E-Lumination - Lighting 3D Scenes Using Examples

Bruno Lourenço and Carlos Martinho

Instituto Superior Técnico, Lisboa, Portugal

October 2013

Abstract

Lighting has much more importance than simply establishing visibility. In cinematography, theatre and videogames, good lighting is expected to create images with emotional content, set atmosphere and mood, direct attention and provide depth to a two-dimensional image. Lighting design is the process of finding light parameters: positions, directions, colours, and intensities to achieve the visual properties wanted for a specific scene. The traditional method of lighting design for a scene is iterative, requiring trial and error until the desired goal is reached. This is a repetitive and tedious work, which requires technical and artistic expertise in the field. Inverse lighting design aims to provide designers with semi-automated approaches and easy to use tools, to configure lighting for scenes, minimizing the time and expertise needed. This work will explore lighting by example. More specifically, the possibility of using videos and images as examples for lighting 3D scenes. The major goal of the proposed system is to emulate in the virtual scenes the same mood present on the short videos or images used as input. Tests were conducted with users, rating the mood of images used as input of the system and the resulting automatically illuminated 3D scenes. The obtained results were satisfactory, presenting some similar ratings.

Keywords: lighting design, inverse lighting, lighting by example, automatic lighting, image processing

1. Introduction

The physical world is revealed by light. Our perception and most visual artistic expressions are greatly dependent on light. Many painters and other artists are known for their lighting mastery as their work depend heavily on lighting contrasts to create memorable paintings. Not only artists, but also scientists and engineers study and learn to manipulate light for both theoretical and practical goals within many areas of interest.

Since lighting is a major influence in visual perception, theatre, film, architecture and videogame communities devote much time finding ways to control and master it. But good lighting is not just about good visibility, lighting also has a major impact evoking emotion, mood and directing attention.

In cinematography, lighting is a major component for achieving good quality video. First of all, to have a proper exposure, the light must reach the camera sensors[1] and raise the signal to proper levels without exceeding the limits. Secondly, lighting should make the viewers forget they are looking at a two-dimensional rectangle and help creating the illusion of three-dimensionality. Finally, as mentioned before, lighting is a great tool for directing the viewers attention and adding mood and feeling to a scene. Although exposure control is not necessary in computer cinematography, the last two are still very important.

There are rules, guidelines and techniques known to the cinematography community (traditional and computer based) that are used to create memorable scenes and images. In this work, we will try to understand why this techniques work and apply them to a virtual environment.

1.1. Lighting Design

Lighting design is the process of finding light positions and parameters that create the lighting configuration to achieve the desired visual properties of a scene. It is an highly application dependent process: in medical visualization we may want to achieve good perceptual qualities while in a movie scene we may want to evoke a specific emotion in the audience.

In the diverse fields that make use of illumination, the traditional approach towards lighting design uses typically a *direct design* paradigm, where the designer specifies all the lights and the required parameters iteratively, then observes the rendered scene, evaluates the results and makes the necessary modifications based on the observation. It is a trial and error approach with active participation of the designer, thus very tedious, time consuming and requiring expertise in the field.

[1]In a digital camera, the most commonly used nowadays.

Another possible approach is based on a *inverse design* paradigm. The designer is presented with some interface allowing him to specify a set of goals for the scene, and then the system finds the lighting parameters that suits best the desired goal. This approach may be faster and less tedious, but still requires the designer to have expertise and the ability to articulate what is the desired appearance in the specific system language.

1.2. Proposed Approach

One of the known inverse lighting design approaches is *lighting by example*, in which the scene is illuminated based in some example input, usually a 2D image or a 3D scene. This work will focus in this type of lighting design, trying to replicate the lighting shown in the inputs using the cinema methods studied, with the objective of recreating the same mood. The possible inputs will be not only images, but also short videos. There are some extra possibilities to explore: if the video has moving lights, the system may also mimic that movement in the virtual scene. The video analysis component will have a premeditated limitation: the input video must be representative of a single camera shot, i.e. with no cuts.

The main hypothesis of this work is the following: *"Is the mood of the input images close to the mood of the correspondent automatically illuminated 3D environment?"*.

To answer this question, the system will be evaluated by users that will classify the mood of a set of preselected input images and the correspondent three-dimensional scenes. The testers will not be aware of the existent relations between the images. The classification will be accomplished presenting the testers with a list of mood defining adjectives and asking them to quantify each adjective recurring to a Likert-type scale. The evaluation of the system when the inputs are videos is left as future work.

1.3. Objectives

In summary, the main goals of this work are:

- Identify lighting setup techniques used in cinema (positioning, relations between each light, relations between lights and the subject in focus, etc.) and understand their influence in the mood of a scene.
- Develop an inverse lighting design method that uses both images and short videos as examples for light placement and parametrization on a 3D scene, making use of the cinema techniques identified.
- Take advantage of the time dimension of videos and make lights in the virtual scene move in the same way when the input is a video.
- Implement a software system to demonstrate the developed method, evaluating its results with users, rating the mood of the input and the resulting 3D scene.

2. Background

Inverse lighting design methods try to make the designer focus on the goals and/or the restrictions for lighting a specific scene, rather than the intricate details of a complete lighting configuration. Inverse lighting can be considered a sub-field of inverse rendering, which is a broad topic and outside of the scope of this work. Patow and Pueyo [1] provide an excellent survey on inverse rendering.

There are some recognized variants of inverse approaches to lighting:

- Architectural inverse lighting, trying to remove iterations from the task of designing the desired lighting for architectural spaces. Tools of this type try to accomplish a specific lighting configuration that suits the needs of the room in question. Possible examples are present in Computer Assisted Design (CAD) software.
- Perceptual inverse lighting, emphasizing the perception aspect of the final rendered scene or object. For example, medical visualization tools focus in maximizing shape, detail and depth perception, directing viewers' attention to important details. This variant is focused on eliminating the need to design lighting in domains where lighting design is a secondary need rather than a central focus.
- Lighting by example, in which the systems take as input some kind of example for lighting and try to replicate it in a 3D environment. The most commonly used examples are 2D images and 3D representations of scenes.
- Interactive inverse lighting, creating lighting designs for three-dimensional scenes in real-time, in order to convey emotion and control tension, usually reacting to some kind of human interaction. Examples are found implemented in videogames or storytelling software.

The next sections will refer some of the most prominent works belonging to each of the enumerated variants.

2.1. Architectural Inverse Lighting

Kawai, Painter et al. [2] presented one of the first inverse lighting works. This work find light settings for real-life environments using architectural visualization tools and radiosity based renderers. The user manually specifies light positions and numerically defines goals for the lighting design. Some of this goals are literal in nature, such as the desired intensity at a specific point of the scene being illuminated. Other goals are subjective, like “pleasantness”, “privateness” or how “spacious” the room must feel. The system, given the goals, finds angles and intensities for the manually positioned lights.

Costa, Sousa et al. [3] used a similar approach and it is recognized to be the most complete approach of architectural lighting design. Their approach makes three assumptions about an environment. First, surface reflectance should be described with Symmetrical Bi-Directional Reflectance Distribution Functions[2]. Second, the environment cannot have participating media[3]. Third, the light model used in rendering obeys the photon nature of light. With these assumptions, this system allows designers to specify almost any illumination goal using scripting. For example, the designer can code a rule that places a constraint on light intensity on a specific surface. Using global illumination rendering algorithms, this system finds the correct light sources and parameters that fulfil all the defined rules.

Schoeneman et al. [4] also address lighting design as an inverse problem. Designers specify a set of desired lighting effects that are expected to appear in the final image, and the system tries to find a solution in which the lighting is closest to the desired goal. The user then paints in the surface of the rendered scene causing a change in the surface radiance functions. These after-painted radiance functions are used as target values in the optimization process that follows. The lighting parameters that are optimized in this approach are light intensities and colours. Light positions are not optimized because the authors claim that users know where to put the lights, but not how to combine them in terms of colours and intensities.

Poulin and Fournier [5] developed a framework for inverse design of light positions through the specification of shadows and highlights in a 3D scene. Users use a mouse pointer to sketch the desired shadow areas, and an objective function is defined such that the shadow region for a point light bounds the sketched regions as tightly as possible.

2.2. Perceptual Inverse Lighting

Shacked’s system [6] was one of the first designed to automatically light objects using a perceptual quality metric. Given a model of a 3D object, its material properties and the desired viewing parameters, the system determines the values of various lighting parameters. Shacked denominated the objective function a *perception-based image quality objective function*, and designed it to quantify the extent to which an image of a 3D object succeeds in communicating scene information, such as the 3D shape, fine geometric details and spacial relationships between objects in the scene. The function consists of an histogram normalization term and an average image gradient. The average image gradient represents the mean change of pixel intensity from one pixel to the next and tries to indicate the visible surface detail. The histogram normalization term represents the uniformity of the overall colour distribution in an image and also represents the overall visibility of an object. Using the geometry of the 3D model, the system derive where silhouette edges should be in an image for a given point of view. An edge detector filter is used to determine where edges appear to be visually. The ratio between the actual edges and visible edges is then used to quantify the quality of shape depiction for a given light configuration.

Gumhold [7] also proposed a perceptual inverse lighting solution. The developed cost function is based on Shannon’s entropy metric [8], a spatially independent term that quantifies the illumination entropy of an image. This term, in practice, provides similar data to the Shacked’s histogram normalization term.

Vázquez and Sbert [9] extended Shacked and Gumhold’s work in some points. Instead of analysing images in a grey-scale format, their system operates on the colour domain, using a distance metric on the CIE 1976 colour space. They also take into account the spatial distribution of pixels when calculating the entropy, instead of considering it as a function of pixel intensities in a spatially independent fashion. More specifically, entropy is computed as a function of contiguous areas of colour across the image.

Vázquez and Gumhold’s approaches used only one light source to light scenes or objects. Shacked’s system could work with multiple light sources, but the test users reported that it was generating confusing results, given the assumptions of the system algorithms.

Lee, Hao et al. [10] proposed a method where an object is separated into areas of local curvature based on its geometry. Then, each identified area was illuminated with a dedicated light source in order to maximize perception. To finalize, the system adds silhouette lighting to enhance the object silhouette and proximity shadows for better depth perception between portions of the object.

[2]SBRDF is a four-dimensional function for which an inverse can be computed. The function defines how light is reflected at an opaque surface, mapping incoming light to outgoing light.

[3]The existing particles in the atmosphere, like dust and water, absorbing and scattering light.

2.3. Inverse Lighting by Example

Ha and Olivier [11] presented an approach to lighting by example based on 2D images. In wavelet based lighting by example, images are used as exemplars (targets) for lighting 3D scenes. Lighting effects are seen as a distribution of different light intensity levels over different locations in images. A wavelet transform captures information about the distribution of energy of an image function $I(x, y)$ over different frequencies at different locations. A transform aims to represent a function by a set of basis functions. Thus, an image can be represented by a set of basis functions, and each basis function is weighted by a coefficient. Applying the same set of basis functions to different image functions results in different sets of coefficients. Each wavelet transform of an image function is defined by a set of normalized energy spectrum coefficients. At the end of image analysis, the system has a set E of normalized energy spectrum coefficients for a wavelet transform of the image function $I(x, y)$:

$$E = \{_i E^{a,b}\} \qquad i = 0...N-1 \tag{1}$$

Where N is the number of basis functions.

The optimization process tries to make the set of coefficients of the rendered image as close as possible to that of the example image. The objective function used in the optimization process uses the Euclidean Distance to measure the distance between two sets of energy coefficients. Using optimization schemes like steepest descent, genetic algorithms and simulated annealing, a solution that minimizes the differences between the example image and the rendered scene will be found. Its optimality depends on the scheme used and its working parameters.

2.4. Interactive Inverse Lighting

El-Nasr and Horswill [12] proposed Expressive Lighting Engine (ELE), an automatic and intelligent lighting control system. ELE uses scene graph information, location of characters, information of light emitting objects in the scene and artistic constraints to create a lighting design, which is adjusted and manipulated dynamically. ELE selects the number of lights, their locations, types, angles and colours. To accomplish this, cinema and theatre lighting design rules are formulated mathematically into an optimization function. Internally, the system divides the scene into n different areas and categorizes those areas as focus, non-focus or background. This information is then used to identify focus, increase depth and contrast based on the specified designer goals. By using a multi-objective function, the number of lights to use for each area is defined, and using a non-linear optimization system based on hill climbing, an angle for each light is computed. 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 colours, non-focus colours, and background areas to select a colour for each individual light in the scene. The multi-objective cost function evaluates the colour against the lighting-design goals including: establishing depth, respecting a colour style, maintaining dramatic tension, adhering to desired hue, saturation, and luminance, and maintaining visual continuity.

The Expressive Light Engine has been extended and tested by the authors in interactive applications, trying to project tension in virtual environments, more specifically in games [13] [14]. ELE have also been used to enhance gaming experiences and direct the players attention to important events in first person shooters by lighting the scenes dynamically [15] [16].

Zupko and El-Nasr developed a System for Automated Interactive Lighting (SAIL) [17] whose goal is to adapt to changes in context given a designer aesthetic and functional intent. SAIL tries to achieve the best possible compromise between what the lighting should naturally look like in a given context, and what the lighting should ideally look like given a designer's goal. The design of SAIL was motivated by the shortcomings of ELE, particularly the restrictions created by its numerical constants, often interdependent and hard to understand. SAIL's first component is image analysis, using image processing to understand a goal image with shading and colour contrast information. The second component of SAIL is object understanding, used to achieve adaptive runtime lighting. This component uses metrics extracted from a goal image and the position of light sources in the environment. It then lights the object 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 image analysis phase on SAIL extracts light direction and light contrast from any 2D image. The detected light direction correlates to the direction of key light and the detected light contrast correlates to the intensity of fill light, roughly. Before any processing, each colour channel of the image is filtered with a Gaussian kernel to remove high frequency information. Then, the filtered image is transformed into a grey-scale image using the Y value from the XYZ colour space [18]:

$$Y = 0.2126R' + 0.7152G' + 0.722B' \tag{2}$$

where Y is the Y term of the XYZ colour space conversion from the sRGB colour space. R', G' and

B' are defined by the function:

$$C' = \begin{cases} \left(\dfrac{C+0.055}{1.055}\right)^{2.4} & C > 0.04045 \\ \\ \dfrac{C}{12.92} & else \end{cases} \tag{3}$$

with C substituted for either R, G or B.

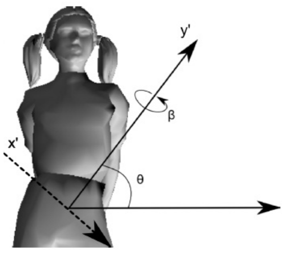

Figure 1: Approximate meaning of the two direction metrics extracted from images.

With the generated grey-scale image , two metrics for light direction are calculated, which roughly correspond to the angles shown in Figure 1. To find θ, 3×3 windows of pixels are analysed across the image, similar to the application of a convolution kernel. For every window, the center of mass of pixel intensity is found and converted to a normalized 2D vector. The average vector for the whole image is converted to an angle to find θ.

The term β as seen in Figure 1 is computed by finding the average center of mass of pixel intensities along the axis x':

$$\beta = \frac{\sum_{i=1}^{n} \left(\dfrac{1}{\sum_{j=1}^{m_i} Y_{i,j}} \sum_{j=1}^{m_i} j Y_{i,j} \right)}{n} \tag{4}$$

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 $Y_{i,j}$ is the Y value of the j^{th} pixel of the line i.

The contrast of the image is directly quantified by the light entropy metric of [7], but is also a factor of β. As β increases, contrast tends to increase and entropy decreases, because of sharper shading gradients present in the image.

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 lighting model: camera direction, key direction and key to fill ratio. For each randomly sampled parameter set, the object is rendered. The rendered image is then analysed by the image analysis component, and a set of image metrics is produced.

In runtime, the system understands the appearance of a model based on the jitter sampled described above. It also understands a goal image through image analysis. 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. To measure closeness, a least squares error metric is used.

3. Implementation

To implement the desired functionalities, a modular system was designed with a pipeline approach in mind. Two major components are present: the Input Analyser and the Renderer. The first has the function of analysing the inputs given by the human user (images or videos) and sending the results to the Renderer. When receiving the results, the Renderer will mimic in the 3D scene the lighting detected by the Analyser.

The Input Analyser is implemented as a separate entity from the Renderer and can receive either an image or a video as input. The processing will start on distinct pipeline stages, depending on the case. When the input is a video, the Video Processor has the task of feeding the Image Processor with a stream of frames for individual analysis. The time difference between each frame of the stream can be chosen by the user through an input parameter. The Image Processor is the core of the E-Lumination system. In this module, the lighting information is extracted from each received image, coming either from a stream of frames or from a single image received from user input. There are three main lighting aspects extracted: Direction, Entropy and Colour. Based on the identified aspects, a lighting setup is selected and its parameters are set. Then, all this information is compiled in XML format, ready to be sent to the Renderer. The Renderer, as the name implies, shows a 3D scene using the lighting setup and parameters chosen by the Analyser, and also supports some simple user interaction, allowing lighting setup changes, parameter tweaking and camera movement. A diagram of the E-Lumination system conceptual model is available on Figure 2.

The following sections will explain the implementation details of the each sub-system composing E-Lumination, splitting each module and individual step into a section.

3.1. Input Analyser

The Input Analyser is an independent entity and the starting point of the pipeline. For this reason, this subsystem is implemented in a separate pro-

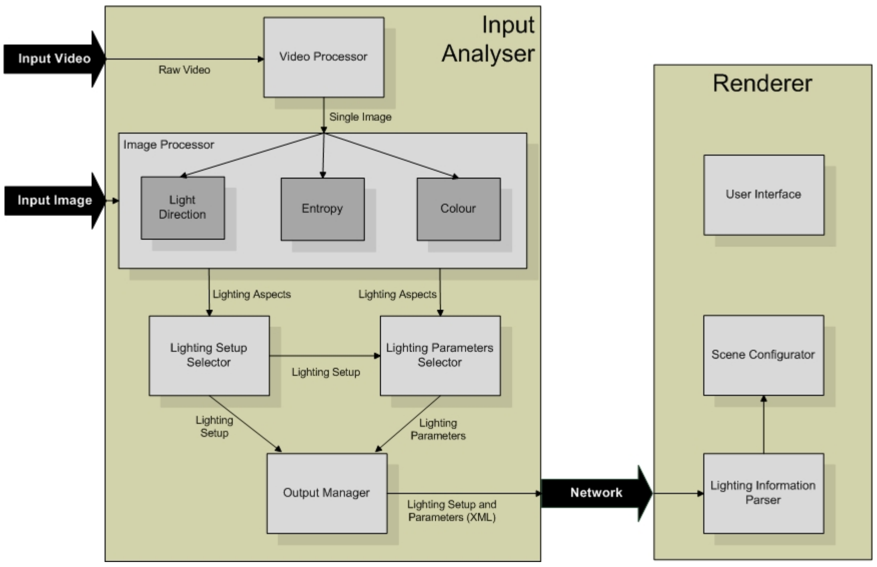

Figure 2: System Conceptual Model.

gram with the C++ native language, arguably exchanging development time and simplicity for runtime speed. The Input Analyser is composed by several modules, the algorithms of each module will be exposed separately in the next sections.

3.1.1 Video Processor

The Video Processor has the single function of decoding the input video using the installed codecs in the operating system and send each raw frame to the Image Processor. If the user wants to avoid the analysis of all the frames of the video, this module prompts the input of a time step in milliseconds (s_{ms}). This time step is used to skip the correct number of frames according to the frames per second presented by the video.

3.1.2 Image Processor

The core operations of the Input Analyser are contained in the Image Processor where several image processing algorithms are applied to the input images. For this reason, the industry standard Open Source Computer Vision Library (OpenCV)[4] is used whenever possible to ease the development process.

The used algorithms are applied sequentially in pipeline fashion as illustrated in Figure 3. The next paragraphs will detail each one of the steps of the pipeline.

Gaussian Filter. This step applies a Gaussian filter also known as Gaussian blur, a typical filter smooth operation. Using a matrix defined by a Gausian Distribution creates a smoothing filter where the pixel located at the middle have the biggest weight, and the weight of his neighbours decreases as the spatial distance between them and the center pixel increases. Applying a Gaussian Filter as a first step of image processing is effective at removing noise, consequentially improving the next steps.

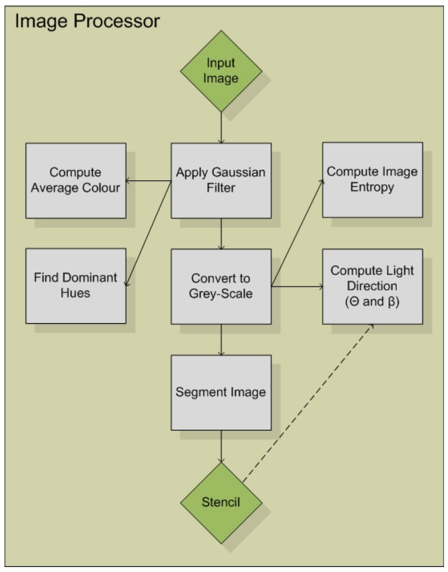

Figure 3: Image Processor Flow.

Convert to Grey-scale. Converting the input image coming from the Gaussian Filter stage to a single channel grey-scale is a necessary step to the next operations. All the pixels of the input image will be represented by an integer value in the interval $[0, 255]$. The method used to achieve this result is similar to the method described in Joseph Zupko's work, SAIL (Section 2.4). As described,

[4]http://opencv.org

the input image is converted to the XYZ colour space [18] and only the Y value is used in the final result. Equations 2 and 3 explain this process. The final single channel image is optimal for lighting extraction and passed to the next stage.

Segment Image. The main purpose of the segmentation is to identify and separate the background and the foreground of the input image. This avoids the analysis of lighting on non-relevant and sometimes misleading areas of the image. The input of this stage is the grey-scale image from the last stage and the result is a binary image, where black represents background and white represents foreground. This binary image will be used as a stencil in the next stages. The thresholding technique is used to perform the segmentation. This method uses a threshold value a_{th} to separate the pixel values in two classes, depending upon the value a_{th}.

The segmentation result is shown to the user with the stencil blended with the original input image. The user is free to correct or enhance the automatic segmentation by changing the threshold value with a slider and observing the changes in real-time.

Calculate Light Direction. The direction is one of the identified light aspects of the Input Analyser. This stage algorithms are based on SAIL (Section 2.4). Two direction metrics are computed, θ and β (Figure 1). Both are based on the center of mass of pixel intensities. The algorithms are only applied to pixels marked as foreground on the stencil image generated previously.

Calculate Image Entropy. Information entropy is defined as the uncertainty on some set of data. When the data is a set of pixels in a monochromatic image, uncertainty means the presence of many different grey-scale tones. Knowing this, the image entropy is a good metric for image contrast. The contrast is bigger as the entropy decreases (there are less variety in pixel values), and vice-versa. We chose an approach based on Gumhold's work [7], represented by Equation 5.

$$H = \sum_{i=1}^{m} p_i \log \frac{1}{p_i} \tag{5}$$

This stage returns a normalized entropy value in the interval $[0, 1]$.

Calculate Average Colour. The Analyser feeds the Renderer with a set of hints recommending the most suitable colours to apply to the lights of the setup. One of the recommended colours is the RGB average computed from the input image. This stage simply runs through all the pixels of the image and calculates the average for each of the RGB channels independently. The grey or black pixels are ignored and do not take part on this calculation, avoiding averages with excessive darkened colours or even black.

Find Dominant Hues. Besides the average colour, the Analyser also sends to the Renderer a set containing the five dominant hues present on the input image. To find the dominant hues, the input image is converted to the HSV colour space. Then this image is split into three monochrome images, each one representing a HSV channel. An histogram is built for the image representing the Hue channel, and its five biggest peaks are extracted. To prevent cases where the peaks are on very similar hues, a small margin around each peak is created, avoiding the selection of other peaks on the close surroundings.

3.1.3 Lighting Setup Selector

This stage of the Analyser has the function of selecting the adequate lighting setup according to the detected lighting aspects. At the time of the writing, the selector will choose between the Three Point Lighting and Cross Keys setups. The selection is made according to the extracted image entropy. The Cross Keys setup is selected when the image has low contrast, represented by an entropy greater than 0.6[5]. Otherwise, when the entropy is less or equal to 0.6, Three Point Lighting is selected, having its parameters set in the next stage. This is a stage with much space for improvements in the selection rules and in the number of supported setups.

3.1.4 Lighting Parameters Selector

When a setup is chosen, its specific parameters must be set. The Three Point setup needs to adjust the ratio between the key and the fill lights. This value is computed based on the entropy extracted by the Image Processor. The entropy value on the interval $[0, 1]$ is inversely mapped to the ratio value in the interval $[0.5; 5]$, i.e. 0 maps to 5 and 1 maps to 0.5. The Cross Keys setup does not receive any parameters.

3.1.5 Output Manager

When all necessary data is created by the Input Analyser, the job of the output manager is to compile and send it to the Renderer subsystem, using a defined protocol. All the information and the respective meta-data is compiled into a human-readable XML format.

The Analyser will try to connect to the Renderer via a TCP network socket, and when the connection succeeds, the whole XML string is sent.

[5]Entropy values come from the Image Processor in the $[0, 1]$ interval.

3.2. Renderer

The Renderer is an independent sub-system implemented under the form of a stand-alone application using the Unity[6] game engine. This sub-system has the responsibility of rendering a 3D scene according to the data received from the Input Analyser, and responding to some simple user interaction.

On startup, the Renderer will present a default scene and start listening for communications from the Input Analyser. When a connection is established, the XML data is received, parsed and ready to be set in the scene through an user clickable button.

The user is able to manually change a set of parameters through the user interface, like move the camera, change or tweak the lighting setup, etc.

When the received XML is the result of a video analysis, each result will be applied to the scene sequentially with the delay shown in the attribute "DeltaTime". To smooth the transitions and movements between analysis results, the values of the lights θ and β angles and the frame entropy are interpolated. To interpolate the numerical data, *spline interpolation* is used.

4. Results

Since the evaluation requires mood classification, a testing approach based on opinions of human users was chosen. The main goal of the evaluation with users was to find an answer the following question: *"Is the mood of the input images close to the mood of the correspondent automatically illuminated 3D environment?"*

A model query was created, containing fifteen images, each one with a set of mood defining adjectives to quantify as described below. The query's images are photographs, movie scenes and screenshots of a 3D scene render. The photographs and the movie scenes were used as input to the E-Lumination system, and the 3D renders are snapshots of the resulting automatically illuminated scenes. The existent relationship between the images on the query is not known nor evident to the testers.

The moods of both the images and the virtual scenes were classified recurring to a list of mood defining adjectives contained in six categories. This list takes inspiration from the dimensionless Hevner's Adjective Circle [19], originally developed to classify moods that can be evoked by music. The adjectives arranged by categories are:

1. Sad, Tragic, Melancholic, Depressing, Gloomy, Dark

2. Dreamy, Imaginary, Tender, Sentimental, Longing

3. Calm, Serene, Tranquil, Quiet, Soothing

4. Happy, Bright, Merry, Joyous, Cheerful

5. Exciting, Agitative, Impetuous, Dangerous, Thrilling

6. Scary, Frightening, Terrifying, Alarming, Shocking

The testers rated each one of the six mood categories using a Likert scale [20], with five options to choose from. The options are: "Strongly disagree", "Disagree", "Neither agree nor disagree", "Agree" and "Strongly agree". The queries asked to rate the mood of the shown images, no request was made to take the illumination or any other factor into special account.

Five different queries were derived from the model query, all with the same set of 15 images. Each one has its images in distinct and random order, with special care being taken to not having a particular image in the same position in two queries. This helps to minimize the bias and influence that a particular sequence of images may have on the testers.

Each tester answered to a single query online and anonymously. All of the five queries, served in round-robin fashion are available online[7].

Thirty-one query answers were registered, distributed by the five questionnaires in the following way: Queries 1, 2 and 3 had six responses each. Query 4 had five responses and query 5 registered eight responses.

To perform a comparison between the inputs and the resulting scenes ratings, a Mann-Whitney U test[8] will be run on the ratings of the adjectives divided on two groups: "Input" and "Result", representing the rating for the input image and the rating for the resulting scene, respectively. The null hypothesis of the test is: "*The ratings of the inputs and resulting scenes have identical distributions*". Since the hypothesis in test is an equality, we are before a bilateral test and the relevant p-value will be two-tailed. A significance level of 0.05 will be used, meaning that all p-values less or equal than 0.05 will reject the null hypothesis.

Figure 4 summarizes the obtained results for the first six image sets, showing the resulting U and p values for all the Mann-Whitney tests performed. All the p-values smaller or equal to 0,05 representing the rejected null hypothesis are marked in red, the remaining are marked in green. Knowing that the U values for this population fit at the range $[0; 480.5]$, a blue bar is representing where each of

[6]http://unity3d.com

[7]http://web.ist.utl.pt/ist157944/e-lumination/query.php

[8]Also known as Wilcoxon rank sum test.

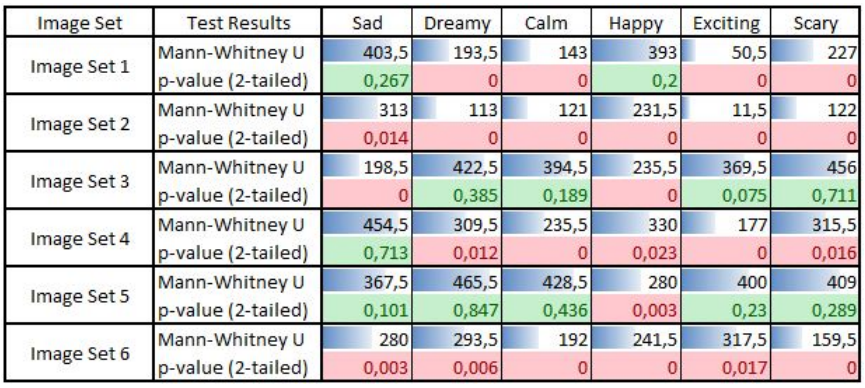

Image Set	Test Results	Sad	Dreamy	Calm	Happy	Exciting	Scary
Image Set 1	Mann-Whitney U	403,5	193,5	143	393	50,5	227
	p-value (2-tailed)	0,267	0	0	0,2	0	0
Image Set 2	Mann-Whitney U	313	113	121	231,5	11,5	122
	p-value (2-tailed)	0,014	0	0	0	0	0
Image Set 3	Mann-Whitney U	198,5	422,5	394,5	235,5	369,5	456
	p-value (2-tailed)	0	0,385	0,189	0	0,075	0,711
Image Set 4	Mann-Whitney U	454,5	309,5	235,5	330	177	315,5
	p-value (2-tailed)	0,713	0,012	0	0,023	0	0,016
Image Set 5	Mann-Whitney U	367,5	465,5	428,5	280	400	409
	p-value (2-tailed)	0,101	0,847	0,436	0,003	0,23	0,289
Image Set 6	Mann-Whitney U	280	293,5	192	241,5	317,5	159,5
	p-value (2-tailed)	0,003	0,006	0	0	0,017	0

Figure 4: Summary of all the performed Mann-Whitney tests.

the U values fit in this range, facilitating visual analysis.

5. Conclusions

The development of E-Lumination required the research of lighting methods and techniques. Most of the available cinematography literature explains clearly what are the objectives of a good lighting setup and what kind of tools and techniques exist to achieve it. This information provided a good basis for the implementation of the lighting setups and their respective parameters in the graphical renderer.

We successfully modelled and developed the proposed system, capable of taking as input images and videos, and then generating the correspondent lighting setup for a three-dimensional scene. When the input is a video, the system is capable to replicate its temporal dimension, moving and changing the lights synchronously with the input video. Even though this feature was not part of the performed evaluation, a demo video demonstrating it is available[9].

To analyse the input images and videos, the implementation applies a sequence of standard algorithms of computer vision and image processing, many with specific adaptations to the domain of the problem. Great inspiration was taken from the existing inverse lighting systems, especially Joseph Zupko's SAIL [17]. One of the biggest advantages of the developed approach, when comparing to others with similar goals, is the possibility of using any image as input without any restrictions like size, aspect-ratio, colours or presence of chroma key. When using videos for input, it is recommended to use videos with a single camera shot, with no abrupt transitions.

Analysing the statistical results of the query data it is visible that four of the six image sets had at least one adjective with similar ratings (Sets 1, 3, 4 and 5). Of these, Sets 3 and 5 had most of their adjectives rated in the same way. Sets 2 and 6 had all of their adjectives rated in non-similar ways.

It is noticeable in the results the influence of facial expressions, body language and possibly previous memories of a specific movie scene. For example, in image sets 1 and 2 most people rated the inputs as exciting, arguably for the aforementioned reasons. For those sets, the resulting scenes were rated in a very different way for the adjective "Exciting" leading to the two most dissatisfying results.

Even with this external "noise" inherent to queries with humans, having two image sets sharing similar ratings in most of the adjectives cannot be attributed to randomness. In a top-view of the statistical data, it is observable that a third of all the adjectives ($\frac{12}{36}$) were rated similarly. These obtained results can be considered satisfactory, taking into account that the environment, characters and camera angles are not replicated in any way from the input image or video.

The results confirm the heavy influence lighting has in the perception of the visual mood and ambiance of a scene. Also, like much of related work in inverse lighting, shows that the inverse lighting by example approach is a valid one. Especially for inexperienced or amateur designers where the advantage of bootstrapping the illumination of a scene is very relevant, instead of starting from scratch.

Observing the statistical results of the Extra Set, there exists enough evidence to conclude that there is no significant difference between the ratings of the three images. This leads to the conclusion that the camera angle affects negligibly the rating of the mood, for the particular scene used in all image sets.

6. Future Work

There are some areas with room for improvement, possibly making the system function better as a whole. The following points will enumerate them.

- The Image Processor is the core of the E-

[9]`http://web.ist.utl.pt/ist157944/e-lumination/demo.avi`

Lumination system, and any improvement or experimentation has the possibility of changing drastically the performance of the whole system. The processor analyses three aspects of the input: Light Direction, Entropy, and Colour. Other light aspects like Hardness/Softness and Texture could also be detected and analysed by this module. With this new aspects, different parameters for the lighting setups can be created and set, in attempt to replicate them.

- Still in the Image Processor, the method used for the segmentation of the input images can be improved and further experimented. The possibility of using more computationally expensive methods, capable of much better segmentations, (usually time consuming and with relatively complicated user input) should be considered. A better segmentation will generate a better quality stencil to be used on the image analysis, improving greatly the detection of the light direction. It is important to evaluate the trade-off between the quality of light aspects detection and the speed of the input analysis stage.

- The module that performs the selection of light setups have two possible options: Three Point Lighting, and Cross Keys. Having a greater number of selectable lighting setups can be an advantage when trying to replicate certain aspects detected by the Image Processor. Even custom setups created on-the-fly adapted for the specific input can perform better than a predefined setup. On the other side, this would significantly increase the complexity of the rules needed to select or create the lighting setup to use. Some more complex setups may even require the Image Processor to identify more light aspects to provide the Setup Selector the necessary data to perform a decision.

- The analysis of videos can be greatly improved. One of the main problems are the occasional large lighting differences detected between subsequent frames. This behaviour is aggravated when the analysis is made in all the frames of the video (without skipping). This leads to jittery results on the 3D scene, where the lighting parameters are constantly changing, sometimes abruptly, when interpolation is not being used, and masking the problem. The segmentation stage is one of the main originators of this behaviour, since a small change in the generated stencil comparing to the previous frame have a big influence on the light direction detected on the actual frame.

- On the Renderer side, the adding of shadows may prove to be a major improvement. It is well known the effect of shadows in mood perception by humans. The use of shadows on the 3D scene would require a very accurate detection of light direction in the Image Processor, since any minor flaw would be much more noticeable.

Not only the system, but also the methodology used for the evaluation can be improved, minimizing the noise from external variables affecting the rating of the images. This would generate better data from the evaluators, and possibly better conclusions.

- Adding different environments may help to minimize the bias experienced by the testers, when rating a snapshot of the 3D scene. Also it would eliminate some of the boredom experienced when rating the "same" image many times, even though the lighting and camera angles are obviously different.

- A change of the characters present on the scene may also influence and modify the ratings for the same reasons of the previous item. Even more important than the character itself, is the facial expression, as it affects greatly the recognized mood.

- The list of adjectives used for user rating is a good point for further research. We tried to accomplish a middle ground in the number of used adjectives, but there are models with much more mood defining adjectives, and others that defend a smaller set with almost no redundancy and confusion between them.

- The developed set of tests only evaluated the system when the given inputs were images. Altough the analysis of videos can be greatly improved, it is functional and preliminary evaluation with users can also be conducted.

References

[1] Gustavo Patow and Xavier Pueyo. A survey of inverse rendering problems. *Comput. Graph. Forum*, 22(4):663–688, 2003.

[2] John K. Kawai, James S. Painter, and Michael F. Cohen. Radioptimization: goal based rendering. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 147–154. ACM, 1993.

[3] António Cardoso Costa, António Augusto Sousa, and Fernando Nunes Ferreira. Lighting design: a goal based approach using optimisation. In *Proceedings of the 10th Eurographics conference on Rendering*, EGWR'99, pages 317–328. Eurographics Association, 1999.

[4] Chris Schoeneman, Julie Dorsey, Brian Smits, James Arvo, and Donald Greenberg. Painting with light. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 143–146. ACM, 1993.

[5] Pierre Poulin and Alain Fournier. Lights from highlights and shadows. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, I3D '92, pages 31–38. ACM, 1992.

[6] Ram Shacked and Dani Lischinski. Automatic lighting design using a perceptual quality metric. *Computer Graphics Forum*, 20:2001, 2001.

[7] Stefan Gumhold. Maximum entropy light source placement. In *Proceedings of the conference on Visualization '02*, VIS '02, pages 275–282. IEEE Computer Society, 2002.

[8] Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

[9] Pere-Pau Vazquez and Mateu Sbert. Perception-based illumination information measurement and light source placement. In *Lecture Notes in Computer Science*, page 316, 2001.

[10] Chang Ha Lee, Xuejun Hao, and Amitabh Varshney. Geometry-dependent lighting. *IEEE Transactions on Visualization and Computer Graphics*, 12(2):197–207, March 2006.

[11] Hai Nam Ha and Patrick Olivier. Lighting-by-example with wavelets. In *Proceedings of the 8th international symposium on Smart Graphics*, SG '07, pages 110–123. Springer-Verlag, 2007.

[12] Magy M. S. Seif El-Nasr. *Automatic expressive lighting for interactive scenes.* PhD thesis, Evanston, IL, USA, 2003.

[13] M. Seif El-Nasr, S. Niedenthal, I. Kenz, P. Almeida, and J. Zupko. Dynamic lighting for tension in games. *Game Studies Journal*, 7(1), 2006.

[14] Magy Seif El-Nasr. Projecting tension in virtual environments through lighting. In *Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, ACE '06. ACM, 2006.

[15] Magy Seif El-Nasr, Joseph Zupko, and Keith Miron. Intelligent lighting for a better gaming experience. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1140–1141. ACM, 2005.

[16] Magy Seif El-Nasr, Athanasios V. Vasilakos, C. Rao, and Joseph A. Zupko. Dynamic Intelligent Lighting for Directing Visual Attention in Interactive 3D Scenes. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(2):145–153, 2009.

[17] Joseph Zupko and Magy Seif El-Nasr. System for Automated Interactive Lighting (SAIL). In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, pages 223–230. ACM, 2009.

[18] T Smith and J Guild. The C.I.E. colorimetric standards and their use. *Transactions of the Optical Society*, 33(3), 1931.

[19] Kate Hevner. Experimental studies of the elements of expression in music. *The American Journal of Psychology*, 48:246–268, 1936.

[20] R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):1–55, 1932.