System for Automated Interactive Lighting (SAIL)

Joseph Zupko
College of IST, Penn State University
321D IST Building
University Park, PA 16803
814-863-2480

jaz147@psu.edu

Magy Seif El-Nasr
School of IAT, Simon Fraser University
250-13450 102nd Avenue
Surrey, BC Canada V3T 0A3
778-782-8180

magy@sfu.ca

ABSTRACT

Successful lighting in video games is more than a physically accurate illumination model. Aesthetics and function are of equal or greater importance. Lighting designers may deviate from physical accuracy to help a player identify an important object or to more powerfully evoke a desired emotion. Under the assumption that fulfilling the pipeline needs of interactive lighting design requires more than solving the computer rendering equation, we set out to develop a *System for Automated Interactive Lighting (SAIL)*. The goal for SAIL was to develop an adaptive system that maintains lighting design goals (aesthetic and functional) in the context of unpredictable, interactive experiences. This paper presents SAIL and the results of a qualitative evaluation of SAIL's contributions. We describe the algorithms of SAIL, where it succeeds, and where it fails. We conclude with a plan for future work.

Categories and Subject Descriptors

I.3.7 [**Three-Dimensional Graphics and Realism**]: Color, shading, shadowing, and texture.

General Terms

Algorithms, Human Factors.

Keywords

Lighting, Interaction, Content Creation Tools, Expert Knowledge.

1. INTRODUCTION

Lighting shapes all visual perception. Film, theatre, and many other design disciplines devote much energy to controlling light. In some cases the motivation is simple visibility. Effort is expended to achieve a minimum intensity to capture visual details. But lighting design is more than just visibility. Lighting can control mood, evoke emotion, and guide visual interest and intent.

Lighting for games is as important as for the disciplines just mentioned. Game lighting techniques have therefore adopted many of the practices of theatre, film, and architecture [1, 2] to achieve similar levels of quality. But games face a unique problem created by interactivity. Lighting designers often carefully preplan and configure lighting given omniscient knowledge of a space and its composition [1]. The light cast on a lead actor's face, for example, will be in consideration of the actor's stage position, posture, and costume as well as the storytelling goals and environment. Unfortunately, game lighting designers do not necessarily have this information. A careful lighting setup at the entrance of a foyer may be completely undone if a player decides to enter through a dining room window.

To address this problem, we propose a *System for Automated Interactive Lighting (SAIL)*. The goal of SAIL is to adapt to changes in context given a designer's aesthetic and functional intent, achieving the best possible compromise between what the lighting should naturally look like given context and what the lighting should ideally look like given a designer's goal.

This system is similar in many ways to the *Expressive Lighting Engine (ELE)* [3], a previous work with a similar purpose, with two key differences. First, ELE used numerical constraints to specify "expressive" terms, such as *visual tension* and *visual focus*, which it used as constraints of its lighting configurations. SAIL uses image analysis to extract lighting cues from an image. The image is used to represent a lighting goal in a somewhat abstract manner (the image can be of anything, such as an illuminated sphere). Second, ELE did not understand how an object responds to lighting. To ELE, the trunk of a tree was equivalent to a human of the same size and girth. SAIL uses an image analysis algorithm in a preprocess step to learn how a 3D object responds to light. It then uses this model to adapt lighting based on the object's appearance.

In this paper, we briefly describe lighting design theory and related work to provide context for SAIL. We then discuss the algorithms of SAIL and finally present results from a qualitative study we conducted to evaluate SAIL's contributions.

Our results reveal two key concepts that form the contributions of SAIL and this paper. First, images can be used by lighting designers to represent desired lighting effect, specifically, contrast and direction. Second, SAIL's concept to adapt lighting is sound, but it may reflect the natural illumination environment too well. At runtime, SAIL tries to reflect a natural illumination environment while applying a desired illumination goal to an

object. However, it does this independent of whether the environment is lit "well" or not. As a result, SAIL can produce lighting on an object that is consistent with the object's environment but is unsatisfactory to a lighting designer. We discuss the contributions of SAIL and this potential shortcoming further in Section 6.

2. TRADITIONAL LIGHTING

Adaptive lighting design is primarily targeted at interactive contexts such as video games. However, the fundamentals of game lighting come from more traditional domains such as film and theatre. In these domains, the role of a lighting designer is to balance several goals, such as visibility, depth, modeling, and to support the story and evoke mood and emotion [4].

In creating a lighting design, a designer considers the environment, the characters (or other objects of interest), and the narrative. She also considers time of day, the style of the work, and the mood or theme [1, 5]. A designer must balance her aesthetic and functional goals with an understanding and respect of the physical nature of light. Understanding the real world behavior of light attenuation, light reflection, occlusion, and other phenomena allows a designer to understand when and how she can "cheat" these phenomena to accomplish her visual goals.

An exhaustive review of lighting techniques is beyond the scope of this paper. We present here a common technique, used by SAIL that appears in both theatre and film. Known as three-point lighting, this method divides character lighting into three lights or sets of light: a key light, a fill light, and a back light [5, 6]. The key light provides dominant direction cues, the fill reduces visual contrast, and the back light isolates the object, punctuating its silhouette. This lighting method allows a designer to deliberately control lighting while still reflecting physical reality. The fill light, for example, takes the place of bounce illumination through careful control of lighting instruments and exposure, allowing a designer to reflect the physical property of indirect illumination while maintaining desired control of an object's appearance.

3. GAME LIGHTING

The lighting design process of games is chaotic, tied deeply to the rapid change of technology. Until recently, most video game lighting was pre-calculated (according to an interview in [7]). Often, an algorithm such as radiosity [2] was used and the radiance[1] of a surface was "baked" into the surface using light maps, which store the view independent radiance of the surface.

More recently, game lighting has split into two distinct approaches (also according to an interview presented in [7]). One approach is fully dynamic lighting. This lighting is based on mathematically simple light primitives and is calculated at runtime. Real-time rendering typically allows for only a few visible dynamic lights per object (8-10) but deferred rendering technologies [8] allow for many visible dynamic lights (40-50). However, due to tradeoffs with a deferred approach [9], non-deferred rendering is still often used, so it is not safe to assume

that more than a few dynamic lights can be used per object in real-time rendering. SAIL uses only two lights per object.

The second divergent lighting approach in modern real-time rendering engines is static lighting with spherical harmonics [10]. Spherical harmonics allow irradiance[2] to be stored at points in space and efficiently applied at runtime. Samples are commonly stored as either a surface aligned texture similar to light maps or in coarsely spaced grids [11]. Although spherical harmonics require more storage and processing than light maps, they offer the capacity to produce effects that light maps cannot, such as bump mapping [12] and low frequency specularity [11]. Spherical harmonics are mostly static but can be rotated [10].

SAIL uses two dynamic lights per object, which should fit within non-deferred rendering engines. Dynamic object lighting is necessary for SAIL to adapt lighting at runtime. However, it is compatible with a statically lit environment (spherical harmonics or light maps) so long as the environment can be sampled for approximate irradiance (using the Ambient Cube [2] of Half-Life 2 for example) to determine natural illumination.

4. RELATED WORK

The field of SAIL is inverse lighting. Here we present a brief overview of this field. For a more complete survey, see [13].

We divide inverse lighting into three subfields. Architectural inverse lighting targets visualization of real world spaces. Perceptual inverse lighting targets the automatic illumination of objects to a perceptual ideal. Interactive inverse lighting is the direct subfield of SAIL and targets interactive, real-time lighting.

4.1 Architectural Inverse Lighting

Figure 1. An architectural space automatically lit by [14].

The premise of most architectural inverse lighting work is the elimination of design by iteration. Traditional lighting design involves an iterative process of configuring lights, rendering a result, evaluating the result, and then repeating until a desired goal is met. Inverse architectural lighting tries to eliminate this process by allowing a designer to code a goal such that a software system can find the light parameters needed to achieve that goal.

One of the first works in this area was [14] (see Figure 1). This system found angles and intensities for lights given: 1) light positions manually specified by a user, and 2) numerically

---

[1] Radiance is the light emitted by a surface.

[2] Irradiance is light incident to a surface.

defined goals for the lighting design. Some of these goals were literal, such as goal irradiance to a surface while others were subjective, such as "spaciousness" and were derived from [15].

[16] used a similar approach to [14] and is the most recent and arguably most complete approach to inverse architectural lighting design. The work of [16] made only three assumptions about an environment: 1) surface reflectance can be described with symmetrical bidirectional reflectance distribution functions (BRDFs)[3], 2) the environment has no participating media[4], and 3) the rendering model used for visualization obeys the photon nature of light.

Within these assumptions, this system allowed designers to specify illuminations goals using scripts. For example, a designer might code a rule that constrained irradiance to a surface within a specific range. The system would then find light sources that fulfilled this and all other rules for a given space.

Inverse architectural lighting appears to be "solved". The work of [16] in particular produced a general-purpose and flexible automated system for designing architectural spaces. This body of work would probably be useful to games for the design of "baked" lighting on in-game architecture. However, the algorithms of inverse architectural lighting are too computationally expensive for real-time rendering and are effectively disjoint from the work of SAIL.

4.2 Perceptual Inverse Lighting

Perceptual inverse lighting [17-20] tries to find the "best" solution to light an object, where "best" is defined using psychophysical theories and the goal of maximizing shape perception and detail. Objects are typically illuminated without user interaction. Work in this field is targeted at practical applications where lighting is a time-consuming necessity rather than an integral and desired process (for example, in scientific visualization).
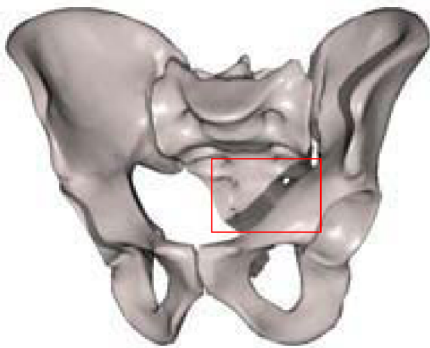

Figure 2. The system of [18] has manipulated lighting to allow the indicated cast shadow to act as a depth cue.

The system of [19] found light source settings to maximize perceptual features such as shape, detail, and depth perception.

[17, 20] targeted similar goals with different variations of a "light entropy"[5] metric based on [21].

[18] is the most recent and complete work in this area. This work approached the problem differently from prior work. Objects were classified in the geometric domain into areas of local curvature. Illumination was then applied in a discontinuous fashion to each area to maximize goals per area. Cast shadows (see Figure 2) were also explicitly considered, the only work in the field to do so.

SAIL is explicitly looking at interactive lighting, while the work of this field did not. However, similar to previous work, SAIL uses image analysis techniques. Specifically, SAIL uses the light entropy metric of [17]. Unlike most of the work in this area, correlated geometry is not used. SAIL only needs an image to specify a goal or understand an object's appearance. Further, because analysis of images at runtime is too slow for real-time rendering; SAIL conducts analysis in a preprocess step. It stores an object's response to lighting as a mathematical model that is used at runtime.

4.3 Interactive Inverse Lighting

The specific subfield of SAIL consists of only two known works, LightKit [22] and the Expressive Lighting Engine [3]. LightKit was a tool designed to interactively light medical models. It used a three-point lighting model similar to SAIL. However, unlike SAIL, it did not adapt to interactivity but simplified the iterative lighting process of traditional lighting design for non-experts.

The Expressive Lighting Engine (ELE) [3] is our prior work. ELE was a dynamic lighting system similar to SAIL but with a focus on "expressive" lighting. ELE lit both characters and environments by applying theatre and film principles. In particular, environments were divided into lighting "zones" (overlapping circles) and characters were lit using a three-point lighting model. The goals of ELE were specified using numeric constraints of subjective illumination, such as tension.

The design of SAIL was motivated by the shortcomings of ELE. In particular, ELE was difficult to use due to the fact that the numerical constraints supplied were often interdependent in hard to understand ways. ELE also did not consider the appearance of an object.

5. SYSTEM FOR AUTOMATED INTERACTIVE LIGHTING (SAIL)

The System for Automated Interactive Lighting (SAIL) is an intelligent automated lighting system (see Figure 3). Its design was based on an assumption that the appearance of an object is important when deciding how to light that object.

---

[3] A BRDF is a function of how a surface reflects light.

[4] Participating media are small particles that absorb, emit, or scatter light. An example of participating media is fog.

[5] Light entropy is based on information entropy and is the quantification of the potential information of an image.
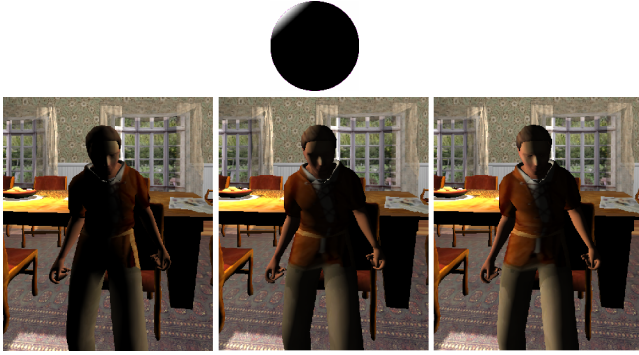
Figure 3. This figure illustrates SAIL. The shaded circle is the goal image. The left image is the man naturally lit. The center image is the man lit to achieve the goal image. The right image is the man lit as a compromise between the two.

SAIL consists of two main components. The first component is image analysis. SAIL uses image processing to understand a goal image. The goal image shown in Figure 3 is of a shaded circle, but the goal image can be of anything with shading and color contrast information. The second component of SAIL is object understanding used to achieve adaptive runtime lighting. This component is very similar to the object lighting component of ELE with the addition of image analysis to understand how an object responds to changes in illumination. This component uses metrics extracted from a goal image and the position of light sources in the environment. It then lights an object, using a three-point lighting model, to achieve an appearance between how the object "should" look based on the image and how it would look naturally based on the light sources. The current implementation of SAIL is shown in Figure 4.

Figure 4. The current implementation of SAIL.

5.1 Image Analysis

SAIL extracts two types of information from an image: 1) light direction, and 2) light contrast. Light direction roughly correlates to the direction of key light and light contrast roughly correlates to the intensity of fill light, but both are a combination of light direction and intensity and their effect on the object of an image.

5.1.1 Direction

Before further processing, each color channel of an image is filtered with a Gaussian kernel to remove high frequency information. This filtered image is then transformed into a grayscale image using the *Y* value from the *sRGB* derived *XYZ* color space as in [17]:

$$Y = 0.2126R' + 0.7152G' + 0.722B' \qquad 1)$$

where *Y* is the *Y* term of the *XYZ* color space conversion from the *sRGB* color space and *R'*, *G'*, *B'* are defined by the function:

$$C' = \begin{cases} \left(\dfrac{C = 0.055}{1.055} \right)^{2.4} & C > 0.04045 \\ \dfrac{C}{12.92} & else \end{cases} \qquad 2)$$

with *C* substituted for either *R, G,* or *B*.

Once the grayscale light image has been generated, two metrics for light direction are calculated. These roughly correspond to angles defined as shown in Figure 5 but are actually derived from ratios of shading and pixel intensity in an image.

To find   as indicated in Figure 5, local 3x3 windows of pixels are analyzed across the image, similarly to the application of a convolution kernel. For each window, the center of mass of pixel intensity is found relative to the window center. This relative position is then converted to a normalized 2D vector. The average 2D vector for the entire image is converted to an angle to find   .
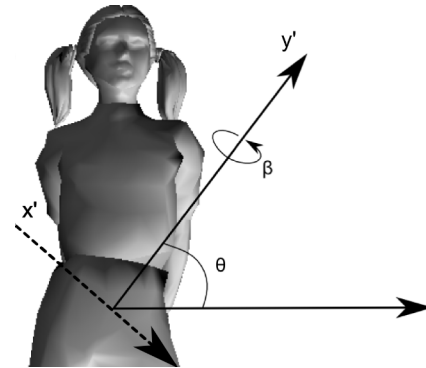

Figure 5. This figure illustrates the approximate meaning of the two light shading direction extracted from an image.

Once   has been found, the term   as indicated in Figure 5 is calculated by finding the average center of mass of pixel intensities along the axis *x'* as indicated in Figure 5, specifically:

$$\beta = \frac{\displaystyle\sum_{i=1}^{n} \left(\frac{1}{\displaystyle\sum_{j=1}^{m_i} Y_{i,j}} \sum_{j=1}^{m_i} j Y_{i,j} \right)}{n} \qquad 3)$$

where *i* is a line out of *n* adjacent lines spread across all pixels in an image parallel with *x'* and perpendicular with *y'*, m_i is the pixel count of line i, and Y_{ij} is the *Y* value of the *jth* pixel of line i.

5.1.2 Contrast

Contrast is directly quantified by the light entropy metric of [17], but it is also a factor of the term β derived in the previous section. As β increases, contrast tends to increase (and entropy decreases) because of sharper shading gradients present in the image. This is why gradient descent is used at runtime to evaluate these terms as will be described in Section 5.3. They do not correlate one-to-one with the parameters controlling light sources.

We did not use the more recent light entropy formulation of [20] because our light direction terms provide information about the spatial configuration of pixels. The update to light entropy in [20] was primarily motivated by a desire to encode spatial information about an image and this is unnecessary in our case.

5.2 Modeling an Object's Response to Light

To adapt lighting at runtime, SAIL needs to understand how an object appears under different lighting conditions. Calculating this information directly from images of an object would be too slow to perform at runtime, so SAIL generates a model of how an object responds to light in a preprocess step.

The model is generated by jitter sampling the space of control parameters of SAIL's runtime three-point lighting model. This model consists of 5 control parameters: camera direction (specified as pitch and yaw), key direction (specified as yaw and roll), and key-to-fill ratio. For each randomly sampled control parameter set, the object is rendered. This rendered image is analyzed by the image analysis component and a set of image metrics (consisting of shading direction and contrast indicators as described in Sections 5.1.1 and 5.1.2) is produced. The surface of all randomly generated image metric sets is stored and used at runtime as a mathematical model of how an object's appearance changes in response to a change in illumination.

5.3 Runtime Lighting

Lighting is applied at runtime using a constrained set of lights based on the three-point lighting model. Three-point lighting consists of a key, a fill, and a back light. The key light provides dominant direction cues, the fill light reduces tonal contrast, and the back light rims the object, isolating it from the background. SAIL only uses a key and a fill light, as the effects of a backlight are high frequency and effectively removed in the Gaussian filtering step of image analysis. Capturing and applying the effect of a back light is left for future work.

The key light is a point light that is limited to only affecting the target object. The fill light is a directional light that is similarly limited. The control parameters for this configuration are a yaw and roll for the key direction and a key-to-fill ratio for the fill intensity. The fill direction is fixed at 90º relative to the key light. The desired key intensity is always 1. The actual key intensity at runtime is an average between the intensity as sampled from an object's environment and a value of 1.

SAIL understands the appearance of a model based on the jitter sampled surface derived as described in Section 5.2. It understands a goal image through image analysis. It derives the appearance of an object under natural lighting by evaluating the jittered sampled surface at the coordinates of the effective key and fill settings as derived from natural light source positions.

Lighting adaptation at runtime is achieved by applying gradient descent to move towards the point on the jitter sampled surface that is closest to the goal image and to the point that is closest to the natural lighting state. Closeness is measured using a least squares error metric between appearance metrics.

6. EVALUATION

This sections details a qualitative study we conducted to evaluate SAIL's contributions. Specifically, we discuss SAIL's usage of image as interface and SAIL's consideration of an object's environment to adapt lighting at runtime.
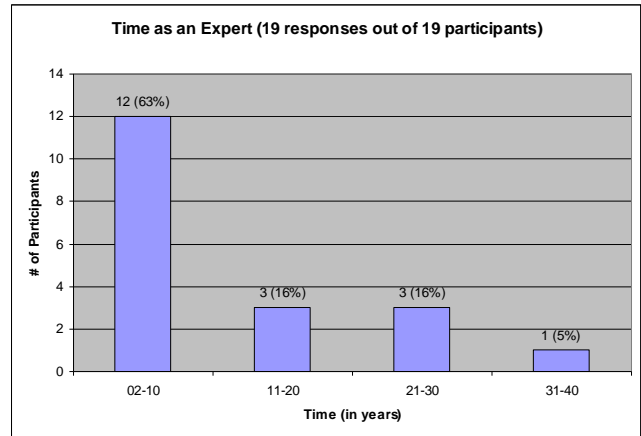
6.1 Sampling

Figure 6. Participants' experience in years.

19 participants took part in our evaluation. Participants were lighting "experts". Our definition of an expert was someone who had a personal or professional interest in lighting. Participants were recruited through personal contacts. Their years of experience are shown in Figure 6 and their specific backgrounds are shown in Figure 7.
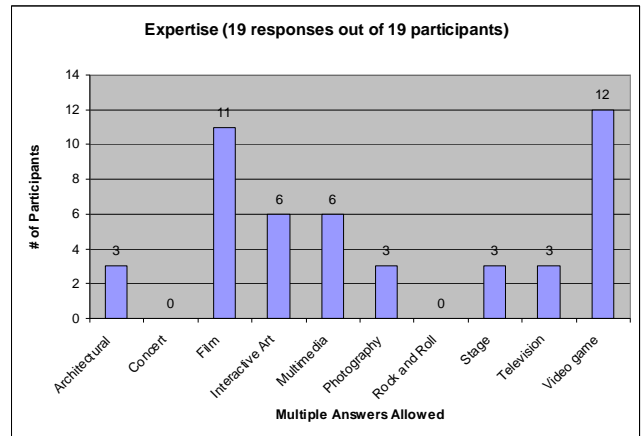

Figure 7. Participants' expertise.

Participants' activity level is shown in Figure 8. This represents the environment(s) that participants worked in. In this figure, "Academic" represents work as an educator or in production of works (such as stage plays) in academic settings. "Production" represents work in paid professional environments, such as at a

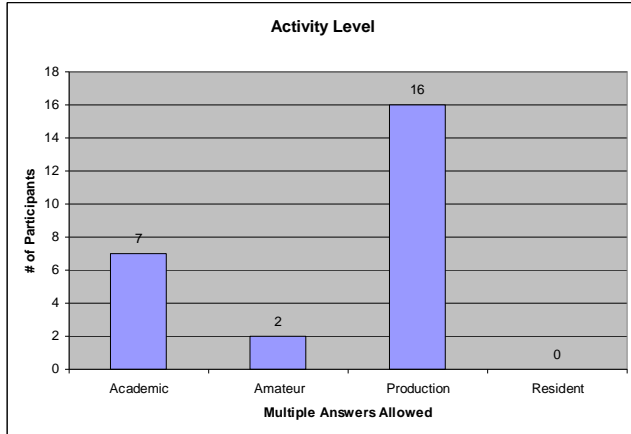video game company. "Amateur" represents work in a variety of environments but without pay or formal training.

Figure 8. Participants' activity level.

6.2 Study Design

Participants took part in 30-45 minute sessions conducted one-on-one with an investigator or a pair of investigators. Each session consisted of three parts.

In part 1, participants interacted with the implementation of SAIL shown in Figure 4. They could move the character, select between three goal images (one is displayed in the upper right of Figure 4), select between three preset light states for the environment, and switch between "auto", "natural", and "attached" mode. "Auto" mode enabled SAIL, "natural" mode disabled SAIL and used only environment lighting, and "attached" mode enabled a "hacked" version of SAIL that completely disabled context considerations (SAIL applied lighting only in consideration of the goal image).

Participants were given enough context to allow them to understand the interface and the basics of SAIL. They were told that the character was being lit automatically and that the environment was not. They were not given specific information about SAIL's algorithms or intended purpose. Participants were asked to interact with the environment until they felt that they could discuss the aesthetics and function of lighting in the demo and particularly, the behavior of lighting on the character.

In part 2, participants were interviewed using a semi-structured approach. A set of questions was used to guide discussion but individual questions could be skipped or additional questions asked based on conversation flow. Participants were not given specific information about SAIL and the investigator deferred answering these questions. For example, if asked, "I'd like to know what you're trying to accomplish here" the investigator might respond, "Let's get back to that later."

In part 3, participants were given complete information regarding SAIL's algorithms and anticipated applications. Any questions such as, "What lighting model are you using?" or "I'd like to know what you're trying to accomplish here" were answered at this time.

We approached the design of the evaluation from a deductive, qualitative theory mindset. We wanted data to argue evidentially about SAIL's usage of images as input and SAIL's consideration of appearance to adapt lighting at runtime. We felt that the views and interpretations of lighting experts were important, because lighting experts are the target of our work and it is their expert perspective that determines whether SAIL is a contribution or not.

Giving participants incomplete knowledge during parts 1 and 2 was intended to encourage them to make comments at a "mid-level", i.e. comments that would be specific but not so specific as to have little meaning outside the very particular constraints of the demo. Further, it was our hope that it would encourage them to discuss their process. We feel this was successful. For example, when asked what their interpretation of the image was, many participants explained their interpretation and also explained why they interpreted it as such.

Part 3 was included for participant satisfaction, as it was expected many would have specific questions after parts 1 and 2 about SAIL (many did). Further, it offered an opportunity to collect directed suggestions from participants about SAIL's future work.

6.3 Analysis

Passages from transcripts created of part 2 were categorized as evidence for either SAIL's usage of images or as evidence for SAIL's consideration of an object's environment at runtime. This data was then further coded into common themes as they were identified. Specifically, we found many participants discussing complete darkness, the lack of directional motivation, and the ability to "detach" lighting from the environment. Data from part 3 was included to support the arguments made from part 2.

6.4 Results

Evaluation is organized to argue two assumptions of SAIL. First, we discuss SAIL's usage of images as interface. Second, we discuss SAIL's consideration of an object's environment and its effectiveness at accommodating changing context at runtime. Note that the numerical codes used to identify participants can be used to locate quotes in the complete transcripts available in [7].

6.4.1 Images as Interface

SAIL interprets an image as shading direction and contrast information. This interpretation was more or less mirrored by participants. For example, participant *10_27_01* noted:

Investigator: So, is there any sense what it's doing specifically right now?

10_27_01: Well, when I look at the image and I look at the character I can see it does seem to be the direction of the lighting and the overall shading. I'd be curious to see how much I could mess with that image.

While there was some confusion about the role of the image at runtime, participants still interpreted the image as specifying lighting direction. For example, participant *10_26_02* noted:

10_26_02: I assume it's where the lighting is coming from so, this […] would be where the light is coming from the top or directionality as opposed to […]. Ya, I'm not quite sure. If I had to venture a guess I would probably say the intensity and the directionality of the character lighting. But I didn't see it on her so I wasn't sure if I was right or not.

This intuitive understanding of the image was probably due to its simple metaphor as a shaded sphere as described by *10_30_05*:

10_30_05: I'm familiar with lighting things in applications that have just a sphere to tell you where the light source is [pause] I'm kind of used to seeing these kinds of things

A more extensive presentation of our data can be found in [7]. Overall, our interviews indicate that an image can be an understandable interface for lighting experts to specify light direction and contrast. Whether or not this understandability extends beyond the simple metaphor of a lit sphere is an open question, but our results present encouraging direction for future research.

6.4.2 Adaptation and appearance

SAIL was successful at adapting to context by mimicking the natural lighting state. However, many participants found the natural lighting state to be unacceptable, either aesthetically or functionally. Therefore, in this section we describe what participants felt was bad about the environment lighting while illustrating SAIL's success at mimicking this environment. This discussion is important as it is indicative of a lighting expert's perspective on a tool such as SAIL, and thus it should help to ground future work on SAIL or automated interactive lighting work in general. This section discusses three themes that arose from generated data: 1) lighting was not sufficiently logically motivated, 2) intensity contrast was allowed to become too great, and 3) SAIL does not support exceptions.

6.4.2.1 Motivated light sources

Lighting direction was not sufficiently motivated by logical light sources. For example, there were large windows present in the environment that were not actually light sources:

Investigator: What are your general impressions regarding this system?

10_29_01: It's a simple system. There's a few things I have questions about. For example, there are windows over here. I'm just wondering why there are no lights that represent the windows, unless [pause] 'Cause I don't see a change going from natural to… [pause]

This problem was comparable in both SAIL's "auto" mode and in "natural" mode, where object lighting was the same as the environment lighting, implying that SAIL successfully mimicked the natural lighting state:

10_30_01: I don't know what kind of technique you use but sometimes when you walk through the rooms of the environment you don't really see the lighting change that much. Just the interaction between the lighting on the character to the environment, it's not that noticeable. For instance, you can see here…

[…]

Investigator: Is that all specifically the natural mode or is…?

10_30_01: No, it's not just natural mode, it's when you have the auto mode which is kind of a blend between natural and your attached lighting.

SAIL produced comparable results to natural mode, successfully adapting to context. This is also illustrated by the following comment about "attached" mode (where context was completely ignored) and SAIL's "auto" mode. Participant *10_28_02* felt that "kick" present in the environment was not reflected in "attached" mode but was reflected in "auto" mode:

Investigator: Would you say based on your interpretation that what the system is doing makes sense?

10_28_02: Yes. I'd want to spend a bit more time with it to see how wandering around in this environment, what the implications were?

'Cause this room for example […], there's more kick coming off the walls. So this [referring to the auto mode] gives me sort of my ambient idea and then the attached [referring to attached mode] looks like it's just being lit from the global source.

The participant later explained that "global source" referred to the lighting as indicated by the image in the demo.

6.4.2.2 Extreme light contrast

SAIL allows for too extreme light contrast:

10_30_04: That's one thing that I found that the darks are really dark. Like you'd never get blacks that black on the character, that kind of thing. […]

SAIL allows portions of the character to become completely dark, which is usually undesirable. As described in an offline conversation with participant *10_27_01*, it is implausible. Any light source in a room, due to indirect reflection, will invariably cause some illumination to illuminate everywhere and allow our eyes to make out some detail. A completely dark area is implausible and limited to a rare, special case effect.

6.4.2.3 Exceptions

SAIL does not support "exceptions" to the rule and lighting experts will invariably want to make exceptions. Every automated interactive lighting system will need to base its behavior on rules. For example, a rule of SAIL is to motivate lighting direction on an object by the direction of lighting present in that object's environment. However, participant *10_29_03* notably liked the demo's ability to completely detach an object's light direction from the environment:

10_29_03: What I really am most fond of is this ability to detach [pause] I like that ability to detach the figure from the environment. Just because it's a kind of shadow like effect where I can make things that I regard as less important [pause]

SAIL was never designed to operate in this way. This ability to detach lighting was a "hacked" feature of the demo for comparison purposes. In practice, SAIL would have no ability to consider or incorporate a complete exception to its rule of motivated lighting direction such as this.

Similarly, we argued in Section 6.4.2.2 that extreme light contrast is usually undesirable. An obvious future rule for SAIL would therefore be to disallow light contrast above a certain threshold. However, SAIL would still need to allow for exceptions to this rule, as described by participant *10_29_06*:

10_29_06: It depends on what exactly you want to do. If she was trying to move [pause] I'm a pensitive person and I come here and now I actually realize that I'm guilty [10_29_06 has moved the character to a spot where much of her face is complete blackness] where she slowly transitions to dark, then I get it.

Our conclusion is that lighting experts will inevitably want to make an exception to any rule that guides an automated lighting system. Therefore, it is important for a system to have an integrated, purposeful mechanism to allow for exceptions.

7. FUTURE WORK

Our plans for future work are twofold. First, we plan to further explore images as interface for lighting design tools. Particularly, we wonder what the limits of a single image are. Can it effectively indicate multiple key lights, complex patterns, or

shadow? Also, can a single image specify lighting for an entire scene?

Further, we will explore modifications to SAIL's adaptation. SAIL's adaptation was successful in that it accommodated context, even when that context is bad. SAIL's adaptive behavior should be modified to limit extreme light contrast and allow for exceptions. For example, it should be possible for a designer to completely ignore contrast limits and create a silhouette or ignore direction constraints and create a theatric spot light effect.

8. CONCLUSIONS

This paper presented the System for Automated Interactive Lighting (SAIL), our adaptive lighting system. Our evaluation of SAIL indicates that images can be an effective interface for lighting experts to specify lighting goals. It also shows that while SAIL successfully considers context, it does so whether that context is good or bad from an expert's perspective.

Our conclusions are that we should further pursue images as lighting interface. We should also consider additional mechanisms for SAIL's adaptation of lighting at runtime. Specifically, SAIL should integrate exceptions to allow designers to violate SAIL's fundamental assumptions (such as maintaining motivated illumination direction). SAIL should also allow for constraints to be imposed on parameters such as light contrast to avoid results such as complete darkness.

9. ACKNOWLEDGMENTS

10. REFERENCES

[1] Birn, J., *[digital] Lighting & Rendering*. 2nd ed. 2006, Berkeley, CA: New Riders. 416.

[2] McTaggart, G., *Half-Life 2 / Valve Source Shading*. 2004, Valve Corporation. p. 97.

[3] Seif El-Nasr, M. and I. Horswill, *Automating Lighting Design for Interactive Entertainment.* ACM Computers in Entertainment, 2004. 2(2): p. 15-15.

[4] Callahan, S. *Storytelling Through Lighting*. in *SIGGRAPH*. 1996: ACM.

[5] Millerson, G., *Lighting for Television & Film*. 3rd ed. 1991, Linacre House, Jordan Hill, Oxford, UK: Focal Press. 466.

[6] Reid, F., *The Stage Lighting Handbook*. 6th ed. 2001, New York, NY: Routledge. 217.

[7] Zupko, J., *A System for Automated Interactive Lighting (SAIL)*. 2009, The Pennsylvania State University: State College, Pennsylvania. *Dissertation*.

[8] Koonce, R., *Deferred Shading in Tabula Rasa*, in *GPU Gems 3*, H. Nguyen, Editor. 2008, Addison-Wesley: Upper Saddle River, NJ. p. 429-457.

[9] Mittring, M., *Finding Next Gen - CryEngine 2*, in *SIGGRAPH 2007*. 2007, ACM: San Diego, California. p. 97-121.

[10] Green, R., *Spherical Harmonic Lighting: The Gritty Details*, in *Sony Computer Entertainment America Tech Reports*. 2003, Sony Computer Entertainment America. p. 47.

[11] Chen, H. and X. Liu. *Lighting and Material of Halo 3*. in *SIGGRAPH 2008*. 2008. Los Angeles, California: ACM.

[12] Blinn, J., *Simulation of wrinkled surfaces*, in *SIGGRAPH*. 1978, ACM. p. 286-292.

[13] Patow, G. and X. Pueyo, *A Survey of Inverse Rendering Problems.* COMPUTER GRAPHICS forum, 2003. 22(4): p. 663-687.

[14] Kawai, J.K., J.S. Painter, and M.F. Cohen, *Radioptimization - Goal Based Rendering*, in *ACM SIGGRAPH 1993*. 1993.

[15] Flynn, J.E., *A study of subjective responses to low energy and nonuniform lighting systems.* Lighting Design and Application, 1977.

[16] Costa, A.C., A.A. Sousa, and F.N. Ferreira, *Lighting Design: A Goal Based Approach Using Optimisation.* EUROGRAPHICS, 1999.

[17] Gumhold, S., *Maximum Entropy Light Source Placement.* IEEE Visualization 2002, 2002: p. 275-282.

[18] Lee, C.H., X. Hao, and A. Varshney, *Geometry-Dependent Lighting.* IEEE Transactions on Visualization and Computer Graphics, 2006. 12(2): p. 197-207.

[19] Shacked, R., *Automatic Lighting Design using a Perceptual Quality Metric*, in *Computer Science and Engineering*. 2001, The Hebrew University of Jerusalem: Jerusalem, Israel. p. 68.

[20] Vázquez, P.-P. and M. Sbert, *Perception-Based Illumination Information Measurement and Light Source Placement*, in *ICCSA 2003*. 2002.

[21] Shannon, C.E., *A Mathematical Theory of Communication.* Bell System Technical Journal, 1948. 27(1): p. 379-423, 623-656.

[22] Halle, M. and J. Meng, *LightKit: A lighting system for effective visualization.* IEEE Visualization 2003, 2003: p. 363-370.