A tool for Adaptive Lighting Design

Joseph Zupko
College of Information Sciences and Techonology
Pennsylvania State University
jaz147@psu.edu

Magy Seif El-Nasr
School of Interactive Arts and Technology
Simon Fraser University
magy@sfu.ca

Abstract

Filmmakers, theater directors, and designers take extreme care in aesthetically composing their scenes. This sense of aesthetics is important to promote presence, immersion, and involvement. Video games have adopted many design theories and techniques from linear media, particularly film. However, interactive environments are dynamic and unpredictable, and thus require the development of theories, techniques, and tools specific to their interactive nature. In this paper, we present a lighting design tool that algorithmically adapts lighting to user interaction within game environments at runtime. Previous work includes our work on the Expressive Lighting Engine (ELE), which uses intelligent algorithms to adapt lighting in real-time accounting for variation in context, narrative, and intended artistic effects defined through numerical constraints. We encoded cinematic lighting techniques within ELE as mathematical formulae that guide the system's choices in terms of lighting to achieve artistic constraints while maintaining visual continuity. The work presented in this paper expands this work in several directions. First, ELE did not take objects' or scenes' materials into consideration. This is important since the appearance of objects under lighting conditions depends on their materials. Second, using numerical constraints as a method by which designers encode their artistic intention for lighting proved to be unintuitive. Thus, we present a prototype that builds on ELE and focuses on exploring solutions to resolve these problems.

Categories and Subject Descriptors: H.5. [**Information Systems**]: Information Interfaces and Presentation (e.g., HCI).

Keywords: interactive environments, aesthetics, lighting design

1 Introduction

Lighting is an important part of creative media, including theater, film, animation, and game productions. Our perception of the world is shaped through its lighting. Although lighting is most obvious for its influence on visibility, it has many other functions due to its effect on mood, perception of time of day, and its influence on visual attention [Block 2000; Millerson 1991; Calahan 1996]. Artists use lighting as a powerful tool to control meaning and form. The great works of Rembrandt and Vermeer are explored today for their lighting composition.

Similarily, lighting in games is important for its influence on mood, dramatic tension, visibility, visual attention, and its ability to affect engagement and immersion [Seif El-Nasr et al. 2006]. The current methods of game lighting design borrow from traditional linear media techniques, which necessitate manual adjustment of many parameters. This manual adjustment is key to creation of desired artistic effects. For example, the illusion of light cast by a window into a room when filmed might be created by many carefully placed and adjusted lighting instruments. Film lighting designers often add several spot lights to a scene to supplement the natural lighting setup. This is done for various artistic reasons, including creating depth and ensuring that a character's eyes, for example, are lit at all times.

Lighting a game environment is more complex due to user interaction which introduces unpredictability. A carefully placed and adjusted lighting design can be completely negated by unanticipated user choices at runtime. This then requires the development of systems that allow game lighting designers to author adaptable lighting compositions to accommodate unpredictable user behaviors. Our research focuses on this problem.

Adapting the lighting composition at runtime is not an easy problem. An easy solution is to use ambient lighting, a technique that evenly lights the entire scene; this technique, however, produces dull and unpleasant environments [Birn 2000]. An alternative solution, which many of the current techniques favor, is to light the environment according to physically accurate lighting models. While this solution produces a realistic design, it does not establish artistic or aesthetic goals. Lighting designers in film, for example, often supplement natural scene lighting with additional lights to convey moods, evoke emotions, or create visual depth [Evans 2006 and 2008]. Thus, in addressing adaptive lighting design, a system will have to address adaptive *aesthetic* lighting design, which is difficult due to its subjective and ill-defined nature. Even though there are many theories on film, theater, and animation lighting design, there is very little understanding of what constitutes good or aesthetically pleasing lighting in a given situation.

In the past, we have developed several systems to tackle this problem. In 2003, we developed a dynamic lighting design system called the Expressive Lighting Engine (ELE) [Seif El-Nasr and Horswill 2004]; ELE encodes artistic goals for lighting and uses artificial intelligence to adapt the lighting composition, including setup, direction, and color, within a 3D interactive environment based on theatric and cinematic lighting design theories represented mathematically within a constraint optimization system. While this system achieved some success, it did not account for different materials used in scene objects, and thus did not always perceptually achieve the intended artistic or aesthetic goals. Also, ELE's interface allowed artists to encode artistic rules in terms of constraints which was not a very well received method by our subject pool. Instead, lighting design by example was suggested as an alternative interface.

In this paper, we will discuss a work in progress that tackles the two problems discussed above. Mainly, we discuss a system that uses images as examples given by designers to the system. The system then uses these images to deduce a lighting setup for 3D objects within a 3D scene, given context and scene layout. The system also uses a visual perceptual model and cinematic lighting design principles to achieve an appearance that is close to the intended aesthetic effect. In other words, it takes into account how objects respond to lighting when developing a lighting setup, thus achieving a perceptually motivated appearance given the artistic effect intended.

2 Lighting Design Process – Traditional Media

To frame the problem of adaptive interactive lighting design, it is useful to describe lighting design in existing media, such as film and theater. The role of a lighting designer is to establish a lighting setup (positions, colors, and angles for lights and their changes within the scene) that serves several goals, including establishing visibility, portraying motivation for direction, portraying appropriate depth and modeling, and evoking mood [Alton 1995; Birn, 2000; Block 2001; Calahan, 1996; Campbell 1999; Crowther 1989; Gillette 1998; Kidd 2001; Knopf 1979; Lowell 1992].

In creating a lighting design, the designer considers the environment, the characters, and the narrative. Specifically, when looking at the environment he considers several variables, including the time of day, the period, the style of rendering, and the mood or theme [Alton 1995; Birn 2000; Block 2001; Brown 1996; Gilette 1998; Millerson 1991]. To accomplish these goals, designers are required to understand how lighting works in the real world and to understand the aesthetic style that they want to achieve. They must also establish a sense of lighting direction, fall-off ratio, shadow density, brightness, and color composition. Different lighting design methods were created for film [Alton 1995; Block 2001; Brown 1996; Millerson 1991], theater [Gillette 1998], and animation [Birn 2000] to achieve these goals.

While a review of lighting design theory is beyond the scope of this paper, we will outline a basic lighting technique that is regularly used in theater [Gilette 1998], film [Millerson 1991], animation [Birn 2000], and games [Evans 2006 and 2008]. The technique is called three-point lighting. This technique is mainly used to light characters but can also be used to light the environment [Evans 2006 and 2008]. Three-point lighting generally consists of three lights or sets of lights: a *key* light, a *fill* light, and a *back* light (more than one light is often used for backlighting an object or scene). The key light provides dominant direction cues while the fill light controls tonal contrast. The back light(s) is used to isolate the silhouette of the object.

In addition, lighting designers also use eye lights to brighten the eyes or body lights to accent a character's body. They also use lighting patterns to evoke specific emotions such as anticipation. Example patterns include: under-lighting a character to enhance abnormality, light flickering (e.g. lightning), and backlighting a character to enhance drama.

3 Lighting Design Process – Games

Lighting design in games has not been formalized. Most lighting design lectures and literature focuses on adapting film lighting techniques such as three-point lighting [Marino 2004; Evans 2006 and 2008]. Most lighting design for games is static and environmental in nature. Level designers develop a 3D environment using a modeling tool such as *Maya*. Lighting is then applied to the level to create the desired architectural aesthetic effects while maintaining illumination for dynamic objects.

Some games use limited dynamic lighting designs. The function of dynamic lighting differs from game to game. For example, some designers attach dynamic lights to characters to ensure they are lit at all times. Player characters can also be coerced into positions where preset lighting will create desired effects.

In general, game lighting design is labor intensive and rigid. Designers must manually place environmental lighting, configure dynamic character lighting, and balance the two to achieve reasonable results given all anticipated cases. Even then, designs can produce undesirable results in cases that designers did not anticipate, or where an assumption has changed. For example, if a dynamic light attached to a character is configured based on the assumption that the average light brightness in an environment will be "very dim" and the light brightness in the environment is later changed, the light may no longer have the desired effect.

4 Previous Work

Previous work on lighting design systems and tools can be classified into two categories. The first is focused on simulating physically accurate lighting (e.g., [Debevec 2005]). The second, more relevant to our work, is *inverse lighting*, where a lighting configuration is computed to achieve a lighting goal specified by a user, often by a 2D image and some correlated 3D geometry [Patow and Pueyo 2003]. Work in this domain can be divided into *architectural* inverse lighting, *perceptual* inverse lighting, and *interactive* inverse lighting.

4.1 Architectural Inverse Lighting

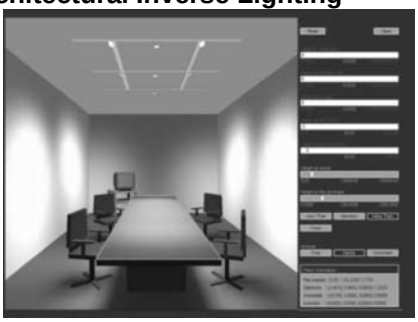

Fig. 1. An automatically illuminated virtual space [Kawai et al. 1993].

Work presented in [Kawai et al. 1993] was one of the first inverse lighting works (Fig. 1). The domain of Kawai's work was finding light settings for real-life environments using architectural visualization tools. The system of this work found angles and intensities for lights given: 1) light positions manually specified by a user, and 2) numerically defined goals for the lighting design. These goals were subjective in nature and derived from previous work by [Flynn 1977]. Flynn explored participants' subjective experiences to lighting in spaces, such as whether a room felt "spacious", using psychophysical techniques.

[Costa, Sousa et al. 1999] used a similar approach and is the latest and most complete approach to the domain of architectural lighting design (Fig. 2). Costa's approach made only three assumptions about an environment: 1) surface reflectance can be described with symmetrical Bi-Directional Reflectance Functions (BRDFs), 2) the environment has no participating media, and 3) the light model used in the rendering environment obeys the photon nature of light. A symmetrical BRDF is a reflectance function for which an inverse can be calculated.

Within these assumptions, Costa allowed designers to specify almost any illumination goal using scripting. For example, a designer might code a rule that constrained intensity on a surface within a specific range. The system would then find light sources that fulfilled this and all other rules for a given space.

Architectural lighting work in-general appears "solved". Costa in-particular has produced a general-purpose and flexible system for

visualization and designing architectural spaces. Unfortunately, the algorithms of this domain are too slow for interactive virtual rendering. Both Kawai and Costa, in-particular, require global illumination rendering (such as Radiosity or Raytracing). Further, the constraints of architectural visualization are very different from those of real-time visualization. Architectural visualization must produce physically possible solutions and must consider real-world factors such as energy efficiency and cost effectiveness. These considerations are therefore a fundamental part of architectural visualization solutions. These considerations are not necessarily true in real-time interactive rendering.

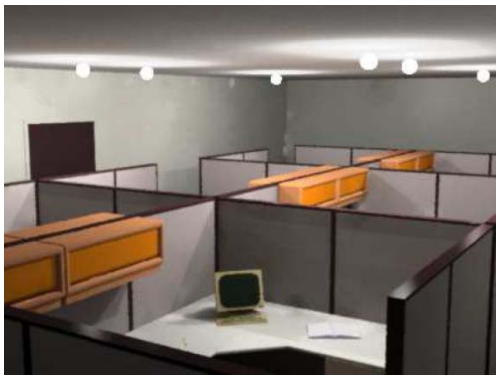

Fig. 2. This figure shows a virtual scene of an office illuminated by the system in [Costa, Sousa et al. 1999].

4.2 Perceptual Inverse Lighting

Perceptual inverse lighting includes the largest body of literature in inverse lighting [Poulin et al. 1997; Shacked 2001; Gumhold 2002; Vázquez and Sbert 2002; Shesh and Chen 2004; Lee, Hao et al. 2006]. Work in this field operates in the image domain, using metrics derived from an image and possibly correlated geometry. In [Shacked 2001; Gumhold 2002; Vázquez and Sbert 2002; Lee, Hao et al. 2006] lighting is optimized without user interaction. The goal of lighting is derived from visual perception theories to achieve "ideal" settings to maximize the perception of shape, detail, and depth. Other works allow a user to specify a goal by annotating images, by drawing shadows or shading.

Fig. 3. The 3D model of a ship in this figure is illuminated by the system of [Shacked 2001].

The system of [Shacked 2001] found light source settings to maximize perceptual features such as shape, detail, and depth perception (Fig. 3). [Gumhold 2002; Vázquez and Sbert 2002] targeted a similar goal but used a metric based on information entropy [Shannon 1948]. All of these works calculated metrics from generated images of an object. Images were generated iteratively and evaluated for lighting. Lighting was then changed and the process repeated. Eventually, the best solution was found.

The work of [Lee, Hao et al. 2006] is the most recent and complete work in this area. Lee approached the problem slightly 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 shadow (Fig. 4) was also explicitly considered, the only work in this field to do so.

This body of literature illustrates that while image analysis is useful for specifying lighting goals, straightforward analysis of images is incompatible with real-time rendering. Rendering images at runtime is too computationally expensive. No prior work in perceptual lighting was applied to the interactive domain. Therefore, the work of this paper offers a unique exploration of real-time interactive only considerations to this domain.

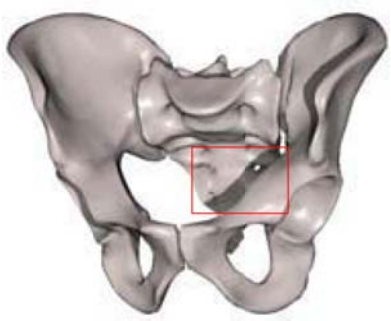

Fig. 4. This image from [Lee, Hao et al. 2006] illustrates that system's consideration of cast shadow on an object to help identify depth in a figure. The red box highlights a shadow that the system explicitly manipulated to help identify the structure of this 3D figure.

4.3 Interactive Inverse Lighting

The only prior work in interactive inverse lighting aside from our work with the Expressive Lighting Engine is LightKit [Halle and Meng 2003]. LightKit uses film's three-point lighting as a constraining lighting model. In LightKit, the key light is a light that provides a dominant sense of direction and it is always located at 45° relative to the orientation of the camera. The fill light is fixed at 90° relative to the key and is maintained at an intensity ratio of 2:1 to the key. Light kit also uses two back lights that illuminate an object from behind and help to isolate it from the background. A user can adjust this lighting configuration by specifying the intensity of the key light or the color of the key light using a warmth scale derived by [Halle and Meng 2003].

While useful for its domain of medicine, the LightKit has limited application to video games. The LightKit simplifies the control of illumination but it still requires a user to actively control illumination at runtime. In the domain of this paper, lighting is generally a supporting element and not a first-order concern for user interaction.

4.4 Expressive Lighting Engine (ELE)

Seif El-Nasr and Horswill [2004] proposed ELE. ELE is a dynamic intelligent lighting system developed based on cinematic lighting design theory to use artistic constraints to compose a

lighting design which it adjusts and manipulates dynamically, accommodating interaction while achieving artistic goals. ELE is divided into three subsystems: an allocation subsystem used to select the number of lights and their locations, an angle subsystem which selects angles for each light, and a color subsystem which selects colors for each light. We will briefly discuss these sub-systems in this section.

4.4.1 ELE – the System

ELE uses stage layout, scene graph information, and artistic constraints to create a light layout. ELE divides the scene into *n* different areas and categorizes these areas as focus, non-focus, or background. This information is then used to identify focus, increase depth, and increase contrast based on designer goals. ELE also determines where to direct a player's attention given characters in the scene. Thus, ELE maximizes a multi-objective function to determine the number of lights to use for each area:

$$p_{opt} = \arg\max_{p} \left(\lambda_v V(p) + \lambda_d D(p) + \lambda_m M(p) + \lambda_{vc} VC(p) \right), \qquad \textbf{(1)}$$

where p is the light configuration and the weights represented by λ are the artistic constraints. Specifically, λ_v is the importance of visibility, λ_d is the importance of depth, λ_m is the importance of modeling, and λ_{vc} is the importance of visual continuity. $V(p)$ is visibility given p, $D(p)$ is depth given p, $M(p)$ is modeling given p, and $VC(p)$ is visual continuity given p.

In determining the angles of light, ELE also takes into account the quality of lights and their influence in projecting depth, modeling, and mood. ELE uses non-linear optimization to select an angle for each key light that minimizes the following function:

$$\lambda_v (1 - V(k,s)) + \lambda_{\text{-}} \left| k - k^{-} \right| + \lambda_m \left| k - m \right| + \lambda_l \min_i \left| k - l_i \right|, \qquad \textbf{(2)}$$

where k and s are defined as the key light azimuth angle relative to the camera and the subject angle relative to the key light. k^{-} is the key light azimuth angle from the previous frame and λs represent artistic constraints. Specifically, $\lambda_{\text{-}}$ is the cost of changing the key light angle over time (to enforce visual continuity), λ_m is the cost of deviation from the mood azimuth angle, m is the mood azimuth angle suggested by the artist, λ_l is the cost of azimuth angle deviation from a practical source direction, l_i is the azimuth angle of light emitted by the practical source i, and λ_v is the cost of deviation from an orientation of light that establishes best visibility.

The interaction of lighting colors in a scene composes the contrast and feeling of the entire image. Similar to the angle and layout systems, ELE uses non-linear optimization to search through a nine-dimensional space of RGB values. It differentiates among focus colors, non-focus colors, and background areas to select a color for each individual light in the scene. The multi-objective cost function evaluates the color against the lighting-design goals, including establishing depth, conforming to color style and constraints, paralleling dramatic tension, adhering to desired hue, saturation, and lightness, and maintaining visual continuity.

4.4.2 Shortcomings of ELE

After experimenting and using ELE for several years with various users, we deduced that ELE is not easy to use. ELE abstracts low-level lighting design constructs (light colors, positions, and angles) into higher level numerical constraints. These constraints include more than twenty-five real numbers that are dependent in non-intuitive ways. Lighting designers who have interacted with ELE often fail to achieve the desired results, even though their results are theoretically achievable within ELE's abstraction.

It is also not known whether ELE's abstraction aligns with how game lighting designers actually conceptualize lighting design. Abstractions always enforce specific limitations and constraints on representation and it is critical that these limitations and constraints are acceptable to game lighting designers for ELE to be useful.

In addition, ELE did not take into consideration the materials or textures of the objects or the environment. This is important as lighting design relies on perceptual quality and appearance, which is a function of all visual elements in an environment.

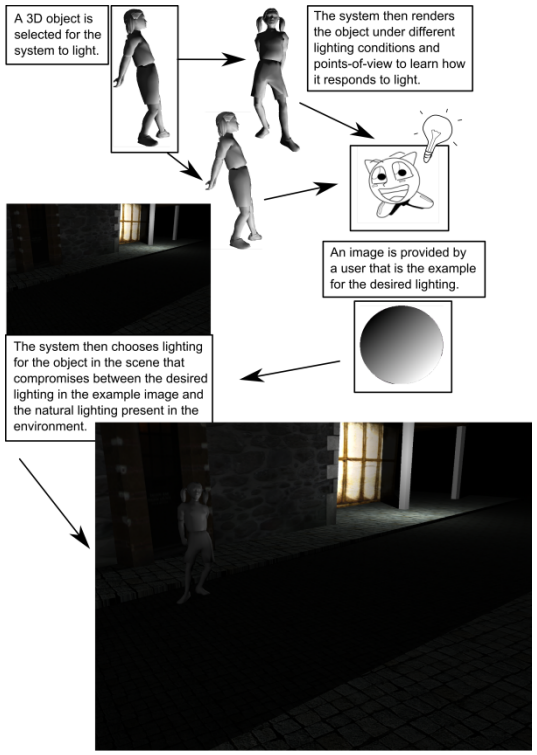

Fig. 5. This figure illustrates the process of the system.

5 Current Work

In this section, we discuss a work in progress which attempts to overcome some of the problems faced with ELE discussed above. In particular, this section will address early experimentation in integrating image-based analysis approaches, color and lighting perception theories, film techniques, and artificial intelligent techniques to achieve intended lighting effects in a 3D environment.

In this current system, we decided to constrain the problem. First, instead of deducing lighting setup for entire scenes including objects and environment, we focuse on finding lighting parameters for an object or a set of objects within a scene. Second, we only deduce lighting direction and brightness contrast. It is assumed that lighting is performed using primitives of Rasterization. Specifically, two lights are used, a *key* and a *fill* light, following film's three-point lighting technique, described

previously. Thus, the problem is narrowed down to finding the best lighting direction and brightness for *key* and *fill* lights for objects within a scene.

Fig. 5 summarizes the system currently under development. The system is composed of two main subsystems. The first subsystem calculates a response surface that encodes the response of an object under different lighting conditions and camera angles using metrics extracted from final renders of the object. The second subsystem is a result of our earlier exploration of lighting a 3D scene using an image as an example [Zupko and Seif El-Nasr 2006]. This subsystem finds light primitives to create illumination at runtime that approximates lighting in an example image as closely as possible based on information from the response surface calculated by the first subsystem.

5.1 Calculating lighting appearance

At runtime, the system needs to know the effect of lighting on object appearance. For example, increasing the intensity of the fill light will decrease the overall intensity contrast on the object by different amounts depending on the object. The system also needs to be able to quickly calculate the illumination metrics for an object given camera direction and illumination parameters. To calculate this information quickly, the system finds a response surface for the object being lit. This process is done off-line. The response surface is a five-dimensional surface of data samples relating independent variables to dependent results. The independent variables are: the *camera direction*, *key light direction*, and *fill intensity*. The dependent variables are the *shading direction* and *intensity contrast* as expressed by a render of the object using the independent variables as render settings.

The response surface is represented as a coarse sampling of points. These points are sampled by rendering the object with random camera direction, key light direction, and fill light intensity settings and then processing the image to extract metrics representing the shading direction and intensity contrast. Random sampling of camera direction, key light direction, and fill light intensity is achieved using *jitter sampling*. Jitter sampling divides the sampling space into an evenly spaced coarse grid of samples and then randomly selects a point within each grid cell. This approach is used as it reduces the number of samples necessary to produce an accurate representation of the response surface but has been shown to be statistically equivalent to an un-constrained random sampling.

5.2 Design by example

The system uses an image supplied by a designer as an example of the lighting effect that she wishes to create. To mimic this effect, the system performs image analysis on the example image. The algorithms for this analysis are based on theories from *visual perception*.

5.2.1 Visual Perception

Visual perception is a discipline that studies the process of sight. Our ability to see is a combination of both the reception of light by our eyes and the interpretation of light stimulus by our brains. Understanding the interpretation of light aesthetic requires an understanding of visual perception theories. The visual perception theories related to our work come from the sub-field of visual perception known as *psychophysics*. Psychophysics maps low-level stimuli to subjective interpretation. A key theory of the psychophysics of illumination perception is *color constancy*.

Color constancy is the human visual system's ability to adapt vision to achieve constant color. Constant color is why gray surfaces still appear gray under different illumination [Adelson 2000] and a green apple looks green under sunlight or fluorescent light. All human interpretation of light occurs through the color constancy effect. Accounting for this effect is therefore important for a system to "see" an image like we do.

There is no single mathematical model for color constancy but see [Barnard, Cardei et al. 2002] for a survey and comparison. These models make several common assumptions. Surfaces are assumed to be diffuse and light is assumed to be "sufficiently distant" such that appearance changes due to illumination are low-frequency. Processing is also assumed to be independent across color channels. This means that the red, green, and blue channels of the RGB color model are each processed independently from the other channels.

A particular mathematical model for color constancy which achieves the common assumptions listed here is *Retinex* [Land and McCann 1971]. A simple version of Retinex used in the system presented in this paper is the *Single-Scale Retinex* (SSR) [Jobson and Woodell 1995]. SSR was selected in contrast to other approaches because it is computationally cheap to apply. It also requires no tuning parameters which is a desirable feature to keep the system robust across changes in context.

5.2.2 Image analysis

The system extracts two types of information from the image: 1) light direction as indicated by shading, and 2) intensity contrast. Light direction roughly corresponds to the direction of a key light but is a combination of the effects of all light sources present in an image. Intensity contrast roughly corresponds to the intensity of a fill light but it is also a factor of the direction of light sources in the image.

5.2.2.1 Finding Intensity Contrast

An image is first processed using the Single-Scale Retinex (SSR) algorithm of [Jobson and Woodell 1995] to produce an illumination image based on the Retinex model of color constancy [Land and McCann 1971]. An illumination image is a filtered version of an image that does not contain high-frequency pixel intensity changes. This is based on the assumption that lighting contributes only low-frequency information to an image while high-frequency information is due to surface reflectance. The original form of SSR is arranged to produce an illumination independent reflectance image:

$$R(x,y) = \log L_i(x,y) - \log\left[F(x,y) * L_i(x,y)\right] \quad (3)$$

where i denotes the color channel, $L_i(x, y)$ is the *ith* channel of the source image, * is the convolution operator, and $F(x ,y)$ is a spherical Gaussian function used to filter the source image. For the purposes of this work the illumination image is the goal term, which is:

$$I_i(x,y) = F(x,y) * L_i(x,y) \quad (4)$$

where $I_i(x, y)$ denotes the *ith* channel of the illumination image. All calculations use the *Y* value of the *XYZ* color space conversion from the *sRGB* color space [Gumhold 2002], which is:

$$Y = 0.2126R' + 0.7152G' + 0.722B' \quad (5)$$

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:

$$C' = \begin{cases} \left(\frac{C = 0.055}{1.055}\right)^{2.4} & C > 0.04045 \\ \frac{C}{12.92} & else \end{cases} \quad (6)$$

with C as either R, G, or B from the RGB color space.

To find terms correlated with the shading direction, first assume that the surface of a given object is Lambertian and convex. The outgoing light intensity from a surface patch covered by pixel i in an image is:

$$L_o(\mathbf{n}, \mathbf{l}) = AL_i(\mathbf{n} \cdot \mathbf{l}) \quad (7)$$

where $\boldsymbol{n}$ is the normal of the surface patch covered by pixel i, $\boldsymbol{l}$ is the direction from the surface patch to an imagined light source, A is the albedo of the surface, and L_i is the intensity of an imagined light source. A *normal* is a unit-length vector with two degrees of freedom that indicates the direction of the "face" of a surface patch, where the face is the side of the patch that responds to light. The outgoing light intensity L_o is the pixel image data and the goal is to find image terms that can be correlated against lighting parameters for a key light and a fill light.

The filtering of the image to produce an illumination image removes high-frequency pixel intensity changes. As a result, it can be assumed that the shading direction in an image is a function of the change in pixel intensities across an image due to an unknown but convex change in surface normal $\boldsymbol{n}$ at each pixel of the image, the low-frequency effects of albedo, and the unknown but predictably low-frequency changes in imagined light direction $\boldsymbol{l}$ at each pixel of the image.

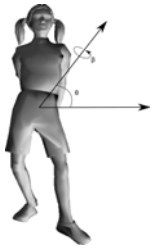

Fig. 6. This image illustrates the definition of image direction metrics θ and β.

5.2.3 Finding Lighting Direction

To quantify this information as an indicator of direction two terms are defined, θ and β, shown in Fig. 6. To find θ, a local 3 x 3 window of image pixels is examined, starting at the upper-left corner of the image. The center of mass is found in the window, where "mass" is defined as the intensity of a pixel. A two-dimensional vector is formed between the center of mass within a window and the center of the window. The window is then "slid" across the image, calculating a vector for each, until every pixel in the image has been the center of a window. The average of these vectors is then calculated and converted to an angle to find θ.

Once θ is found, the term β can be calculated. β is defined as:

$$\beta = \frac{\sum_{i=1}^{n} \left(\frac{1}{\sum_{j=1}^{m_i} Y_{i,j}} \sum_{j=1}^{m_i} jY_{i,j} \right)}{n} \quad (8)$$

where i is the ith line out of n adjacent perpendicular lines to the axis that β is defined around, m_i is the number of pixels in a line i, and $Y_{i,j}$ is the intensity of the jth pixel of the ith line.

To find a term correlated with intensity contrast, the entropy metric of [Gumhold, 2002] is used. This is defined as:

$$H = \sum_{i=1}^{m} p_i \log \frac{1}{p_i} \quad (9)$$

where m is a number of bins used to segment the *intensity* value of a pixel, and p_i is the probability of a pixel falling into the bin i. p_i is defined as:

$$p_i = \frac{c_i}{n} \quad (10)$$

where c_i is the number of pixels falling into a bin i and n is the total number of pixels in the image. The i of a pixel is defined as:

$$i = \left\lfloor m\left(Y' + \frac{1}{2}\right) \right\rfloor \quad (11)$$

where Y' is the *intensity* value for a given pixel normalized to the range [0,1] and m is a number of bins used to segment the Y' value of a pixel. The bins are currently set to m = 30 bins.

5.3 Lighting the scene

At runtime the system makes decisions by using the response surface, the state of the rendering environment, and an example image provided by a designer. The system uses two sets of inputs for each object in the scene: one is the illumination features extracted from an example image and the other is the illumination features that would be created if the lighting motivated by the scene were applied to the object. The first set of features is given by the image analysis process of the system while the second set is derived from the response surface. This is because while the light source parameters in the scene are given (the system knows explicitly where light sources are located in the scene) the results of lighting an object with those parameters are not.

Therefore, the system derives a set of illumination features by using the response surface. Generating these features is a matter of finding the nearest sampled points on the surface to the point of the current camera direction and motivated light source parameters (key direction and fill intensity). Once the points are found, the illumination features for each point are averaged.

The system then uses stochastic gradient descent to step from the natural illumination towards the ideal illumination. The best solution is the point that produces illumination metrics which are equally distant from the natural and ideal. Essentially, the system