Charles Lohr

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County in partial fulfillment
of the requirements for the degree of
Master of Science
2010-2011

Defended on May 12, 2011

Advisory Committee:

Dr. Zary Segall

Dr. Yelena Yesha

Dr. Marc Olano

Acknowledgments

I'd like to thank everyone who made this project possible. I am thankful for the support of my friends while working on this project.

Thanks to Dr. Zary Segall for providing the concept and a great deal of guidance while working on this thesis. His guidance was of great value and without him, there is no way this project would have been successful. Thanks to Chad Eby for his help in providing additional resources of people performing similar projects. He also helped greatly with suggestions and directions to go in. Thanks to Will Murnane for helping greatly with the editing of this thesis.

Thanks to Dorothea F. Brosius of the Institute for Electronics and Applied Physics, University of Maryland, College Park, MD for creating this template in L^AT_EX. It relieved a great amount of work when getting started.

And lastly, thanks to my parents for allowing me to use a fairly large area in their home to perform the research. They even had a hand in commenting on some of the early drafts.

Table of Contents

List of Figures	iv
1 Introduction	1
1.1 Conventional Lighting	1
1.2 Lighting Technology Methods	2
1.3 Accent Lighting	6
1.4 Projectors	6
1.5 Semantic Light	8
2 Previous Work	12
2.1 General	12
2.2 Non-Standard Huan-Machine Interfaces	12
2.3 Color Perception and Color Temperature	17
2.4 3D Projection Mapping over Arbitrary Surfaces	19
3 Overall Process	21
3.1 Overview	21
3.2 Fiducials	23
3.3 Object Recognition	24
3.4 Coordinate Mapping	32
3.5 User Interface	35
3.6 PDF Search Application	37
4 Future Work	46
5 Conclusions	53
A Bill of Materials	55
B General Notes	56
List of Abbreviations and Glossary	58
Bibliography	59

List of Figures

3.1	Overview of system	22
3.2	Work-flow of System	22
3.3	Overview of a fiducial.	24
3.4	GLSL Image Processing Pseudocode	28
3.5	Raw output from camera	29
3.6	Output from GLSL Shader	29
3.7	Dot Identification Pseudocode	31
3.8	Fiducial Finding Pseudocode	39
3.9	CollectBits function	40
3.10	Forward mapping map	41
3.11	Mapping example	42
3.12	Remapped scene into projector space	43
3.13	Projector output for PDF tool	44
3.14	Camera input for PDF tool	45
3.15	Visible light picture of PDF tool	45

Chapter 1

Introduction

1.1 Conventional Lighting

Conventional lighting typically uses incandescent or fluorescent lighting to light an area. This lighting is typically controlled through switches, timers, motion sensors, and other simple controllers. In some cases the lighting control is tied into a central system, but, even then, it is rare for the lighting to be controlled in any kind of context sensitive manner.

While great strides have been made in dispersing the light efficiently (light diffusion gratings, etc.), light still comes out of the source non-uniformly. The color temperature and brightness of the light itself typically cannot be controlled on a small scale. There have been a number of innovations to aid in solving some of these problems. One example is the use of accent lighting. This lighting is typically dimmable in order to allow people occupying the room to manipulate the overall appearance of the room.

These innovations still leave much to be desired. The control for these systems is typically manual, and usually only allows for dimming for an entire set of lights. While the lights are usually individually aimable, there is usually no system which can automatically re-aim them. Typically these accent lighting systems use incandescent bulbs, which are terribly inefficient and generate large amounts of heat. In

order to overcome these shortcomings, we use a DLP projector that makes use of a metal halide lamp. Ideally, a low-resolution, all-LED matrix could be used for pinpoint lighting. While the LED matrix would not have as fine grained control over an area as the DLP, it would still be a great improvement over current systems. It would also be an improvement over DLP due to its added efficiency.

1.2 Lighting Technology Methods

There are many technologies used for lighting. The market is currently dominated by incandescent and fluorescent lighting. Due to recent legislation, lighting will be shifting more toward the fluorescent end, as many types of incandescent lamps are going to be banned in the near future [1, 2]. Because there are many applications for which fluorescent lights are inappropriate, this may give rise to more demand for LED lights, driving the cost of LED lamps down further. We hope that in the coming years, LED technology will continue to become cheaper, more efficient and able to be computer controlled in greater detail. This section goes over some common lighting mechanisms along with the pros and cons of each. Table 1.2 contains rough numbers for many different types of lighting. Many forms of lighting are not generally used for area lighting, but they may be used for decorative or accent lighting.

Light Type	Efficiency	Granularity (size/time)	Controllability
Incandescent	5-15 lm/W	Medium	Medium
Halogen	5-24 lm/W	Medium	Medium
Fluorescent	50-90 lm/W	Medium	Poor
Gas Discharge*	10-78 lm/W	Medium	Medium
HID	65-150 lm/W	Poor	Poor
LED*	55-110 lm/W	Good	Good
DLP*	0-15 lm/W	Excellent	Excellent

* = Not typically used for area lighting.

Table 1.1: Typical characteristics of lighting products.

Incandescent Lights

Incandescent lamps have been commercially produced for over 100 years, and have changed very little in that time. A tungsten wire heats up because of an induced current. The mechanism for light production is simply that the filament gets “white hot.” Most of the light emission is in the infrared range, thus reducing even the most efficient incandescent bulbs down to 17.6 lumens per watt, or 2.6% efficient [3]. Even halogen and other slightly higher efficiency bulbs do not have efficiency over 30 lumens per watt. Because the filaments of light bulbs act as black body radiators, their color profile is very similar to that of the sun. Since our eyes have evolved to be comfortable with a color profile similar to that of the sun, it is to be expected that our eyes will be more comfortable with that color profile. For

our applications, incandescent lights would be suitable because they can easily be turned on and off quickly. However, they would be inappropriate in many other ways because of their size and its low efficiency.

Plasma based lights

Fluorescent lighting has been around for nearly the same length of time, and it uses a different mechanism for emitting light. Inside a fluorescent lamp, a high voltage excites atoms such as mercury, neon, etc., causing them to emit a narrow-wavelength band of light. Most common fluorescent area lights still use mercury, which emits light in the ultraviolet spectrum. Phosphors takes this ultraviolet light and convert it into a variety of other wavelengths so that it appears white. The “white” color produced by fluorescent lights is not described by a smooth function, it is made up of peaks. The eye’s ability to only perceive distinct dimensions of color causes humans to see it as white. Fluorescent lamps achieve much higher efficiency than incandescent lights, typically 50-90 lumens/watt. Fluorescent lamps are not suitable for what we hope to achieve, as they do not dim well, are not easily aimed, and have a limited range of colors that they can produce.

Gas discharge lamps, such as neon, xenon, etc., are very similar but require a higher voltage to operate. They differ in that they do not require phosphors, since they are typically used to make various colors. Phosphors may be used to obtain different colors. They typically do not achieve fluorescent levels of efficiency. The peak efficiency of a tri-phosphor neon lamp is 78 lm/w [21]. These types of lights are

typically not used for area lighting because of the cost and complicated installation process.

HID or High Intensity Discharge lamps, which include mercury vapor, metal halide and high pressure sodium lamps are common in large area outdoor lighting. They have longer life, yet are less efficient and take longer to turn on than fluorescent lamps [4]. They are ideal for their application but do not apply well to our project.

LED Lights

We will primarily focus on LED lights since they are the most applicable to our project because of their speed, size, dimming and efficiency. LED lights use solid state technology to emit light. They have no moving parts, high temperatures or state changes. Instead the light is emitted by exciting a carefully crafted diode. LEDs have a long life (50,000 hours+) and can have high efficiency (80+ lumens/watt) [5]. Additionally, because LEDs are solid state, they can be turned on and off at high speeds without being damaged or having their expected life reduced. LEDs can have a very high energy density, making it possible to focus them accurately.

All LEDs can only produce a single wavelength of light. Most commercial “white” LEDs only contain a blue LED element and a phosphor coating to control the color temperature. Alternatively, if color changing is desired, LEDs can produce any color—including white, or off white—efficiently from red, green and blue LEDs. This gives us excellent control over the spectrum.

We can simulate most of the properties of LED lighting with our DLP projector. For many of our examples, however, we use a higher density grid than would be practical with LED systems.

1.3 Accent Lighting

Designers typically focus on light as more than a means to allow people to see things. When designers do not have enough control using area lights, they can turn to accent lighting. Accent lighting is used to highlight certain objects, such as plants, pictures, or specific areas of a room, etc. Many times this type of lighting is controlled independently from the large area lighting. This allows users to control the lighting of their surroundings to a limited extent. We hope to greatly improve this, allowing fine tuned control of lighting the users' surroundings and relieving them of the need to control it themselves.

1.4 Projectors

Projectors are a means of casting light out into a scene in a highly controlled manner. All current projectors do this by starting with an intense white light, then casting it at an element to remove light from certain areas. This makes projectors intrinsically very inefficient. For example, they typically use as much power when projecting mostly black as they do when projecting mostly white, although many fewer photons are released from the projector. This, in addition to the complicated optics, makes projectors very inefficient for area lighting purposes. We've chosen to

use a projector to prototype our system because of the incredible degree of flexibility and accuracy that projectors offer. The only reason our technology probably should not use a projector in its final form is because of the low efficiency and high cost that a projector entails.

Most projectors currently use a halogen or metal halide bulb as the primary source of light. This makes it difficult to dynamically change the overall brightness of the scene efficiently because the internal bulb’s brightness cannot be changed quickly enough. As stated before, this causes dark scenes to be incredibly inefficient to create. There is an alternative to metal halide that is coming—LED projectors [6]. By using sets of RGB LEDs as the lighting elements, a number of efficiency and quality benefits can be obtained because LEDs can be pulsed. Commercially available LED projectors currently have poor optical coupling of the lighting element and typically do not use RGB LEDs, but hobbyists have created prototypes improving on this. Because of the current impracticality of LED projectors, we have chosen to use a traditional DLP solution.

DLP

DLP stands for Digital Light Processing. It is a technology used by many current generation projectors. It was developed by TI and uses an array of up to two million hinge-mounted microscopic mirrors to turn individual pixels on or off [22]. It turns these mirrors on and off several thousand times per second, and can represent colors by use of a color wheel. The color wheel only permits red, green

or blue light through at any given time. When the color wheel presents a color, the DLP adjusts the mirrors to represent the corresponding image for that color.

While any projector technology could be used for our prototypes, DLP enjoys the best contrast ratio and most vivid colors of any current technology. This enables us to have a high degree of flexibility with our lighting. While this technology is inefficient in this application, it gives us a suitable platform on which to prototype our system. We will use DLP to develop and demonstrate all of the various technology required for Semantic Light. In real-world applications, a DLP projector would probably never be used for lighting because of its high cost and poor efficiency; it is used strictly as a prototyping mechanism.

1.5 Semantic Light

In discussing Semantic Light, we hope to address shortcomings in current accent and area lighting systems. This type of lighting will give us fine grained control of light over an area. It will also enable a computer to decide the lighting requirements for an area, relieving humans of the need to control this fine grained control over lighting. By using a computer to intelligently control the light, great strides in the quality of lighting and increased dynamic control can be achieved.

In order to achieve this, a number of hurdles must be overcome. We will discuss these hurdles here, and they will be revisited throughout the course of this paper.

Fine-Grained Control

In order to enjoy many of the benefits of Semantic Light, fine grained control is needed. There is no specific requirement for a particular level of control. However, more is always helpful. With more individual lighting zones comes more ability. A coarse system may only be able to light up a soft circle around a page of paper, whereas a fine system may be able to light up just the page itself, or even highlight items on that page. The ability to control more than a handful of individual lights is critical.

We will be focusing on a fine grained system for our prototype. The projector we are using has a resolution of 1920x1080. This means that there are roughly two million lighting elements that can be individually controlled. Because of our hardware limitations, namely a GeForce 8600 in the computer we were using, we were only able to operate at a resolution of 1280x720. This, however, does not have much of a negative impact since the resolution could be substantially lower and still achieve our goal. If a system was expected to project detailed text onto a surface, the higher resolution would be needed.

User Control

While Semantic Light removes the need for a user to have detailed control over the individual elements of a lighting system, it is still useful for a user to be able to give the computer high-level hints as to how a scene should be lit. A user may want more or less light in specific locations, or even colors in a manner that is

not algorithmically predictable. In order for a Semantic Light system to be useful, it must therefore be able to take user input. Ideally, this user input should not require the use of traditional input devices (i.e. mouse, keyboard) for general use. However, these systems may be useful if the system requires textual input once in a while. Devices such as a remote control could be used and are an appropriate interface mechanism for this type of system, but it may be the case that even they may become obsolete by the use of a more advanced or device-free user interface.

We propose that the user control could be derived from images of the scene itself. A user should be able to move sheets of paper, fiducials, or use hand gestures to give the system cues. The interface needs to be intuitive purpose-specific. Systems that have very fine grained control will allow the user to interface with buttons, whereas more coarse systems must rely on the motion of objects and other mechanisms.

We will be focusing on a finely controllable system, and will provide buttons to the user. The user can interact with these buttons and will allow the user to give the system input in a variety of interesting ways.

Automation

Users need to be able to interact with the system to give it high-level input. However, even more critical is automation of the system. We need an algorithm to be able to take very few inputs, such as those provided by a user, and to control a large number of outputs, such as each of the lights. This automation will be

the other main focus of our paper. It will make the user’s experience much more enjoyable. While the behaviors implemented by our algorithm may be somewhat simplistic, we hope to lay the groundwork for much more advanced control systems.

Performance

The system needs to have very low latency. If there is a perceivable lag between the user’s input and the system’s output, it will prove to be a suboptimal interface and not provide the user with a comfortable experience. If the turn around time is much more than a few tens of milliseconds, the lag will likely be noticed by the user of the system, yielding a negative impact on the user’s experience.

Chapter 2

Previous Work

2.1 General

While there are many sources of innovative work in the technical areas with which we are concerned, not all of them are academic papers. Many current state-of-the-art implementations exist primarily in project form and may not have academic papers. In other cases, academic papers are available, but they are not as comprehensive as the other media that discuss the technology. Instead, authors have used alternative mechanisms for publishing their work, such as direct-to-Internet. This makes finding academic citations on some related topics difficult.

2.2 Non-Standard Huan-Machine Interfaces

There has been extensive experimentation with a variety of control mechanisms for computers that go beyond a normal mouse and keyboard. Much of the experimentation started with large-scale multi touch monitors. Rather than using capacitive and resistive touchscreens, which are common and have a high degree of accuracy with minimal cost [7], there are interesting ways of handling user input in large areas. A few tactics relevant to the proposed system have been implemented, we will discuss the systems which we took inspiration from.

Cameras have been an attractive solution for these large area input devices for some time. BULLSEYE [8] uses actual 3D props for specialized and unusual human-machine interfacing. It considers many properties of these props to determine their location, orientation and other properties. Because this system works in 3D and is not really geared for user input, it does not require the camera to be in the same place as the display.

Many of these systems use color fiducials because of the three dimensions of color. This is to allow tracking of more objects without the need for high resolution cameras [10]. They use transparent video devices for the main display. This mechanism is also used in Billinghamurst and Kato's work [9], but their mechanism for tracking is a more along the lines of what we wish to accomplish. They use fiducials to track objects in camera space. They were able to use shapes to increase the the quality of image.

All of these tactics use visible light. We wish to make a system that can operate in a dark room, or operate in an area with very uneven light; therefore, we cannot operate in the visible spectrum. We chose to focus on the infrared spectrum. While we do not have many unique colors to work with, we were able to focus on a more structural approach. FTIR, the Wiimote systems and reacTIVision were our primary inspiration for our mechanism of tracking objects.

FTIR Camera-based Systems

FTIR Stands for Frustrated Total Internal Reflection. In this system, light is trapped inside a clear material because of its refractive index. Jeff Han [11] was one of the first to make progress into a practical system that used FTIR. His work involved a screen with an image projected onto it, attached to an acrylic sheet. The acrylic sheet is constructed such that infrared light shone inside of it will not leave due to the refractive index of the sheet-air interface. Infrared light shines in the edge of the sheet and becomes trapped, and cannot escape out either face. By touching the surface, the user modifies the refractive index of the interface at that location, and this causes light to scatter out the back of the enclosure. By using infrared light to illuminate the sides of the enclosure, the image being projected to the user does not interfere with user interactions.

The escaped IR light is received by a camera on the back of the enclosure and processed. This camera needs to capture at high speeds and have a relatively high resolution in order to be able to accurately capture user input at a speed suitable for general use. Once the data is captured, computer algorithms determine where the users are touching the screen and what is simply noise. Input is then coalesced into a series of points and then provided to Han's software. His work has been used by many hobbyists to great success and is the foundation for many large scale touchscreen systems. It has even been turned into a company called PerceptivePixel.

Training on this system is typically performed by selecting a handful of points to tell the computer the camera-space locations of the screen-space points.

Point-Light Tracking Systems

There is an interesting topic that has attracted a great deal of interest over the last two decades: human motion capture. There are a number of different forms of motion capture, mechanical, ground-based, electromagnetic, acoustic, image-based and optical [23]. Both the image-based tracking methods are very attractive to us because they are not very obtrusive. Most of the image-based systems use illuminated markers placed at specific locations on the body.

Using multiple cameras or other light detecting devices, the absolute location of all of the markers can be found. Most are capable of differentiating the points and do not rely on continuously tracking them to make sense of which point is which. We feel an image-based approach is best for us when tracking objects such as fiducial dots. Most of these approaches emit light at points of interest, we focus on locating a dearth of infrared light at points of interest.

In a similar concept, Johnny Chung Lee uses the Wiimote's infrared camera made to track IR lights [12]. The obvious difference in his work is the use of the Wiimote's camera. Rather than doing the processing on a raw video signal, the video can be processed by a separate module and can easily return the location of up to four unique points at a resolution of up to 1024 x 1024.

The more subtle change is in Johnny Chung's use of individual LEDs to emit the light rather than relying on scattered light from FTIR. This enables a much more precise tracking of points because the spot produced in the camera image is smaller. The only drawback to this is that the system can no longer involve only

the human body as the only source of input, at least without adding reflective tape, etc. However, it has the advantage that it can be used without being attached to a screen.

Like the FTIR mechanism, training is done on only a handful of points. This system also does not cater very well to our project because the targets need to be powered and the system can detect a maximum of four points.

reacTIVision

A relatively new system that is very close to what we are attempting to achieve is reacTIVision. This is a system oriented for use on a table and uses fiducials in order to locate tagged objects and allow interaction with those objects [13]. reacTIVision proposes what they call a TUI or Tangible User Interface. Our system is somewhat similar to this in concept. Their system is able to have many unique tags and can detect people's fingers.

reacTIVision's fiducials can be uniquely identified by using topological segmentation [14]. Topological segmentation allows us represent unique entities as different topological graphs. This allows fiducials to be very robust under glancing viewing angles and be completely rotation independent. This is done by representing parts of a topological graph as shapes on the image. Many different images can produce the same graph, and when trying to distinguish two different fiducials from one another, the graph of the fiducials trying to be identified is compared to all graphs in the library. Though this type of fiducial's properties are generally good, they

provide limited storage and error correction.

The reacTIVision implementation of amoeba-form fiducials has certain drawbacks for our purposes. While these fiducials are exceptional for image processing and robustness, they occupy a large area on the page. They need to be round and cannot be made into a shape ideal for integration alongside text on a page. They are also susceptible to errors when there are additional markings nearby in the image.

In addition to these technical issues with the application of reacTIVision to the proposed system, the library would be difficult to interface with and does not provide a low enough latency to allow for seamless integration. Because of this we intend to learn from reacTIVision, but not follow it directly.

2.3 Color Perception and Color Temperature

When we see color, we are seeing light, of three dimensions of light [25]. We primarily detect color using our cones. These distinct dimensions of light are generally subdivided into red, green and blue; also known as the primary colors of light. We can see many more colors by mixing these colors together. Another note may be made in that most video output devices to computers cannot faithfully represent the entire color space, but instead only cover most of it. While for our purposes we will be working in a limited RGB space for many purposes, however, it may make sense to work in another color space in the future.

There has been much research on the human perception of color and light. The eye has varying sensitivity to many wavelengths, and it uses this sensitivity

to perceive what we understand as color. CIE 1931 [15] is one of the most notable studies in this area. This was a comprehensive study that looked deeply into how the eye perceives color in a scientific way. It serves as a foundation for color representation in many fields of engineering. This study provided insight into the best ways to represent colors in computer graphics and in lighting. The study also provided understanding of the perception of color temperature and how the brain comes to understand the shapes of surfaces based on color temperature. There is a great deal of information that our brains derive from the overall balance of colors.

The colors we call white are not defined by a single spectrum. There can be warm whites, almost a yellowy or orangey color, such as when the sun is low in the sky or cool whites, like the blue-tinted xenon lamps on cars. Both convey different moods and can have a large impact on a user's senses [17]. When lighting a scene, it is important to keep in mind the effect that the color temperature will have on the people in that area.

A better understanding of color temperature may be found in our discussion of black body radiation. Much of the original work in this arena was done by Kirchhoff in the mid-1800's [16]. This was largely based on the physical phenomenon of black body radiation, which is when hot objects radiate energy in the form of light, where the spectrum of the light is mostly dependent on the temperature. The color produced by an ideal black body radiator at a particular temperature Kelvin is called the color temperature of the light that the radiator emits.

There is value in using other colors than just various shades of white when lighting a scene. Itten points out [17] a number of interesting physiological effects

on people and animals when “warm” or orange-red colors are used versus “cold” or green-blue colors. People even perceive the temperature of their surroundings to be cooler or warmer depending on the color of the room they are in. The effects of non-white light are outside the scope of our work at this early stage. For more information on this, see Chapter 4.

Johannes Itten and Faber Birren have done work to understand the perception of color temperature without reference to black body radiation. They found that people are affected by the apparent coolness or warmth of colors in a room. Some notable effects are that people feel cold at different temperatures depending on the colors used within a room. Warmer colors cause people to feel cold at lower temperatures than cool colors.

2.4 3D Projection Mapping over Arbitrary Surfaces

The area of projected imagery has been explored for over 10 years. Brown [24] discusses a number of topics directly related to our system. While they describe a large complicated view of what could be done with some of the systems we’ve developed here, they present many concepts that are later presented in more detail in later works such as radiometric compression and basic warping of images for projecting onto 3D surfaces using multiple projectors. Because their system uses light in the visual range, the projectors are capable of self calibration.

There has been extensive research in more practical projection of images and textures onto arbitrary surfaces by a project called NuFormer [18]. Their focus was

on their system’s artistic and entertainment value. Their system requires that a model of the building they will be projecting onto be created beforehand, to make it possible to decide how to project light onto the surface of the building from one or more projectors. They do not include the information necessary to map the building into projector space. We will not need to handle anything quite as complicated as what they are doing, and the end goal is different. For this system, we only care about mapping to camera space, not to world space.

Much of the NuFormer work appears to be based on other previous 3D Projection Mapping projects. One of those works is sARc, Spacial Augmented Reality for Architecture. In this paper the authors go into great detail over topics that we do not need to focus on such as matching focal planes, geometric warping and radiometric compression [19]. While radiometric compression is something we could use, it is considered to be out of the scope for this project. Additionally, their project focused much more on faithful image presentation on arbitrary surfaces. We wish to focus particularly on lighting. Therefore we do not need to be as concerned with the details needed to faithfully represent the image.

Chapter 3

Overall Process

3.1 Overview

We intend to address only a small aspect of the field of Semantic Light. Our system is intended to be a proof-of-concept for many aspects of Semantic Light. Specifically, we will use a camera that can look at pre-tagged items in a scene and then identify them as unique entities. Once this is done, the system can observe their motion and respond by lighting the objects accordingly. Our system needs to accomplish object recognition, addressed in Section 3.3, forward mapping from camera space to projector space in Section 3.4, and drawing and clipping in accordance with the mapping in sections 3.5 and 3.5. There are a number of supporting features that need to be present, and they will be discussed as they are encountered.

Our system is implemented using a single OpenGL interface written in C++. It is designed as a monolithic application with modularity in the code. All of its calculations are performed inside a single program in order to minimize complications due to inter-process communication. Additionally, we chose C++ and OpenGL rather than an interpreted language because of the need for high performance. Too much latency between incoming of frame and output of video would add lag, and that lag would make the interface feel slow and unresponsive.

A picture of the system in action can be seen in Figure 3.1. All of the visible

entities, other than fiducial dots, are being projected from the projector above it.

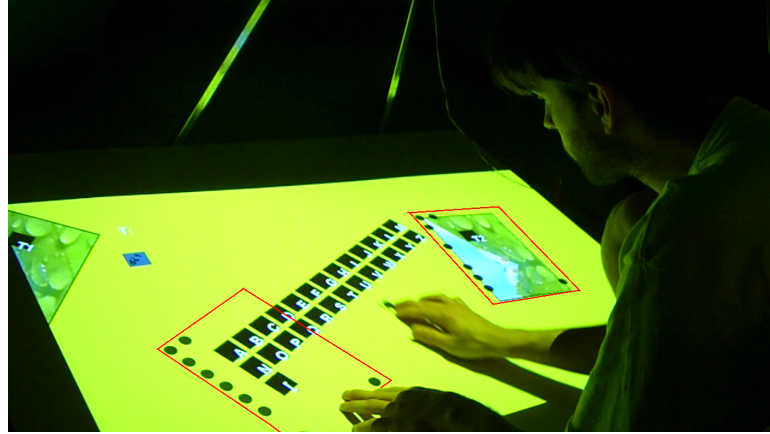


Figure 3.1: *Overview of system using basic user interface. Papers are outlined in red. They are all white except for the dots.*

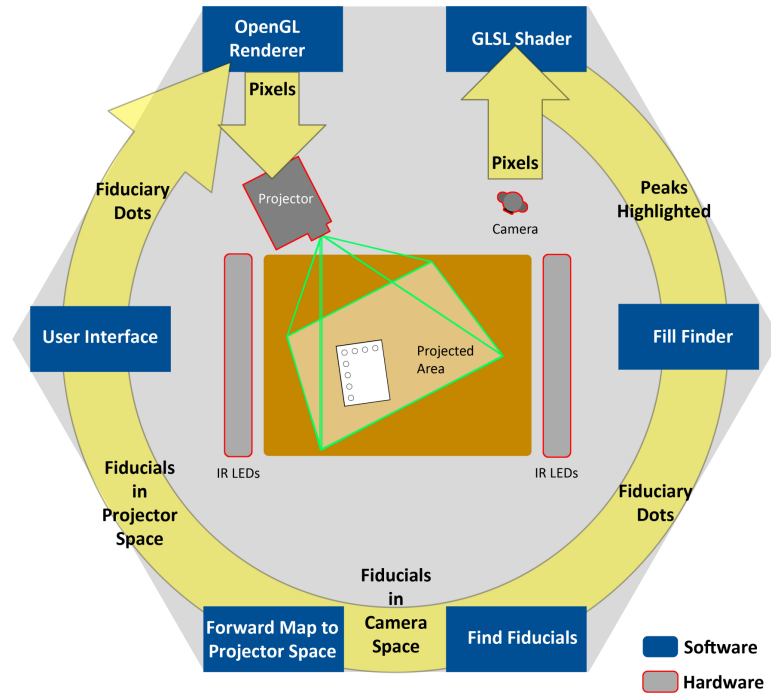


Figure 3.2: *Work-flow of System*

Figure 3.2 shows a block diagram of how the system works. While there is

some global information needed at several steps, the primary progression is quite linear.

3.2 Fiducials

Fiducials can be geometrically dependent or independent. If a fiducial is geometrically independent, then its features can occur anywhere spatially around the fiducial. However, if it is dependent, then the features must appear in specified places relative to other features. Our fiducials are very simple. Rather than using truly geometrically independent symbols like those used by reacTIVision (2.2), we have opted to use fiducials that are geometrically dependent, yet rotation independent. Our fiducials are composed of ‘L’ shapes, which are ideal for printing onto the edges of paper and for having a higher data density. This is demonstrated in Figure 3.3.

The dots in Section B of Figure 3.3 are data bits and present or not present and do not heavily affect the fiducial detection system. It is advantageous, however for there to be at least a few data bits set to allow for more accurate determination of the angle of the fiducial. By having black dots further from the corner of the ‘L’, the arctangent values are comparatively more precise.

Although we could use a system that supports error correction to handle mild data corruption without loss of fiducial tracking, this is not currently implemented. This is because we did not see a critical need since our system works reasonably well without it. This, however, would greatly reduce the confusion of fiducials. As it

stands now, it is possible for the system to believe one fiducial is actually another, which can cause jittering. This is something to address in the future.

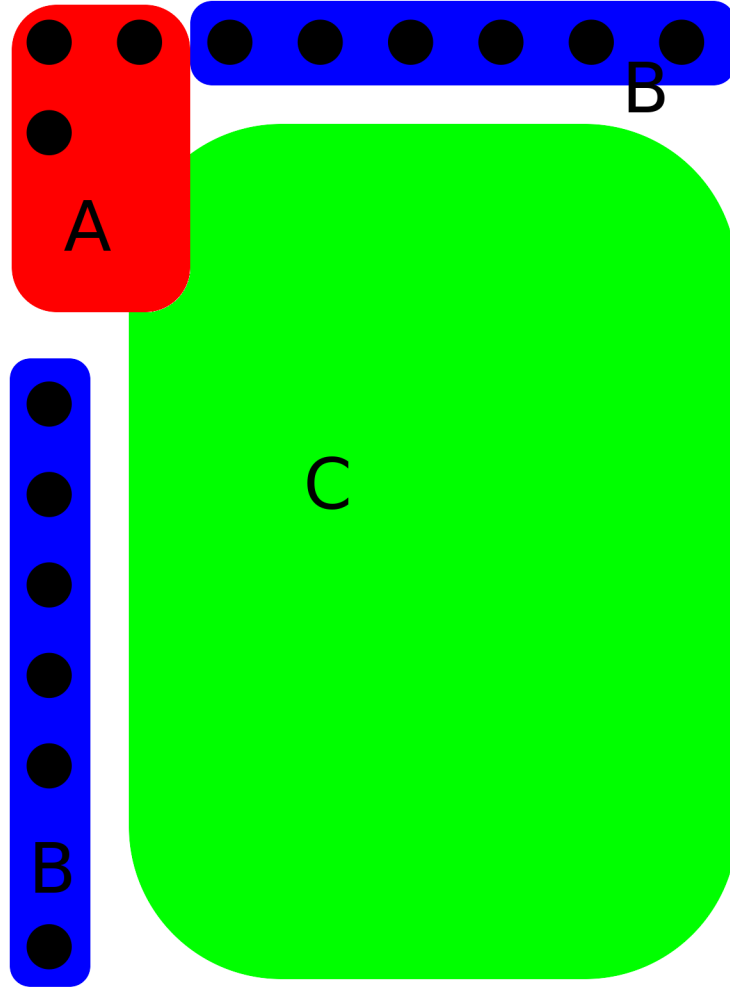


Figure 3.3: *Overview of a fiducial. A is the unique ‘L’ marking, B is the data bits, and C is free area that is ignored by the system but can be used by people.*

3.3 Object Recognition

The goal of Object recognition is to provide information about the location of all tagged objects in a scene from information acquired by a camera. Because

of the need to see only tagged objects and not interference from the projector, we chose to use infrared light to illuminate the scene and to use markers that respond to it. Because the camera does not take pictures as RGB, and instead uses yuv422, we then process the raw image data into RGB for analysis. Once the analysis was complete, OpenGL highlights all fiducial dots.

A CPU Algorithm looks for and finds the location of the fiducials, weeding out all the highlighted areas that are just noise. It does this by searching the scene for a marked area. Then, it fills that area, while determining its size, and marks its center as a potential fiduciary dot. This algorithm is run once per frame. Therefore, it is imperative for all operations to happen quickly. Special care was needed when crafting this code in order to minimize memory accesses and jumps and determine the needed speed.

Infrared Illumination

Because we need to locate geometric shapes in the scene, we do not want to allow light from the projector to create false positives. Our solution was to use a wavelength of light that is not emitted by the projector. We decided to use 850nm infrared LEDs to provide light to the scene. While the entire room could appear black to a user, the room would, in fact, be lit by infrared light.

The system relies on the LEDs to illuminate the scene whenever the system is in use. Our LEDs use approximately 20 watts to light the entire area, which is more than we would like. We considered only lighting the scene when the camera's

aperture was open. However this proved to be too time consuming to implement. While we were able to isolate the shutter signal inside the camera, we were not able to blink IR LEDs to this shuttering. We expect this enhancement could provide a boost in efficiency and possibly quality.

Camera Modification

The system should have a camera that can read pictures at a resolution of 640x480 or more, at high frame rate of 30 FPS or greater, cost a reasonable amount of money and be able to take these pictures in infrared. We were unable to find a camera to meet these requirements. We chose a Logitech® QuickCam® Pro 9000 an off the shelf, widely available webcam. It is able to capture frames at the desired resolution effectively, and it has exceptional optics. This camera, like almost all cots cameras, has an IR block filter, thus not exactly what we needed.

To solve this problem, we had to disassemble the camera, remove the IR block filter, add appropriate optics-grade glass, and then reassemble the camera. This allows the camera to capture both visible light and infrared light. Next we added a 750nm IR pass/visible block filter in front of the camera, allowing only infrared light to pass through to the camera. By using this filter, we can now use the projector to light the scene however we wish, without affecting the light the camera is receiving. This also has the added benefit of allowing the camera to view the scene with no visible illumination.

In Figure 3.5, one can see the pieces of paper with the fiducials clearly, but

not see the projected image. This occurs because of the infrared filtering.

Camera Processing and OpenGL Enhancement

The information coming from the camera has poor contrast due to abrasion of the lens and uneven lighting of the scene. This means fiducials may often be missed in one area or erroneously observed in others.

Our algorithm reads a 24-bit image representing the infrared light received by the camera and provides a 1-bit map of areas the it believes could be fiducial dots. It does this by looking for round depressions of light. This is accomplished by convolving a small Gaussian-like kernel across the image, and at each point calculates the difference between this Gaussian-like filter and the actual value at that pixel. This causes areas that are darker than their surroundings to be marked. Fiducial dots pop out fairly effectively in this process.

The pseudocode in Figure 3.4 shows our algorithm. The resulting map typically has spurious noise intermixed. It is necessary to do further processing in order to find the individual dots. This process is done in OpenGL because the convolution of the kernel is very compute intensive, and the GPU (or Graphics Processing Unit) is substantially faster than the CPU at performing this operation. Figure 3.6 shows the output from the algorithm where the blue is the intensity of difference and the white points indicate values that are TRUE in the return value.

We currently run this algorithm in camera space because it is simpler to do it that way. It may make more sense to run it in projector space, or some third

space that is more linear in the areas most of the fiducials will be seen. Chapter 4 describes this in more detail.

GLSL SHADER(*Pixels*)

```

1  for  $p, x, y \in Pixels$ 
2      do
3           $t \leftarrow 0$ 
4          for  $lx, ly \in Neighbors$ 
5              do
6                   $dist \leftarrow \sqrt{lx^2 + ly^2}$ 
7                   $diff \leftarrow \text{GREYVALUE}[p] - \text{GREYVALUE}[\text{PIXELAT}[x + dx, y + dy]]$ 
8                   $t \leftarrow t + diff * (1.1 - dist)$ 
9           $t \leftarrow (t/100. + .7)$   $\triangleright$  Found through experimentation
10          $t \leftarrow (-.02/v + 1.23)t$   $\triangleright$  Boosts areas that are poorly lit
11         if  $t > 1.0$ 
12             then
13                  $hit \leftarrow \text{TRUE}$ 
14             else
15                  $hit \leftarrow \text{FALSE}$ 
16         save  $t, hit$ 

```

Figure 3.4: *GLSL Image Processing Pseudocode* - the input is pixels from the camera in RGB, the output is the processed image, as seen in Figure 3.6.



Figure 3.5: *Raw output from camera with calculated fiducial dots overlaid in green.*

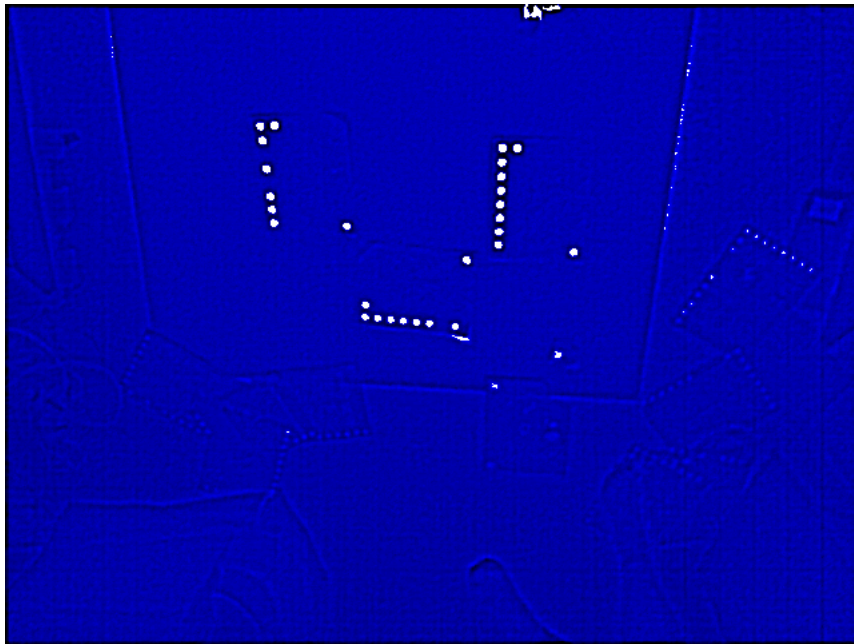


Figure 3.6: *Output from GLSL Shader. Note that the centers are not calculated yet.*

Dot and Fiducial Identification

Before fiducials can be located and oriented, the dots that make up the fiducials need to be identified. This is done by locating areas of the image that are marked by the GLSL shader, then mapping them as unique areas by filling those areas individually. Each of these filled areas is evaluated by quality, mostly depending on the number of pixels that constitutes it: the closer the sample is to the ideal number, the better quality it is judged to have. While the shape of the fiducial matters, we have found that simple pixel counting is sufficient for reasonable results. These dots are then stored into an array. The pseudocode for this can be found in Figure 3.7.

The next step, after the dots have been found, is to identify the fiducials. This is done by looking at every dot and searching for nearby dots where the two nearest form a 90 degree angle. If there is skewing because of a glancing angle to the camera, the angle should still be close enough for operation. This 90 degree angle becomes the corner and scale for our fiducial. Next, the algorithm looks for dots in a uniform pattern along each edge of the 'L.' If at any point the fiducial is found to be invalid because of an impossible pattern, it is thrown out, and the search continues. Even with our algorithms for fiducial identification, we may get false positives, and that is why we suggest the use of ECC/EDC for future iterations of the system. A rough overview of this algorithm can be seen in Figure 3.8.

If a fiducial candidate is found, the dots from both legs are converted into bits. These bits are examined, and error correction and detection could be performed at this point. However, currently, no error detection or correction is used. As many

FINDDOTSFORSOUP(*Pixels*)

```

1  id ← 0
2  for x, y ∈ Pixels
3  if ¬ISFILLED[x, y]
4    then
5      (centerx, centery, count) ← FILLAREA[x, y, id]
6      id ← id + 1
7      if count ∈ range
8        ADDDOTTO_SOUP[centerx, centery, count]

```

FILLAREA[] returns number of pixels filled and contiguous marks pixels as filled.

ISFILLED[] returns whether or not the cell has previously been filled.

ADDDOTTO_SOUP[] appends the triplet to an array of dots for later processing.

Figure 3.7: *Dot Identification Pseudocode*

as ten bits are available before detection or correction, and the system could be extended to find bits in other directions. The output of this pass is an array of fiducials along with the ID of that fiducial. These IDs do not change from frame to frame and are determined by the pattern of the fiducial, and so are assigned at the time the fiducial is printed.

3.4 Coordinate Mapping

Coordinate mapping is necessary to model where everything is. The coordinates produced by the fiducial algorithm are in camera space. This does not correspond to anything of great significance. In order to do anything interesting, the algorithm needs to be able to output light where these objects are. In order to know where to project the light, we must map the data from camera space to projector space. Once the coordinates are in projector space, light can be cast out into the scene at the appropriate locations. This section covers the processes needed for both producing and using this map.

Training

Many models such, as those discussed by Lee [12], use just four points along the parameter and one interior point, or variants with up to nine known points to train the mapping. These points for training are typically projected by a projector, and the user is expected to manually annotate the learning points which are projected.

We have chosen a more robust mechanism. There are two configurations: a regular configuration for operating mode, and a learning configuration for training. For the learning configuration, the IR Pass Filter needs to be removed from the camera so that the camera can see the light projected by the projector. Then, in a dark room, the projector projects a small ray of light at the surface on which the system will be used. The camera finds the brightest point. The system correlates the brightest spot in the camera to the brightest spot being projected. It proceeds

to do this for a relatively dense grid of points, roughly 32×16 , until the entire screen is mapped. By projecting a ray of lights from a known point on the output, we can now associate it to a known point on the input. The 32×16 grid produces a full mesh of all visible points in camera space and their corresponding points in projector space.

This mechanism is robust enough to automatically correct nonlinearities in the camera’s and projector’s optics by approximating them with a dense mesh, making complicated equations to determine closed-form solutions to these nonlinearities unnecessary. This also causes all issues with perspective correct mapping to be diminished since the warping effect is limited to very small areas, and we are not operating anywhere near the horizon [20]. While we have an affine grid which warps, we have a very fine affine grid, so the warping is limited.

The generation and rendering of this map is handled by OpenGL. By using this tool, we are able to get all of the benefits of OpenGL, such as built-in triangle/quad rendering, interpolation between training points, and the ability to easily store the output image.

Use of Map

With the complete mesh of points discovered during training, a mapping is possible. However, the mesh of points is difficult to use to transform from camera space to projector space. To alleviate this, we use OpenGL to render this grid as a mesh to a high-resolution, floating-point RGB texture, with vertexes in the

positions that the camera sees them and vertex colors dependent on the position that the projector projected them. This creates something that looks like an unusually shaped, nearly rectangular object with a solid color gradient. It is then stored as a lookup map. By storing the texture at a relatively high resolution, additional interpolation can be achieved. This method fully interpolates the mesh, using a linear interpolation that was computed ahead of time.

Whenever one wishes to locate the appropriate position to draw something, one need only look up the position in the map based on the location the camera sees the thing. Then, the contents of the map at that point give the coordinates in projector space, where features should be drawn. A summary of this can be seen in Figure 3.11.

Reverse Mapping

Many times, there will be features displayed on the projected surface, such as buttons, that need to find the camera data for the points under them. This makes it possible for buttons to be clicked, even if the system does not detect a fiducial at that location.

A reverse map is acquired by performing a very similar operation to the forward mapping. However, when rendering this lookup table, the positions of the points in the mesh correspond to the positions the projector is projecting, and the color corresponds to the location the camera saw the point. Using this lookup table, any point on the output image can be correlated to any other point on the input. By

using the inverse of the map found in Figure 3.10, we can render what the scene would look like in projector space in Figure 3.12. By using this reverse mapping in real time, we can verify the alignment of the system and see what the projector sees in infrared overlaid on the scene.

3.5 User Interface

User interfaces often contain windows that users can drag around by one means or another, and ours is no exception. The windows contain buttons users can click and feedback to the user in some way. We have chosen an alternative method of moving windows. Our windows have no virtual handles, they are controlled completely by the position of fiducials. This enables users to tangibly interact with the windows and move them around. Windows may also be scaled or rotated, traditional UI location rules will not suffice. Because Windows may overlap even if the sheets of paper are not, we must take focus into consideration. Because windows can be occluded by other windows, a user may bring a window to the foreground by moving the page associated with the window slightly. X, Y, rotation and scaling all need to be taken into consideration when rendering the window. The contents of the window needs to move with the window and contents' location within the window needs to be preserved. With our UI, we must not only move, but also rotate and scale the contents accordingly.

This means buttons on these windows will exist within a third coordinate system: window space. A transform is performed on the UI elements in order to

move them into global projector space. This transform can easily be handled by the OpenGL interface underlying our system. The only remaining concern is that some times objects need to be aware of where they are in global projector space, for things like buttons that will be interacting with a user. Because of this, the transform still must be able to be performed on the CPU.

Buttons and Objects

When a button is rendered, its projector-space coordinates are stored. This allows the reverse map to find the strength of a fiducial at its location by looking at the output of the GLSL shader in its location. The part of the image that the button cares about is the intensity of the output from the shader in camera space. This lets the buttons obtain operator interaction information from the operator by use of a fiduciary dot. Rather than use the 1-bit “hit” output, it uses the shader’s intensity output. By allowing this variable output, noise is less likely to cause spurious inputs.

Button clicking is done slowly, and button re-clicking needs to be timed as well. Because we get a variable amount of “click,” or how much the system believes a fiduciary dot to be in a particular location, we can allow the button to be three visual states: unclicked, clicking, or clicked. As a user attempts to click a button, by moving a fiducial dot underneath it, the button will slowly grow more and more red, until it reaches a threshold. At that point, the button will “click” and cause whatever user interface events are programmed to be executed. The button will then reset to black and wobble. This wobbling indicates to the user that the button

has been clicked. If the user wishes to click a button twice, he/she needs only leave a fiduciary dot at that location.

This system is not well suited for text input, and although this is possible, physical keyboards are recommended. The system does have a number of other possible control elements that we will not explore. For more information on these, see Chapter 4.

Drawing other high-level objects is fairly straightforward. Many entities could be placed as “full size” windows. One could overlay video inside one of these, or treat it as a texture, either static or rendered elsewhere. This would make it possible to project scenes inside these windows with which the user could still directly interact. Windows and other tools can come in a variety of styles; some may look like toolkits and may contain keyboards, while others may simply shadow some real-life paper.

3.6 PDF Search Application

Our current demonstration is a PDF searcher. We decided to make an application in which the user can interact with a PDF file. This includes lighting the sheets appropriately as well as enabling the user to search the PDF and highlight the results. This portion of the project provides a comprehensive demonstration of this system.

We decided to use a fixed PDF, then load the PDF into the program as a database so that the system is aware of what is on every page and where it appears. We used a third party tool to find the locations of all of the words in the PDFs.

This third party tool lacks accuracy which can be seen in our end product, but this inaccuracy is irrelevant to the task at hand. The example merely serves to illustrate the function of the system.

The PDF is printed onto a series of pages with fiducials written on them. We then select some of these pages for use in the system. Figure 3.14 shows what these pages look like when printed from the camera's infrared view. The system can now identify which pages the user has lying out in the camera's view based on the fiducials printed on them.

The system tracks these pages as they move. The system highlights the pages as they move so that the user can more easily see them. Pages that the user has not recently interacted with will fade into the background in order to avoid overloading the user with irrelevant information. The user can also search these documents using a regular keyboard. As the user types a term to search for, the system will find all occurrences of that word in the document. In the event that it finds a word on a page, it will highlight the page by brightening it and will also put yellow light on all instances of that word on the page. An example of this can be seen in Figure 3.13.

If a page is obscured such that the system cannot read the fiducial, as could happen when the user stacks all of the pages on top of one another, the application will just brighten the stack of pages by highlighting the first page. It will not use any yellow highlighting until the page that contains the words that it wishes to highlight is made visible to the camera.

```

BUILD FIDUCIAL LIST(DotSoup)

1  for dot  $\in$  DotSoup
2      do
3          (dotc1, dotc2)  $\leftarrow$  FINDTWO CLOSEST(dot)
4          (ddx1, ddy1)  $\leftarrow$  (dotc1.x - dot.x, dotc1.y - dot.y)  $\triangleright$  Direction 1
5          (ddx2, ddy2)  $\leftarrow$  (dotc2.x - dot.x, dotc2.y - dot.y)  $\triangleright$  Direction 2
6          angle1  $\leftarrow$  (atan2(ddx1, ddy1) +  $\tau$ ) mod  $\tau$ 
7          angle2  $\leftarrow$  (atan2(ddx2, ddy2) +  $\tau$ ) mod  $\tau$ 
8          if angle1 - angle2  $\simeq \tau/4$ 
9              then  $\triangleright$  Which dot is in the “down” local direction?
10                 dangle  $\leftarrow$  angle1
11          elseif if angle2 - angle1  $\simeq \tau/4$ 
12              then  $\triangleright$  “down” is in the cw direction from right.
13                 dangle  $\leftarrow$  angle2
14          else
15              continue
16          ADD FIDUCIAL(COLLECTBITS(dot, dangle))

```

COLLECTBITS[] is defined in another figure.

FINDTWO CLOSEST[] finds the two closest dots to *dot*

For convenience we set $\tau = 2\pi$.

Figure 3.8: *Fiducial Finding Pseudocode.*

```

COLLECTBITS(dot, dangle)

1  ddot  $\leftarrow$  FINDCLOSESTDOT(dot + (cos(dangle), sin(dangle)) * estDist)

2  newEstDist  $\leftarrow$  | ddot - dot |

3  outputNumber  $\leftarrow$  0

4  bitNumber  $\leftarrow$  0

5  for expected  $\in$  DownBits

6      do

7          estPos  $\leftarrow$  ddot + (cos(dangle), sin(dangle)) * newEstDist

8          opos  $\leftarrow$  FINDCLOSESTDOT(estPos)

9          if | opos - estPos | < .3 newEstDist ▷ Is dot here?

10             then

11                 outputNumber  $\leftarrow$  outputNumber  $\vee$  (1  $\ll$  bitNumber)

12                 ddot  $\leftarrow$  opos ▷ New dot is starting point

13             else

14                 ddot  $\leftarrow$  estPos ▷ Guessed dot starts

15             bitNumber  $\leftarrow$  bitNumber + 1

16             ▷ Run similar for RightBits if used.

17  return outputNumber;

```

FINDCLOSESTDOT[] finds the closest dot to the input location.

Figure 3.9: *CollectBits* function. *Dot* is the location of the corner dot, and *dangle* is the down angle.

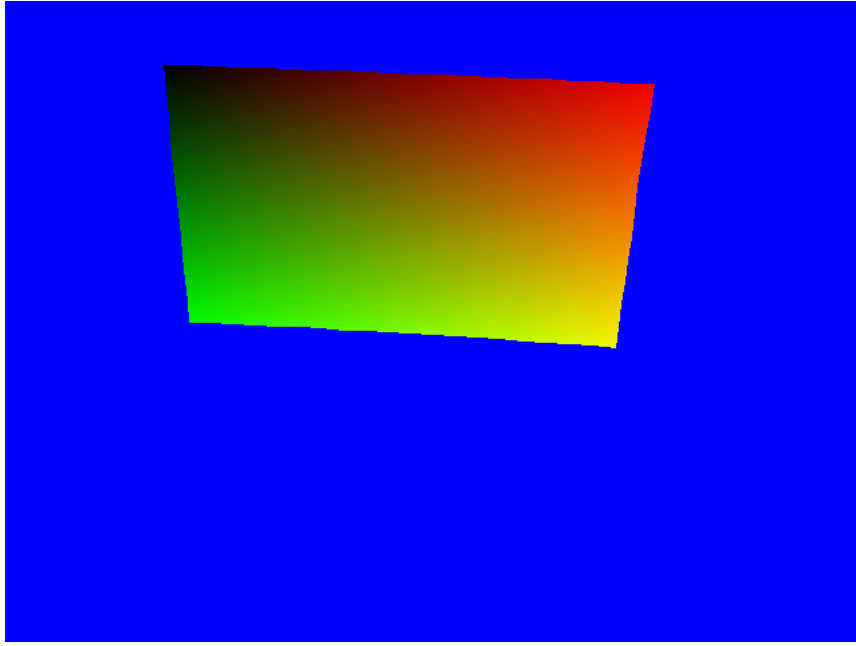


Figure 3.10: *Forward mapping map. The quantity of the red component corresponds to the x projector space coordinate and the quantity of the green component corresponds to the y projector space coordinate.*

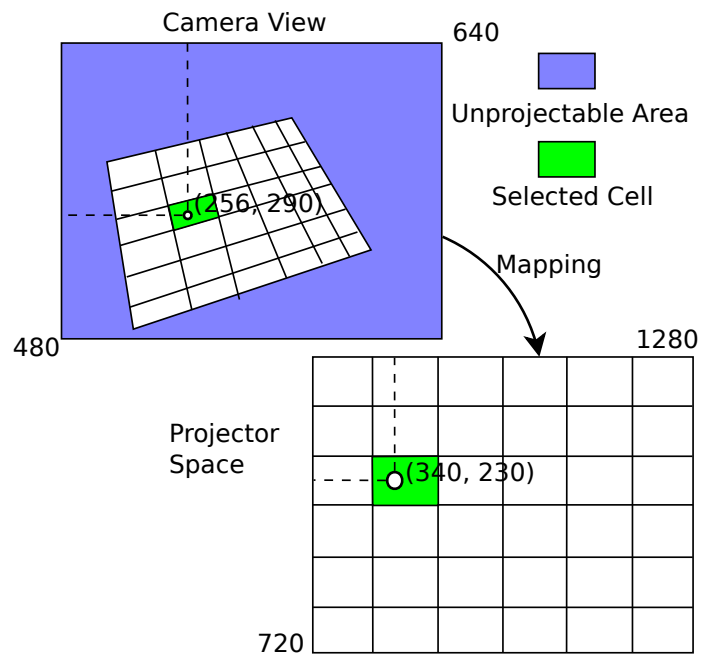


Figure 3.11: *Mapping example. This example maps a marker at $(256, 290)$ in camera space to $(340, 230)$ in projector space by looking up the (R, G) component in the forward map.*

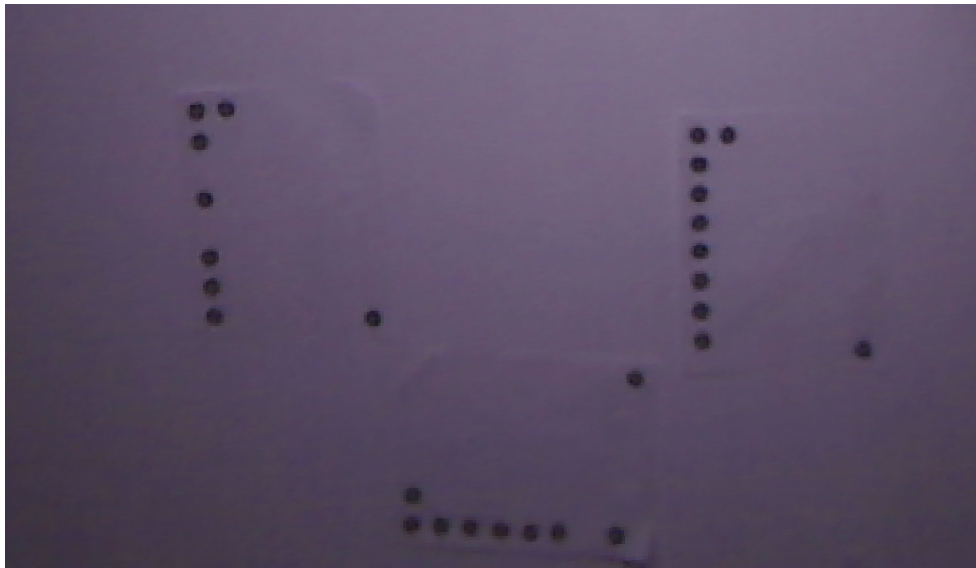


Figure 3.12: *Remapped scene into projector space*



Figure 3.13: *Projector output for PDF tool. The darker pages are pages that have not been moved in a while; the lighter pages are ones that have been moved recently. The yellow areas are where the word “results” appears in the paper. Note that the word “results” is actually being projected onto the surface.*

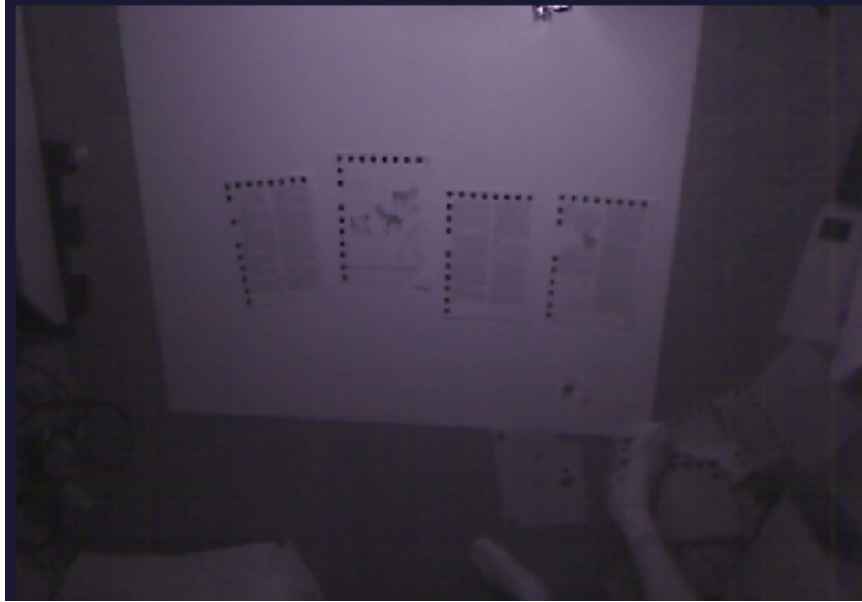


Figure 3.14: *Camera input for PDF tool. This shows the scene above as the camera sees it. While images are being projected onto the surface, the camera only sees the scene without the additional lighting.*

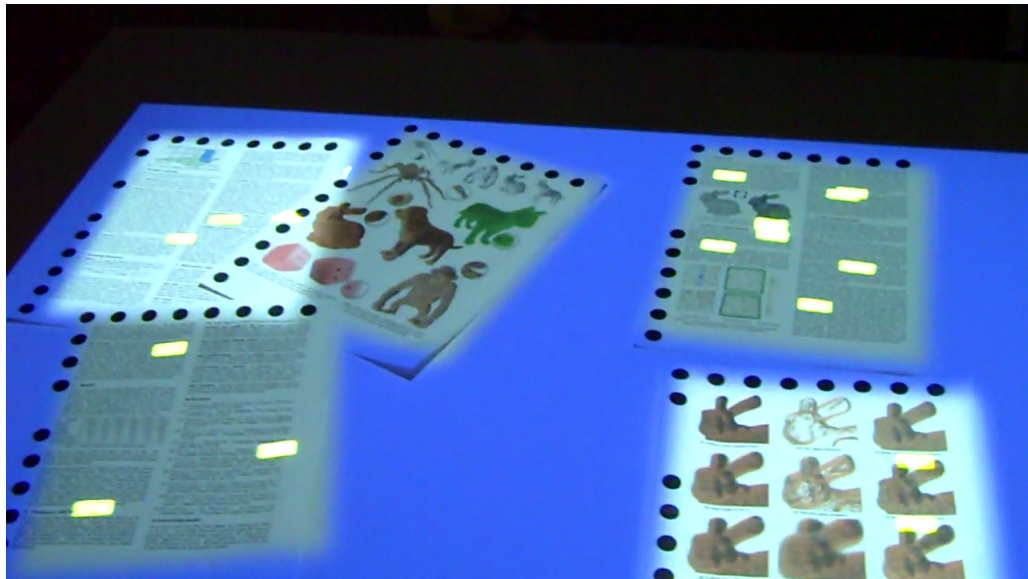


Figure 3.15: *Visible light picture of PDF tool. This is what the user sees - the paper, and real scene along with the projected overlay.*

Chapter 4

Future Work

There are a number of topics that came up throughout the research, development and discussion of this system that I was unable to address during its production. They are covered in this chapter.

LED-Camera Synchronized Flashing

In some cases, lighting used to light a scene can be synchronized to the camera's shutter. This is common with still cameras, i.e. their flash bulb. This makes it possible to catch scenes that are in motion without having them blur. There are some video systems that do the same thing; they flash 30-60 times per second in sync with the shutter. These systems are not very common now, however if we could implement one I believe it could improve quality of the images captured by the camera significantly.

Currently, the exposure time on the camera is turned down as low as possible to reduce the blur caused by fiducials moving quickly, as when a user moves a sheet of paper. This has negative impacts such as the need to increase gain, which adds noise to the picture. If we were to flash the IR LEDs at a substantially higher power, but only for a few milliseconds, we could get a snapshot of the scene with no blur and not overload the IR LEDs. This could greatly improve our tracking capabilities

while fiducials are in motion.

Infrared Ink

Currently, we use standard black ink to print fiducials. This is because the black ink from a printer absorbs infrared light, creating a black dot that our algorithms can lock onto. While ideal for tracking purposes, it is not aesthetically pleasing because we end up with black dots scattered around.

Ideally, we should be able to print the markers using ink that is primarily absorptive to infrared but is transparent to visible light. Additionally, we could print the material intended to be visible to the user onto papers using ink that is absorptive in specific visual ranges, but transparent in the infrared range. This would produce less obtrusive fiducials and make the system more useful for a variety of printed media.

Fiducial Error Detection and Correction

As mentioned in Section 3.3, it would be helpful for the fiducial markers to have error detection and correction built into them. This would enable a much more robust location mechanism. It would help prevent loss of tracking during motion or poor lighting, and it would help prevent false positive and false negative fiducial detection.

Very few modifications would need to be made to the system since fiducials are already tracked by ID. The IDs would just need to be run through the EDC or

ECC algorithms to ensure that detected fiducials are in fact real, and in the event there was a problem reading a few bits, they can be corrected.

More Extensive Use of Colors

Itten [17] discusses many interesting properties of the psychological and even physiological impacts of color. We would like to apply these interesting properties to our system. It could provide interesting color effects and help in our goal of making the users more comfortable, especially if the system is ever used for room lighting.

If we make these alterations, it would also make sense to add additional inputs to the system, such as actual room temperature and position of people within the room. There is a great deal of context in color as discussed by Itten and others that is not currently considered by the system.

Chromaticity Shifts

When the eye's retina is exposed to surrounding light of different types, its perception of color shifts [26]. There has been work by many different people trying to understand this effect. While many of these experiments come up with different results in terms of the actual shifts, it is clear that there is a shift. It would be interesting to enable our system to take this into account to make images have higher fidelity than they could if they were just displayed with regular vivid colors.

The system could analyze the room's light using another sensor to take into consideration outside light, other objects in the room and anything else that could

alter the light content of the room. By taking this into consideration, the system could simulate how the eye would perceive what is being projected, then adjust that accordingly to allow the user to see the picture the way it was intended.

Radiometric Compression

Radiometric compression is described by sARc in the discussion of their system [19]. They consider the angle of the light being emitted from the projector to the surface, and they alter the brightness being projected in different zones accordingly. This is important because surfaces at glancing angles or at distances need to be lit more strongly than surfaces perpendicular to the projector in order to achieve a uniform apparent brightness across the entire surface. This would likely be easy to implement in our case because we could take radiometric readings during the calibration phase.

Operate in Other Space

Currently, our fiducial recognition takes place in camera space which greatly simplifies many of our problems since there is a direct mapping to screen space already established. However, this does not work as well if the camera is at a glancing angle. It may make sense to perform the dot recognition GLSL and fiducial finding algorithms in another space, possibly projector space.

Kinect

The Kinect is a unique user input device for the Microsoft Xbox 360[®]. It is a camera which can provide the depth of everything in its view as well as the color. While our system works particularly well for fiducials and detecting dark spots, we cannot effectively interpret human gestures or “pointing.” We feel that the Kinect could open up a wide variety of new and interesting mechanisms of input. Because the Kinect can acquire depth information, it would be possible to identify objects like someone’s hand without any markers. Additionally, the Kinect could use our mapping technique, and perform the calibration using the same technique described earlier.

Because people’s hands could be located by the Kinect, a user could simply point to what they want, and the system could identify it and “click” the appropriate button rather than manipulating a token.

Other UI Primitives

We currently only implement buttons as a form of input. Because we can have many fiducials on the screen at once, we could easily implement persistent sliders, audio mixers, x/y maps, color pickers, or other interesting types of input that demand variable values from the user. I feel that a good example of this would be a system that takes several values the user could set. Several bars corresponding to these values could be presented to the user and the user would place a fiduciary dot on each. The user could alter multiple values in real time, and have very quick

control over the values. This would provide a more interesting system allowing new complicated input forms, and a genuinely unique user interface.

White Board Reading

Another interesting possibility would be to use a white board as a valid form of input. Instead of using fiduciary dots printed onto pieces of paper, users could simply draw onto the white board whatever they wished to input to a computer. This raises a number of challenges that are not yet resolved in this paper. One major problem is that our system is geared entirely to finding fiduciary dots and not lines and other shapes. Another is that white boards do not provide the same sort of surface that we are used to working on, however, the advantages would be that this could provide both a very large touchscreen-like interface as well as everything else the system provides.

A sample use of this form of input could be a video game, such as a tower-defense. Tower defense games are a popular video game genre in which a user must form a maze using automated turrets for streams of several computer-driven opponents to move through. The user does this to make it as difficult as possible for the opponents to reach one side from the other. This would allow users to actually draw the map and place the towers. This would give users free-form control of the map and provide a much more fluid interface for designing the maze. Currently, most tower defenses games only permit mazes made up of discrete blocks instead of continuous areas.

We were unable to find an implementation of a tower defense like to this proposed system. Therefore, much of the technology needed to develop such an extension need to be developed from scratch. This could take a great deal of time to accomplish.

Chapter 5

Conclusions

Semantic Light is a topic that is still in its infancy. We have implemented a usable Semantic Light system. We have shown that it has relevance and substantial potential, and we have only begun to scratch the surface. We hope to inspire interest and curiosity in the future of this topic. While our implementation may be left by the wayside, our hopes are that portions of this project may be part of future work in the area. We have established a conceptual foundation and concrete example for further work on the topic. While we have shown a few possible uses for Semantic Light and most of our work has been on the toolkit side, we hope that this does not inhibit the reader's imagination to the plethora of possible applications for this system. Semantic Light provides a new and unique method for user-computer interaction and introduces potential for future research.

In review, these following main points have been made: We flood the scene with infrared light and use it as our primary mode of input to a computer. This prevents contamination of the input from the output lighting from the projector or other light source. We can do this by filtering out visible light and by using a camera that sees infrared light. By using algorithms to process this image, we can discover a great deal of context and information about the scene to the computer.

We have provided a mechanism by which the working surface can be mapped

so that points seen by the camera can be correlated to locations where the projector must project to hit those points. This process can map areas in a detailed manner and automatically takes care of non-linearities. It is robust and very fast, making it suitable for use driving a user interface.

We have developed a system that can find fiducial dots in a scene and compile them into a map of the locations of fiduciary markers. These markers have position, rotation, scaling, and ID information calculated for them. This can be used to fuel a user interface or obtain other information from the users of such a system. It is possible to make a user interface where the “windows” can be controlled using sheets of paper, and they do not need to remain aligned straight up and down during their use and may instead be rotated. This provides an intuitive and interesting interface for the user.

Lastly, we have shown that given basic location information and a goal, the system can present light in a useful and aesthetically pleasing manner. This has been shown with a program that lights pages of a PDF document, and adds highlighting on specific locations. This program is both responsive and relatively accurate and shows a potential real-world application for Semantic Light.

Semantic Light is an interesting field that we hope will take root and be developed further. It is a fecund field which has endless possibilities. We hope we have sparked your imagination and inspired you to take interest in this area of research.

Appendix A

Bill of Materials

Below is a list of materials used when constructing the system. This list is complete to the best of my knowledge, however there were things used that were lying around and may have been accidentally excluded. Also, this is the rough estimated cost for my implementation. This does not take into account suggested changes that should be made.

Part / Description	Notes	Approximate Cost
5x 1x6x6' Plank Wood	For frame	\$25
20x #6x1 Wood Screws	For frame	\$2
1x 2'x4' Plywood Sheet	For frame & mounting	\$8
1x 2'x2'x3/8" ABS Plastic Sheet	Projector mounting	\$18
3x M3x18 Screws	Projector mounting	\$1
Power Strip		\$4
IEC Power Cable	For projector	\$4
Mini USB 2.0 High Speed Hub		\$10
15' HDMI Cable	Computer connection	\$20
QuickCam Pro 9000	Camera interface	\$70
Mineral Oil	IR Modification	\$5
Benchtop Power Supply	For IR LEDs	\$60
2 3/4" x 6' x 1/8 Aluminum Strip	For IR LEDs	\$19
8x3W 850nm IR Power LEDs	For IR LEDs	\$35
4x 20hm x5W Resistor	For IR LEDs	\$5
790 nm IR Pass Filter	For camera	\$30
Optoma HD 20 Projector		\$900
	Approximate cost	\$1186

Appendix B

General Notes

Camera IR Filter Removal

On the QuickCam Pro 9000[®], there is an IR filter on the inside of the Carl Zeiss[®] optics. You must remove the outer casing on the camera, then use a soldering iron to carefully melt away and pull off the connection to the optics focusing device. Once this is done, the optics can be removed from the main camera PCB. Take care to never make contact with the main CCD, and if dust ends up on it attempt to blow it off from a distance as not to get any water particles on the CCD surface.

With the optics removed, on the inside, there is a small circular disc. The back (CCD side) coating is IR-block. You should coat this in mineral oil, use a toothpick to spread it around. Once it is coated, you can scrape the edge of it using an exacto knife. Scrape until you cut all the way through the coating. You will find that it easily chips off using a sharp blade. Care should be taken to minimize the damage to the actual glass surface when doing this. The more damage done to the glass surface, the less contrast will be available on the final product. Feel free to use a q-tip to remove the coating flakes and add more mineral oil during this process.

Once you feel satisfied that the coating is off and most of the flakes are removed, you can use clean q-tips and rubbing alcohol to remove the remainder of the mineral oil from the surface of the optics. When you have completely removed all flakes and

mineral oil, you can reassemble the camera.

An additional note is that it is possible to remove the plate entirely, however it is very difficult to mount a new plate in its place. By removing the plate, the properties of the optics are changed and you must either cut the base of the optics slightly, or add additional glass in place of the glass with the filter on it that was removed.

Quality of Fiducial Finding

We use the QuickCam Pro 9000® at 640x480 resolution. We also have not found a way to remove the IR filter and remove glare, our process for removing the filter degrades the quality of the optics. We also mount the camera far away from the projected area, thus individual features will appear smaller. In addition to that, because we are not flashing the LEDs, motion detection is not great. All of these considered, our fiducials must be fairly large for the system to see the clearly. We were only able to fit 18 dots in our L configuration. Even these are slightly smaller than we would like. If some of the drawbacks were fixed, the detail could be increased considerably, shrinking the Fiducials even further and improving tracking quality.

List of Abbreviations and Glossary

LED	Light Emitting Diode - Device used to create light when voltage is applied. It emits only a primary wavelength of light. See 1.2 for more information.
DLP	Digital Light Processing - Process used in many TVs and projectors to create projected or presented images. See 1.4 for more information
IR	Infrared Light - Light that is outside of the visible spectrum but can be seen by modified cameras. In the scope of this paper, we mean in the range of 720 to 940nm Visible Light - Light making up the spectrum of what we see. In this paper, we are referring to light that can be represented by red, green and blue, and in the range of roughly 350nm to 700nm.
°K	Color Temperature / Degrees Kelvin Black Body Radiation - White light is typically classified by color temperature, or the temperature of a black body radiator. See Section ?? for more information.
GLSL	OpenGL Shading Language - A high performance programming language that can be used for general purpose computation, geared toward graphics. Fiducial - Marking that is easily discernible by a machine vision system. For our system, it is a large black dot.
lm	Lumen - Measure of total light output as it is seen by the human eye. It is defined by the light of one candela in a one steradian cone. Infrared and ultraviolet light does not contribute to total lumens because they are not visible to the human eye.
lm/W	Lumens per Watt - Measure of efficiency of a light source - the lumen is a measure of brightness, and a watt is a measure of power.

Bibliography

- [1] Taub, Eric A. LED Bulbs for the Home Near the Marketplace <http://www.nytimes.com/2010/05/17/technology/17bulb.html> Retrieved 2010-11-25
- [2] Davidson, Max European Union ban on lightbulbs leads to a dim future <http://www.telegraph.co.uk/earth/greenerliving/6083119/European-Union-ban-on-lightbulbs-leads-to-a-dim-future.html> Retrieved 2011-02-12
- [3] Keefe, T. J. The Nature of Light <http://www.ccri.edu/physics/keefe/light.htm> Retrieved 2011-02-12.
- [4] N.C. Division of Pollution Prevention and Environmental Assistance Energy Efficiency in Industrial Lighting DPPEA-FY03-09 Bulletin.
- [5] CREE Datasheet Cree XLamp XP-G LEDs Product Family Datasheet CLD-DS20 REV 4
- [6] Elektor Magazine DIY LED Projector Issue 375, March 2008
- [7] Sears, Andrew; Shneiderman, Ben 1991 High precision touchscreens: design strategies and comparisons with a mouse *International Journal of Man-Machine Studies*, vol 34 no 4 pp 593 - 613.
- [8] Andersson, R. L.; Gibbon, D. C.; Lyons; R. P. AT&T Labs-Research 1999 BULLSEYE: A Compact, Camera-Based, Human-Machine Interface Presence, Vol. 8, No. 1, February 1999, pp. 65-85.
- [9] Kato, Hirokazu; Billingham, Mark 1999 Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality* pp. 85-94 October 20-21, 1999 San Francisco.
- [10] Cho, Youngkwan; Park, Jun; Neumann, Ulrich Fast Color Fiducial Detection and Dynamic Workspace Extension in Video See-through Self-Tracking Augmented Reality Unknown Publication.
- [11] Han, J. Y. 2005 Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*

- [12] Lee, Johnny Chung. 2007 Low-Cost Multi-point Interactive Whiteboards Using the Wiimote <http://johnnylee.net/projects/wii/> Retrieved 2011-02-20.
- [13] Bencina, Ross; Kaltenbrunner, Martin, 2005 The Design and Evolution of Fiducials for the reacTIVision System Music Technology Group, Audiovisual Institute; Universitat Pompeu Fabra, Barcelona, Spain
- [14] Costanza, E. and Robinson, J., 2003 A Region Adjacency Tree Approach to the Detection and Design of Fiducials Department of Electronics, University of York, York, United Kingdom
- [15] Smith, Thomas; Guild, John, 1931-32 The C.I.E. colorimetric standards and their use *Transactions of the Optical Society* 33(3): 73-134
- [16] G. Kirchhoff (1860). On the relation between the Radiating and Absorbing Powers of different Bodies for Light and Heat, *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science* Fourth Series, July 1860. pp 1-21
- [17] Itten, Johannes; Birren, Faber (1970). The Elements of Color: A Treatise on the Color System of Johannes Itten Based on His Book The Art of Color *New York: Van Nostrand Reinhold* ISBN 0-442-24038-4 / pp. 45-49
- [18] NuFormer 3D Projections on Buildings <http://www.projectiononbuildings.com/> Retrieved 2011-03-15
- [19] Bimber, O., Wetzstein, G., Emmerling, A., and Nitschke, C. (2005) Enabling View-Dependent Stereoscopic Projection in Real Environments In proceedings of *International Symposium on Mixed and Augmented Reality (ISMAR'05)* pp. 14-23
- [20] Heckbert, Paul S.; Moreton, Henry P. Interpolation for Polygon Texture Mapping and Shading *State of the Art in Computer Graphics: Visualization and Modeling* New York, 1991. pp. 101-111
- [21] Diffin, Bernard Dispelling the “Inefficient Neon” Myth EGL NEON NEWS, Spring 2006.
- [22] Texas Instruments DLP Technology How DLP Technology Works <http://www.dlp.com/technology/how-dlp-works/default.aspx> 2011-04-18.
- [23] Frey, William; Zyda, Michael; McGhee, Robert; Cockayne, Bill Off-the-Shelf, Real-Time, Human Body Motion Capture for Synthetic Environments, Com-

puter Science Department Naval Postgraduate School, Monterey, CA 93943-5118, USA 1995

- [24] Brown, Michael S. Projected Imagery in your “Office of the Future” Computer Graphics and Applications, IEEE July-August 2000 pp 62-67
- [25] Stiles, W. S.; Wyszecki, Gunther Color Science Wiley Classics Library Edition Published 2000 pp. 117-135.
- [26] Stiles, W. S.; Wyszecki, Gunther Color Science Wiley Classics Library Edition Published 2000 pp. 429-440.