Global Illumination and Image Appearance

Vincent Léon

Master of research intership report

Global Illumination and Image Appearance

Author:
Vincent Léon

Supervisors:
Kadi Bouatouch
Rémi Cozot
FRVsense

UNIVERSITÉ DE RENNES 1

Abstract

In Computer Graphics, light plays an important role in the appearance of a scene. Choosing a different light configuration can lead to many different aesthetics for the final image. Lighting design is even more complex when using sophisticated global illumination rendering methods. In this report, we present a new approach to automatically design a lighting configuration according to the lighting goal specified by the user. The lighting goal is expressed as a set of target parameters. These target parameters are used to set up an objective function such that, when the function is minimized, the aesthetics of the image rendered using these optimal light parameters meet the requirements set by the user. Our results show that this method can be used to automatically design a lighting configuration that will give to the final image a classic photographic aesthetics, such as high-key or low-key aesthetics.

Keywords: inverse lighting, global illumination, aesthetics, optimization

Acknowledgments

I would like to thank my supervisors, Kadi Bouatouch and Rémi Cozot for giving me the opportunity to work on this subject, for their guidance, for their advice and for proof-reading this report.

I am also grateful to Mickaël Ribardière and Adrien Gruson, members of the FRVsense team, for their advice on using the `mitsuba` rendering platform.

I would like to acknowledge the IRISA laboratory staff as a whole for the rigorous yet friendly atmosphere I worked in.

Finally, I would like to thank my fellow interns for the many talks and laughs we had together.

Contents

1 Introduction

1.1 Background

In computer graphics, there is a set of sophisticated rendering methods that simulate all the interactions between light and the geometry in a 3D scene. These are called global illumination methods. They take into account multiple phenomena such as reflection, refraction and scattering, as shown in figure (1). These interactions have been formulated as an integral equation by J. Kajiya in [5].

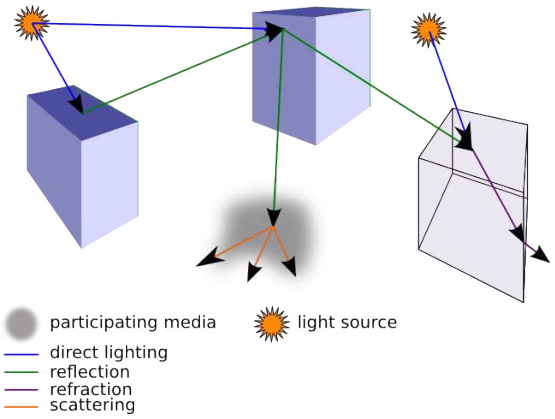


Figure 1: Reflection, refraction and light scattering in a 3D scene

The solution to this integral equation is the luminance function that gives for each point of the 3D scene its luminance. Once the integral equation has been solved for each point of the scene, an image is rendered using a virtual camera: for each pixel on the sensor of the camera, we evaluate the contribution of the point of the scene as seen through the pixel, as show in figure (2).

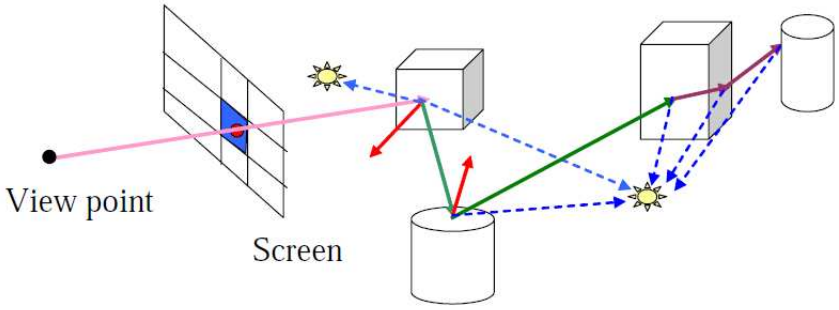


Figure 2: Rendering with a virtual camera, colors are the same as figure (1)

However, since many phenomena are accounted for by global illumination algorithms, it is difficult to know beforehand what the resulting image will look like. For instance, we would like an object of a scene to be bright, with a lot of contrast, while we would like the background of the scene to be dark and uniform. Answering such specifications can be a time-consuming and tedious effort if it is not done automatically. Thus, a trial-and-error approach is not satisfactory.

To obtain a satisfactory result automatically, the user has to express his aesthetic intent by providing a set of target values for properties of the final image. The goal is to change iteratively the parameters of the 3D scene to fit the specification. There are a lot of parameters to take into account. Among them are:

- Position, size, luminance of the light sources,
- Reflectance of the materials,
- Distribution of the objects in the scene.

The methods used to compute these lighting parameters to fit a set of constraints are called inverse lighting methods. In these methods, an objective function that accounts for the target parameters is defined to express the intent of the user. The objective function expresses the distance between the current lighting design and the desired result. To find the optimal set of parameters, an objective function has to be minimized.

There are many ways to express the aesthetic intent, as there are many possible aesthetics. In this Master report, we will mainly address two target aesthetics used in films and photography: high-key and low-key pictures. High-key and low-key aesthetics focus on one object of the scene, called henceforth *main object*.

(a) High-key image (b) Low-key image

Figure 3: High-key and low-key images, courtesy of Olivier Chauvignat

High-key images, such as the figure (3a) are bright, do not have too much contrast, nor feature dark shadows cast by the main object. Low-key images, such as the figure (3b) have a darker tone, their contour lines are highlighted and the background is usually black. They also feature high values of contrast.

We designed a new inverse lighting framework. Given a set of target parameters provided by the user, we set up a platform that can render a scene with global illumination techniques and allows to minimize the objective function to find the desired parameters (light source size, position, luminance) to meet the user's intent.

In our case, we use two light sources: a key-light and a fill-light. The key-light is the main light source in the scene. It is used to light the main object. The fill-light is used to control the shadows cast by the main object. The key-light sets the main direction of lighting, and the fill-light lights the main object from a side angle relative to the key-light.

1.2 Contribution

The objective functions are usually defined for a single purpose, such as in [12] where the goal is to make the final image easy to understand. Our goal is to have a broader range of possibilities. Using our method, the user can express his aesthetic intent by defining target parameters. Thus, our objective function is completely configurable: it is made of several terms, each one taking into account a target value. The importance of each term is specified by using different weighting factors.

1.3 Outline

This report is organized as follows. In section (2), we will present existing methods to modify the aesthetics of a image. These methods can be classified in two categories: post-processing methods (2.1) and inverse lighting methods (2.2). We will then discuss these methods and present the objectives of our method. Section (3) presents the overall framework for our approach (3.1), further details about our contribution (3.2) and validation results (3.3). In section (4), we give further details about the implementation of the method. Section (6) presents and discusses several results obtained using our new inverse lighting method. Finally, we conclude our report in section (7) and discuss different avenues to further enhance our method.

2 Related Works

Several approaches have been developed for modifying an image aesthetic. Previous works can be classified in two categories: signal-processing methods and inverse-lighting methods. This section details for each category notable works and their applications.

2.1 Post-processing methods

The methods discussed in this section consider aesthetic design as a signal-processing step applied to an image or a model of the scene. In [13], the authors describe a method to extract a set of pixel color and luminance transformations from example pairs of images. These transformations represent an implicit style. The style, learned from examples, is transferred to an image by applying these transformations.

Other methods use 3D information during the post-processing step to provide a better understanding of the shape of the object. In [10], the authors use a 3D mesh of the scene. Considering the vertex luminance signal is S, a low-pass signal S_σ is applied: for each vertex, its luminance value is computed as the mean of the luminance of its direct neighbors in the mesh. Given S and the low-pass filtered signal S_σ computed previously, a contrast signal $C(S)$ is computed as follows:

$$C(S) = S - S_\sigma \tag{1}$$

In order to enhance the final visualization of the model, the enhanced signal is computed as follows:

$$U(S) = S + \lambda C(S) \tag{2}$$

Using this method, the contrast of the final image is enhanced around the edges of the principal object, bringing a better understanding of the shape of the objects.

The methods we discussed in this section enhance the appearance of an image by transferring a style or by improving its contrast. However, in the context of global illumination, we want to control the lighting of the scene more precisely, by taking into account its 3D structure. The following section introduces a category of methods that allow a more precise lighting design.

2.2 Inverse Lighting

Inverse-lighting methods allow us to compute an ideal lighting configuration for the 3D scene, given a lighting specification. This lighting specification acts as a target that represents the desired result. It can be defined through different interfaces, as explained in [7].

Inverse-lighting methods are usually built with the following structure :

- The target is specified by the user,

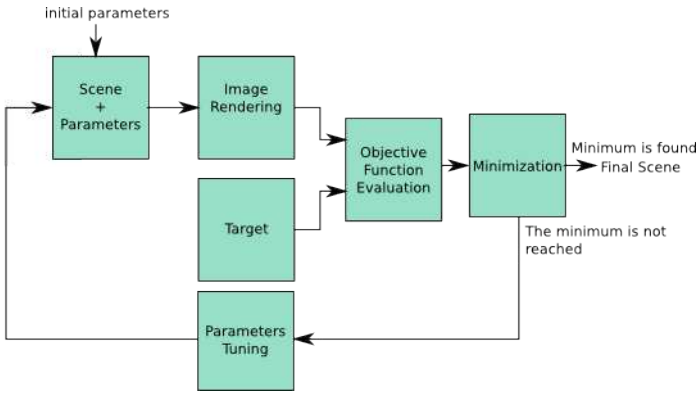


Figure 4: Principle of inverse lighting

- The 3D scene is rendered using an initial set of parameters,
- An objective function measures the distance between the rendering result and the target,
- If the distance is too important, the scene's parameters are changed and the process is repeated.

Inverse-lighting methods differ by the type of target they use, the optimization process, but also the parameters they modify. In the rest of this section, we will distinguish two sub-categories of inverse-lighting methods: image-based and global methods.

2.2.1 Image-based methods

The method exposed in [11] requires a pre-rendered image as a target goal. The distance between the target image and the rendered image is the objective function to minimize to get the desired parameters of the lighting. This target image is provided by the user through a painting interface.

In [11], inverse-lighting is performed in the context of radiosity. Radiosity-based methods simplify the lighting equation [5] by dividing the scene's surfaces into small surface elements, or patches. The radiosity (luminous flux emitted per surface element) is considered constant over a patch. Solving the rendering equation is thus equivalent to solving a linear system of equations :

$$B_i = E_i + \rho_i \sum_{j \in \{patches\}} F_{ji} B_j \tag{3}$$

where B_i is the radiosity of the patch i, E_i is the emission of the patch i (0 if the patch is not on a light source), ρ_i its reflectivity, and F_{ji} is the form factor between the patch

j and the patch i, it is the proportion of energy leaving patch j that arrives at the path i.

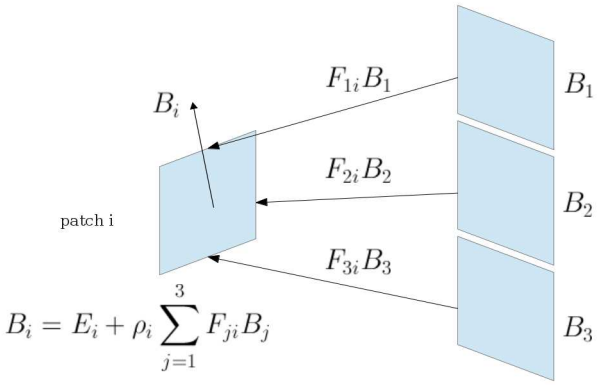


Figure 5: Principle of radiosity

The solution of the system of equations (3) is a radiosity vector which elements are the radiosities B_i of the patches. Rendering an image is equivalent to finding for each pixel the contribution of the visible patche i. Note that B_i comprises the effect of direct and indirect illuminations. This contribution is the radiance of the pixel $L_i = \frac{B_i}{\pi}$, B_i being the radiosity of the patch i visible through the pixel.

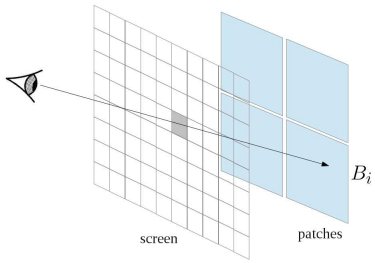


Figure 6: rendering an image

By painting on an image of the scene with a brush, the user provides a target radiosity vector Ψ which elements are the desired radiosity for each patch of the scene. The goal of this method is to find the intensity of each light source in the scene. The radiosity equation is solved for every light source independently. Thus, for each light source L_i, we solve the system of equations (3) to get a radiosity vector Φ_i. When considering every

light source at the same time, the radiosity vector can be defined as a linear combination of the vectors Φ_i :

$$\hat{\Psi} = \sum_{i=1}^{n} w_i \Phi_i \tag{4}$$

In order to find the contribution factor w_i of each light source, the objective function $||\Psi - \hat{\Psi}||$ has to be minimized using a least-square method. Light source positions and orientations are fixed, and only their intensity is automatically computed. Moreover, the user has to provide the target image using the painting interface, which can be a time-consuming task.

The SAIL model presented in [15] does not use an image of the scene as a target. Instead, the provided target is an image of an illuminated sphere. The target image is thus independent of the scene. However, generating and understanding such an image can be a difficult task for the user, as the simple geometry of the sphere may only allow a coarse specification of the lighting goals. The method discussed in [15] uses a configuration made of two light sources (key light and fill light). It computes a parameter set (key-to-fill ratio, orientation and position of the key light) that minimizes the distance between two vectors of appearance metrics : one vector computed on the illuminated sphere and another computed on the image of the 3D scene. These vectors are defined as a 3 element vector (θ, β, C) where (θ, β) are the apparent light-direction in spherical coordinates, and C is a measure of contrast.

The goal of [15] is to provide automatic lighting design in real time. A mathematical model describing how surfaces reflect light is computed as a preprocessing: for a set of points in the parameters space, the value of the appearance metrics vector (θ, β, C) is pre-computed. At runtime, the function is minimized by applying gradient descent toward the point that is the closest to the target sphere image.

The main difficulty in a model such as SAIL is to provide an image of an illuminated sphere that communicates the desired lighting configuration. Instead of using a target image, the methods presented in the next sub-section use a set of target values that better characterize the desired configuration.

2.2.2 Global Methods

The methods of this category do not use an image as a target. Instead, they use a set of target values for descriptors that describe an image's aesthetics. These descriptors have to be chosen carefully. The final objective of such methods is to minimize a objective function, often a linear combination of several terms.

In [6] the authors use a linear combination of three terms that measure the mean brightness, non-uniformity and peripheral characteristics of the lighting of the scene. The weight assigned to each of these terms is specified by the user so that he can construct his own constraint. The optimization variables can be light source direction, surface element radiosity B_i, reflectivity ρ_i or emissivity E_i. The objective function is minimized iteratively using the BFGS minimization algorithm [9].

The method introduced in [12] computes the lighting configuration that is optimized for understanding the shape and fine details on the scene's objects. It uses an objective function that is a linear combination of terms defined as follows:

$$f_q = w_1 f_{mean} + w_2 f_{var} + w_3 f_{hist} + w_4 f_{grad} + w_5 f_{edge} + w_6 f_{dir} \tag{5}$$

- f_{mean} measures the distance from the mean luminance to a target value
- f_{var} measures the distance from the luminance variance to a target value
- f_{hist} measures the distance of the luminance histogram from an equalized histogram
- f_{grad} measures the magnitude of gradients
- f_{edge} evaluates the appearance of edges
- f_{dir} measures the elevation of light sources

The optimization variables considered here are the direction of the light sources (θ_i, ϕ_i) and the intensity of its diffuse and specular components, as the rendering process uses standard OpenGL shading. Optimization is performed iteratively using a gradient descent method.

In the following section, we will discuss the above presented methods, and present the different properties our method should have.

2.3 Discussion

The aesthetic qualities of an image rendered using global illumination depend on many parameters: the number of light sources, their geometry, the reflectivity of surfaces, etc. It is wise to consider that a post-processing process, as described previously, may not provide all the clues on these parameters for efficient lighting design. Therefore, we will concentrate on inverse-lighting methods.

Inverse-lighting methods such as the one described in [11] can be used to determine values for several parameters (in that case, the luminance of light sources). However, this method works under the assumption that the user can provide a target image that describes his lighting goal. This is a time-consuming process and although the result may feature the right highlighted areas, it may not communicate the aesthetic expected by the user.

The approach taken in [15] does not require the task of painting light on surfaces. The target used is an image of an illuminated sphere. However, generating such an image that communicate the desired aesthetic seems to be a difficult task.

The work presented in [12] relies on an objective function to improve the appearance of an image in order to communicate its shape as well as possible. The objective function is made up of target terms that are computed from the scene. It is then minimized to provide an optimal set of parameters (orientation of light sources, specular and diffuse intensities). However, we think that there are several limitations to this approach. This method has been designed to improve the understanding of 3D shapes. This gives to images a certain aesthetic. However, the goal of our work is to cover a broader range of aesthetics. The user should be able to specify target values that are accounted for in the objective function to control the lighting design process. As we are working in the context of global illumination, the lighting design process will have to account for multiple reflections. Our approach should be generic enough to not rely on a specific global illumination rendering algorithm. When rendering with global illumination, the interactions between multiple light sources is important. Instead of optimizing for a single light source at a time, as in [12], we want to optimize the parameters for all the light sources in the scene at the same time. The solution should also provide flexibility in terms of the nature of light sources, whereas only point lights are considered in [12]. In particular, we are interested in studying the influence of a light source's dimensions.

Similarly to [12], we think it is important to specify a main object for the scene to evaluate the aesthetic properly. This is analogous to photography where an isolated subject is considered. The rest of the image that does not represent this main object is considered as background.

The objective function is only one step of the optimization strategy. As it is a weighted sum of several terms, we want to study the slope of the hyper-surface that represents the value taken by each term of the objective function over the light sources' parameters space. We want to provide a better understanding of the contribution of each term: one term for instance may prevent the optimization process from reaching a global minimum. In the next section, we will introduce our approach, based on [12] for defining the objective function and for designing the optimization process.

3 Principle of the approach

Our approach is based on [12]. However, because our goal is different, several modifications have been made to accommodate these differences as will be explained in this section.

3.1 Framework

Our method is based on the following framework (7):

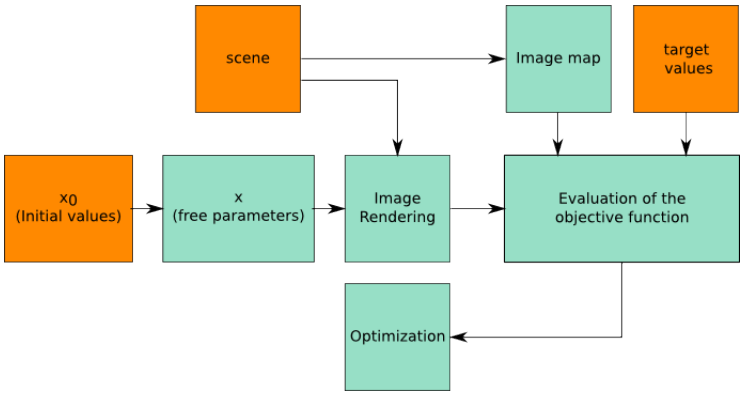


Figure 7: Our framework. Darker, orange boxes represent user input.

This section will detail each block of this diagram. Section (3.2) will give further details on the free variables, the evaluation of the objective function and the optimization process.

Our goal is to modify the aesthetics of a scene by changing the appearance of the main object in the scene. The 3D scene has two light sources identified as the key light and fill light and the main object of the scene is specified. In order to evaluate the aesthetics of the scene using the objective function, we have to differentiate the different zones of the image. In a pre-processing step, a mask image of the scene is computed. It indicates whether a pixel corresponds to the main object or to the background. It also identifies the pixels that correspond to two classes of edges: outer edges (between the object and the background) and inner edges (that correspond to edges inside the object).

Before the optimization process begins, free variables have to be chosen and properly initialized. These free variables are the parameters of the scene that will be tuned to change the image's appearance. As with any gradient-based function minimization process, different initial values can lead to different results, because of local minima. The vector that contains the initial values is noted x_0. The parameters can be the position, the orientation, the dimension, and the intensity of the light sources.

The objective function we are using uses a set of target values to describe the desired aesthetic. These target values are constant, and have to be set before running the optimization process. We will detail them in a sections (3.2.4) to (3.2.8). They are stored in a vector t.

Given the initial parameters x_0, a first luminance image I is computed using global illumination. The choice of the rendering method does not have an influence on the overall framework, as computing the objective function only requires the luminance image I. Methods such as radiosity, path tracing or photon mapping can be used, without compromising the overall framework.

Using the previously computed image and the image map, the objective function f_q is evaluated. The objective function f_q is computed as a linear combination of terms:

$$\begin{aligned} f_q = w_1 f_{meanBack} + w_2 f_{meanObj} + w_3 f_{varBack} + w_4 f_{varObj} + w_5 f_{hist} \\ + w_6 f_{grad} + w_7 f_{edgeIn} + w_8 f_{edgeOut} \end{aligned} \tag{6}$$

Each term in f_q takes into account a target value. The main difference between our objective function and the one used in [12] is that all the terms we use can be configured by the user. The terms of the objective function are normalized and defined as follows:

- $f_{meanBack}$ and $f_{meanObj}$ are computed as distances to a user-provided target mean luminance values for the background and the object respectively.
- $f_{varBack}$ and f_{varObj} are computed as distances to a user-provided target luminance variance values for the background and the object respectively.
- f_{grad} is computed as the distance between a target gradient value and the gradient value over the principal object of the scene (defined by the user).
- f_{hist} is the Kullback-Leibler divergence between the normalized luminance histogram of the image, and a user-provided distribution.
- f_{edgeIn} and $f_{edgeOut}$ measure the distance between a target value and response to an edge detector for two classes of edges.

We use as distance, for instance for the term $f_{meanObj}$, the difference $|\bar{l}_{Obj} - t_{meanObj}|$ where $\bar{l}_{Obj}$ is the mean luminance computed over the object's pixels in the rendered image I and $t_{meanObj}$ is the corresponding target value specified by the user.

In our approach, in contrast to [12], the target values are not computed from the scene, but are directly provided by the user. By doing so, we allow the field of attainable aesthetics to be larger. For instance in [12], the lighting aesthetics that makes shapes more difficult to understand would be penalized, whereas in our approach, if the user gives such a specification, this type of aesthetic would be possible.

The optimization step is used to find a set of parameters that minimize the objective function. In [12], the optimization is performed for each light source alternatively. However, it is necessary in the context of global illumination to consider all the light sources at the same time, as both fill light and key light contribute differently to the aesthetic

of the image and both have a specific role. At each iteration, the scene is rendered, then new values of the parameters to optimize are computed.

3.2 Contribution

This section details all the differences between our approach and the one introduced in [12]. As our goal is different, we use a different objective function and the optimization method needs to fulfill certain properties.

The goal of our method is to automatically design a lighting configuration that brings a target aesthetic to the scene in the context of global illumination. In [12], the goal was to design a lighting configuration that helps to understand the shape of the main object in the scene. In our context, we want to obtain an image with a certain aesthetic, not necessarily the most understandable one, and we want the user to have control on the aesthetic of the scene using target values for image descriptors. Furthermore, in the context of global illumination, the range of attainable aesthetic is wider. Within that goal, as we want our approach to be as generic as possible, any rendering method can be used, and any kind of light source can be considered (point light source, area light source, directional light source).

3.2.1 Choosing the free variables

The output of our algorithm is a set of lighting parameter values that minimizes the objective function. These parameters can vary with the kind of light source chosen: a point light can be defined using a 3D position and a luminance value. An area light source has a size that also influences the lighting of the scene. Spotlights have even more parameters, such as the orientation or the falloff angle. Here, we are using two spherical light sources and we set as free variables the radius of each source, and the key-to-fill ratio. That is the ratio of the luminance of the key light to the luminance of the fill light.

3.2.2 Computing the mask image

Some terms in the objective function have to be computed in specific zones of the image: edge, background or object pixels. To identify these subset of pixels, a mask of the scene is pre-computed. The mask is generated using the 3D rendering engine: a ray is cast from each pixel of the sensor. If the ray intersects the main object, a luminance value of 255 is assigned to the pixel. Otherwise, the pixel has a luminance value of 0. Edge pixels are also detected and are split in two categories:

- Outer edges are caused by occlusion between the main object and the background,
- Inner edges are caused by a discontinuity of surface orientation on the main object.

Outer edges are extracted by simply detecting the neighboring background and object pixels. Inner edges are extracted by evaluating the change in surface orientation between neighboring pixels. An example of the image mask is given below:

Figure 8: Mask identifying the different zones of the scene

In the figure (8), the pixels in white represent the main object, whereas the pixels in black represent the background. Light grey pixels represent inner edges and a dark grey represents outer edges.

3.2.3 Objective function

The objective function (eq.6) is a linear combination of several terms. Each term is defined as a distance between a value (mean luminance, variance, gradient amplitude) computed on a rendered luminance image I and a target value provided by the user, and is normalized so that its value ranges within [0,1]. Without normalization, some terms would have such small values that their influence would be too small compared to the terms with a bigger range of values. Weighting terms w_i bring another level of control to the user: by using different values for the w_i, he can build up his own objective function. By using a weighting value of 0, the user can completely inhibit a term.

3.2.4 $\mathbf{f_{meanObj}}$ and $\mathbf{f_{meanBack}}$

These two terms control the mean pixel luminance value on the object and on the background respectively. We use two different terms for the object and for the background as the mean luminance values for the foreground and the background are characteristics of high key and low key images.These two terms control whether the object and the background appear dark or bright.

The target values $t_{meanObj}$ and $t_{meanBack}$ are mean luminance values, defined between 0 and 255. The terms $f_{meanObj}$ and $f_{meanBack}$ are computed as follows:

$$f_{meanObj} = \frac{|\bar{l}(object) - t_{meanObj}|}{max(t_{meanObj}, 255 - t_{meanObj})} \tag{7}$$

$$f_{meanBack} = \frac{|\bar{l}(background) - t_{meanBack}|}{max(t_{meanBack}, 255 - t_{meanBack})} \tag{8}$$

where $\bar{l}(object)$ and $\bar{l}(background)$ are the mean luminance over object pixels and background pixels respectively, computed using a global illumination algorithm.

3.2.5 $\mathbf{f_{varObj}}$ and $\mathbf{f_{varBack}}$

These two terms measure the distance between a target value and the standard deviation of the luminances of the main object's pixels and the background's pixels respectively. The pixel luminance can exhibit strong variations for a subset of the image. On the contrary, the variation of pixel luminance can be very small. Here again, we distinguish between background and object pixels, as it gives more control over each image zone's appearance. For example, the background can be characterized by a small variation of luminance while the main object has a strong variation. The target terms for standard deviation are t_{varObj} and $t_{varBack}$. The expressions for the terms f_{varObj} and $f_{varBack}$ are given below:

$$f_{varObj} = min(\frac{|\sigma(object) - t_{varObj}|}{t_{varObj}}, 1) \tag{9}$$

$$f_{varBack} = min(\frac{|\sigma(background) - t_{varBack}|}{t_{varBack}}, 1) \tag{10}$$

where $\sigma(object)$ and $\sigma(background)$ are the luminance standard deviation over the main object's pixels and the background's pixels respectively.

3.2.6 $\mathbf{f_{grad}}$

The gradient term f_{grad} is used to control the shading gradient on the object's surface. Shading gradient are important because they bring information about the perception of an object's shape. A strong shading gradient will result in a strong emphasis of the geometry of the object, bringing a aesthetic different than low amplitude shading gradient. The shading gradient term is computed for the pixels of the main object only.

A norm of gradient image is computed using the current image I provided by the global illumination algorithm. First, two images are computed using a pair of Sobel filters of kernels A and A^T with:

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \tag{11}$$

Using these filters, we can compute two gradient images $G_x = A \star I$ and $G_y = A^T \star I$. We obtain for each pixel $p_{i,j}$ the gradient vector $\nabla l_{i,j} = (G_x(i,j), G_y(i,j))$. The average shading gradient norm is then computed as follows:

$$g(object) = \sqrt{\frac{1}{N} \sum_{p_{i,j} \in object} |\nabla l_{i,j}|^2} \tag{12}$$

where *object* is the set of pixels marked as object's pixels in the image mask, and N the number of these pixels. The final term f_{grad} is defined as:

$$f_{grad} = \frac{|g(object) - t_{grad}|}{max(t_{grad}, 255 - t_{grad})} \tag{13}$$

3.2.7 $\mathbf{f_{hist}}$

As explained in [8], luminance histograms must be used with other descriptors to express an image's aesthetic. However, by using supervised learning on well known aesthetics (such as high key, low key, and medium key images), a signature histogram can be extracted for each class. These signature histograms are then used as target that will be compared with the luminance image's histogram.

Minimizing the objective function brings the image's histogram closer to the target histogram. This has an impact on the image's mean luminance value and on the standard deviation. The f_{mean} and f_{var} terms have an influence on the value of the term f_{hist}. However, f_{hist} describes the complete luminance distribution and does not provide any spatial information, whereas f_{mean} and f_{var} evaluate statistics for distinct areas of the image, so these terms are more complementary than they are redundant.

This term is evaluated as the Kullback-Leibler divergence between two distributions. The KL divergence is defined as follows:

Definition 1 *For two discrete random variables p and q such as p is absolutely continuous with respect to q, the Kullback-Leibler Divergence of p from q is:*

$$D(p||q) = \sum_i p_i log(\frac{p_i}{q_i}) \tag{14}$$

Using the image's normalized histogram h has the distribution of p and the target histogram t_{hist} as the distribution of q, we can compute the value for the KL-Divergence. For proper normalization, the final term is:

$$f_{hist} = 1 - e^{-\frac{D(h||t_{hist})}{\alpha}} \tag{15}$$

where α is used for normalization. $f_{hist} = 0$ implies that the target image and the rendered image have the same luminance histograms.

3.2.8 $\mathbf{f_{edgeIn}}$ and $\mathbf{f_{edgeOut}}$

These two terms measure the distance between target values and the response to an edge detector for two categories of edges. An edge detector is applied to pixels marked as edge pixels in the image map. The gradient image computed for f_{grad} and a Laplacian image are needed to compute these two terms.

The expression of this term is the following:

$$f_{edge} = \frac{1}{t_{edge}}(t_{edge} - \frac{1}{N_{edge}} \sum_{(i,j)\in edge} O_{edge}(p_{i,j})) \tag{16}$$

where t_{edge} is the desired edge detection response, N_{edge} is the number of pixel in the considered class of edge (inner or outer) and O_{edge} is the edge detection operator which response is in the range [0,1].

3.2.9 Optimization process

In the previous section, we specified an expression for each term of the objective function. The optimization process will find the minimum of the objective function by evaluating it at different position in the space of parameters.

The optimization algorithm has to fulfill several properties. The free parameters have different order of magnitude, and can be bounded: when considering a spherical light source, its radius should be strictly positive. As a result, the optimization technique should take into account bound constraints.

We use a gradient descent technique to minimize the objective function. However, our objective function does not have an analytic expression: we can write a partial expression f_q (equ.6) but it uses the luminance image I acquired using the rendering engine. Thus, we can not differentiate it directly. Partial derivatives are approximated numerically by evaluating the objective function at differential steps in the direction of each free variable. Since the parameters do not have the same order of magnitude, these differential steps should have different values.

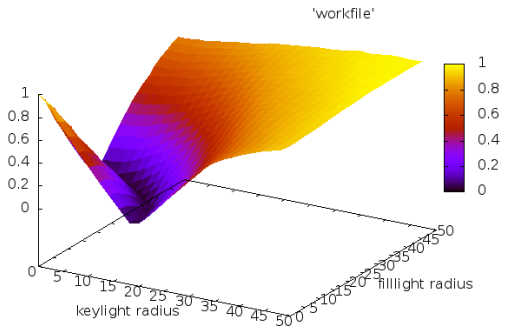
The optimization step uses the L-BFGS-B algorithm [14]. It is a memory limited implementation of the BFGS algorithm used in [6] that takes into account additional constraints such as bound parameters.

3.3 Partial results and tests

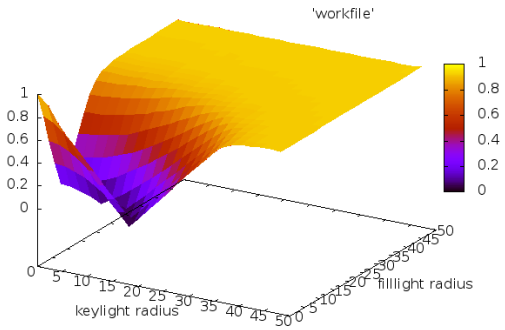
In section (3.1) and (3.2), we have presented the overall framework and detailed our objective function. We use function minimization to find optimal light parameters. It is important to verify the relevance of the different terms in f_q. For a term to be relevant for the optimization, it has to satisfy two properties:

- It should not add two much local minima in the objective function,
- Its impact on the final image should be predictable.

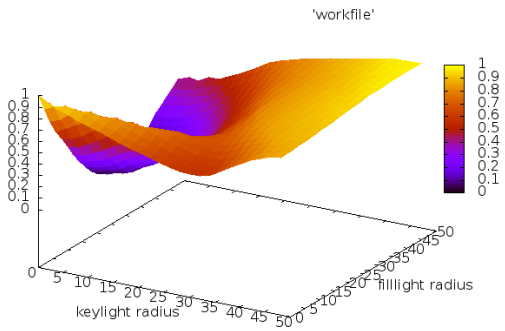
In this section, we will present several minimal test results, along with the evolution of each objective function term depending on the lighting parameters. To represent this evolution, we only use two free parameters, and evaluate the objective function f_q for different values of these parameters. Using this process, we can see whether or not the term might introduce too many local minima that could compromise the minimization.



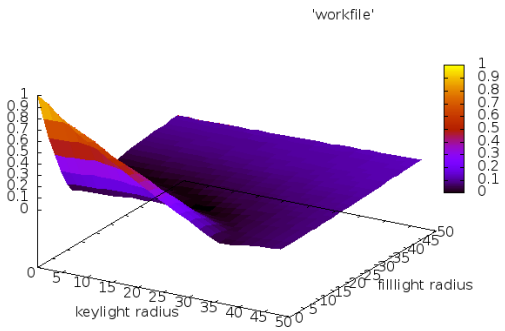
(a) $f_{meanObj}$



(b) $f_{meanBack}$

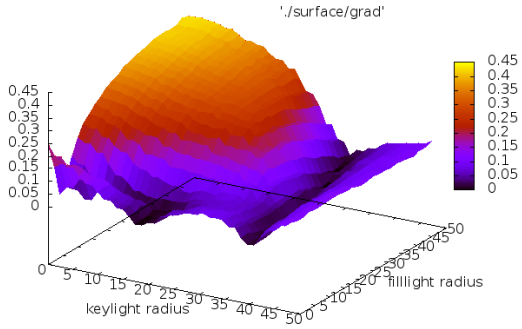


(c) f_{varObj}

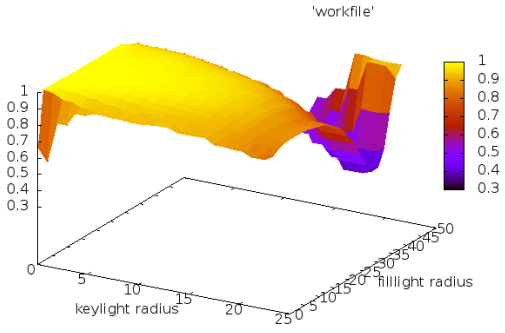


(d) $f_{varBack}$

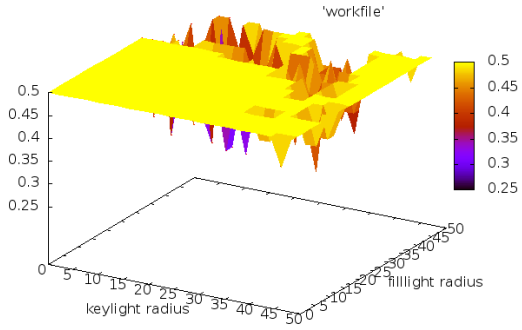
Figure 9: Evaluation of each term of the objective function depending on the key-light and fill-light radii



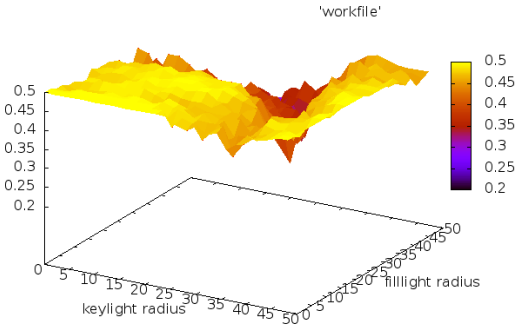
(e) f_{grad}



(f) f_{hist}



(g) f_{edgeIn}



(h) $f_{edgeOut}$

Figure 9: Continued

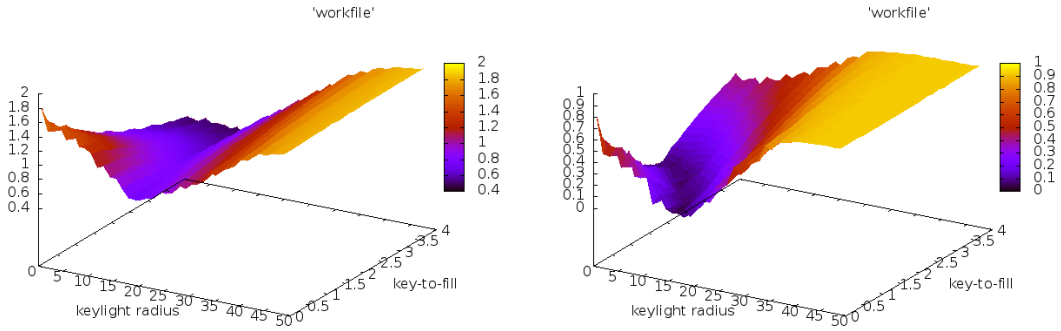
Figure (9) shows the values taken by each term of f_q when varying the key-light and the fill-light radii. The key-to-fill ratio is constant and has a value of 0.5. Target values are defined as follows:

target term	value
$t_{meanObj}$ and $t_{meanBack}$	120
t_{varObj} and $t_{varBack}$	45
t_{grad}	50
t_{hist}	high key signature histogram
t_{edgeIn} and $t_{edgeOut}$	0.5

As we can see, the values of the $f_{meanObj}$, $f_{meanBack}$, f_{varObj}, $f_{varBack}$, f_{grad} and f_{hist} terms are smooth and the global minimum is easily attainable. However, the f_{edgeIn} and

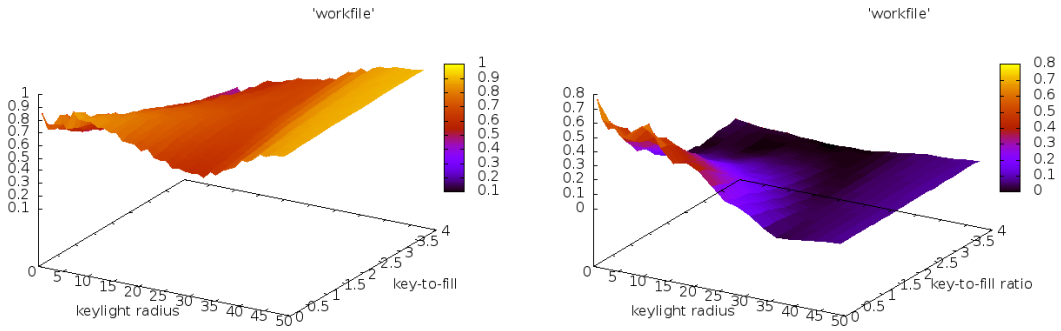
$f_{edgeOut}$ are not smooth and have numerous local minima. Since we are using a gradient descent method to minimize the objective function, using these terms would compromise the optimization process.

The same test has been performed when considering only the key-light radius and the key-to-fill ratio as free variables. The fill-light radius is set to a constant value. Figure (10) shows the objective function's values, computed for key-light radiuses ranging from 0.1 to 50 and for a key-to-fill ratio ranging from 0 to 4.



(a) $f_{meanObj}$

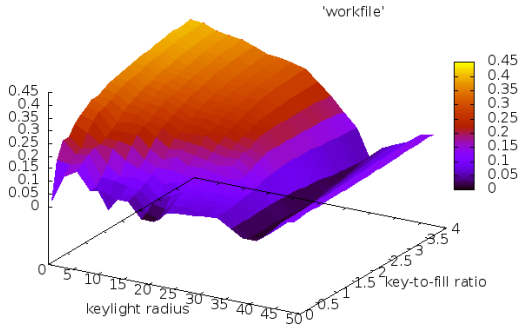
(b) $f_{meanBack}$



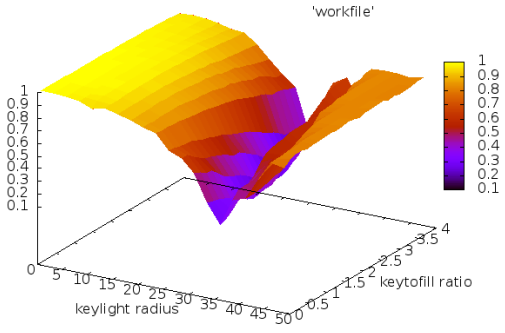
(c) f_{varObj}

(d) $f_{varBack}$

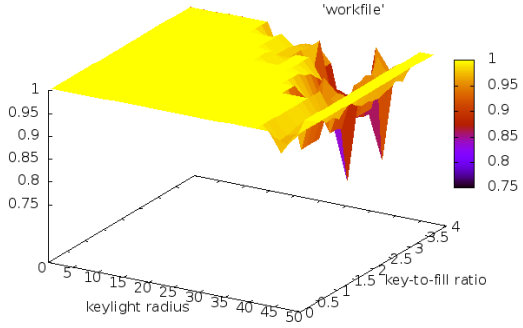
Figure 10: Evaluation of each term of the objective function depending on the key-light radius and the key-to-fill ratio



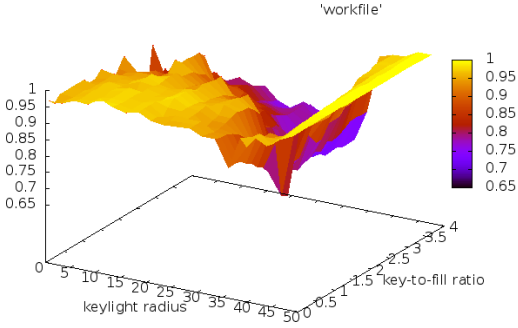
(e) f_{grad}



(f) f_{hist}



(g) f_{edgeIn}



(h) $f_{edgeOut}$

Figure 10: Continued

In this case as well, we can see in figure 10 that the terms $f_{edgeOut}$ and f_{edgeIn} might compromise the minimization, as they present numerous local minima and the surface does not seem to be smooth. As a result, we have decided not to use these terms in the objective function. The shading gradient and variance terms also describe the variation of luminance over the surface's object, and the surfaces all present a smooth descent toward global minima.

We can evaluate our optimization method using a similar method: we compute 6 images by only considering the remaining terms $f_{meanObj}$, $f_{meanBack}$, f_{varObj}, $f_{varBack}$, f_{grad} and f_{hist} one by one.

(i) $f_{meanObj}$ (j) $f_{meanBack}$

(k) f_{varObj} (l) $f_{varBack}$

(m) f_{grad} (n) f_{hist}

Figure 11: Optimization result by considering only one term for f_q

The images in figure (11) are rendered by using the optimal lighting configuration obtained with our optimization method. The image (11j) is obtained by considering $f_q = f_{meanBack}$ while image (11l) is obtained by considering $f_q = f_{varBack}$ and so forth. Figure (3.3) compares the target value used and the values computed on the luminance image obtained with our method:

function term	target value	final value
$f_{meanObj}$	128	128
$f_{meanBack}$	128	131
f_{varObj}	45	44.87
$f_{varBack}$	45	27.57
f_{grad}	50	43

To obtain these images, the optimization process searches in a 3-dimensional space: the parameters are the key-light and fill-light radiuses, along with the key-to-fill ratio. The values computed on the final luminance image are close to the specified target values.

4 Implementation of the model

This section gives more details about the implementation of our method. The framework introduced in section (3.2) makes it possible to separate the optimization process from the rendering platform. As a result, our method is independent from the rendering process and another platform could be used. In our implementation, we are using the `Mitsuba` software for rendering.

4.1 Main program

The structure of our program is as follows:

Algorithm 1 Main program

1: **User input :** x_0, w, t, scene
2: computeImageMask() ▷ call mitsuba
3: setupObjectiveFunction()
4: x = optimizationFunction()
5: **return** optimal parameters x

The main program proceeds as follows. The user has to provide a scene, initial parameters values x_0, target values t, and weighting parameters w for the objective function. Computing the image mask is done using the rendering engine that provides all the information we need. Indeed, we cast a ray toward the scene through each pixel, and determine which object is intersected. Then, the objective function is initialized with

Algorithm 2 Optimization function

while minimum not reached **do**
  $x_n = gradientFunction(x_{n-1}) + x_{n-1}$
  $x_{n-1} = x_n$
end while
return optimal parameters x_n

the values of t and w. The optimization process then starts from the initial position x_0, and compute new parameters values iteratively, as shown in algorithm (2). After the minimization process is complete, the final values of the free parameters are returned in the vector x.

The following pseudo-code illustrates the computation of the objective function:

Algorithm 3 Objective function

```
1: function FQ(x, t, w, mask, scene)
2: modifyParameters(scene,t)
3: I = renderScene()                    ▷ lumiance image rendered using mitsuba
4: h = histogram(I)
5: grad = gradient(I)
6: lapl = laplacian(I)
7:
8: # Term initialization with target values, weights and input images
9: fhist.init(w_hist,t_hist,h)
10: fmean.init(w_meanObj, w_meanBack, t_meanObj, t_meanBack, I, mask)
11: fvar.init(w_varObj, w_varBack, t_varObj, t_varBack, I, mask)
12: fgrad.init(w_grad, t_grad, gradient, mask)
13: fedge.init(w_edgeIn, w_edgeOut, t_edgeIn, t_edgeOut, gradient, laplacian, mask)
14:
15: # Computation of the objective function's value
16: return fhist.eval() + fmean.eval() + fvar.eval() + fgrad.eval() + fedge.eval()
17: end function
```

The objective function is passed as a parameter to the optimization function. It is called iteratively, until the minimum is found. As shown in algorithm (3), the luminance image I is obtained by calling the rendering platform. Using this luminance image, target values t, weighting factors w, histogram, laplacian and gradient image, every term of the objective function is evaluated.

The main program is implemented using the python programming language. Python offers numerous advantages:

- Python is an object-oriented language,

- Numerous python modules are available,
- Various C/C++ libraries offer python bindings,
- Python syntax is not verbose, and easy it is to debug.

Python being an object-oriented language, it is easy to replace one part of the framework without compromising the rest. For instance, each term in the objective function is modeled its own class, in the object-oriented programming sense. In this way, we could easily replace the f_{hist} without compromising the rest, as long as the class has an initialization and an evaluation method.

Numerous python modules are readily available. In our implementation, we use `scipy` [3] for function minimization, and `numpy` [1] for matrix operations. For image processing, we use the `OpenCV` library [2], which offers python bindings while keeping native code performances.

4.2 Rendering platform

We use `Mitsuba`[4] for rendering and for computing the image mask. It is an open source rendering software that is extremely modular. By creating a new plugin, it is possible to define new surface materials, new types of light sources, or to implement new rendering algorithms.

A scene in `Mitsuba` is described in a `.xml` file. Every object in the scene is described, including sensors, light sources, objects but also the rendering technique, called integrator.

```
<shape id="keylight" type="sphere">

    <transform name="toWorld">
        <translate x="250.0" y="100.0" z="-500.0" />
    </transform>

    <emitter type="area">
        <spectrum name="radiance" value="500.0" />
    </emitter>

    <float name="radius" value="16.1" />
</shape>
```

Figure 12: Declaration of the key-light source

Figure (12) shows how an area light-source is declared in `Mitsuba`. Notice that our spherical emitter is declared with an identifier `id="keylight"`. This is how we separate the key light from the fill light. During the optimization process, the parameters of the light sources are modified. We could use the python bindings for `Mitsuba` but unfortunately it is not possible to change the area light sources' parameters directly in python. This is why we directly parse and modify the `.xml` file to change the scene's parameters.

`Mitsuba` is also used to compute the mask image. Since it allows integrator nesting, we use two integrators for computing the mask image. The first integrator is used to create a binary image that tells whether a pixel is an object or a background pixel. The principle is the following: for each pixel, a ray is cast toward the scene. If the ray intersects the main object, then the pixel is white. In any other case, the pixel is black.

The second integrator is used to localize edge pixels. As detailed in section (3.2.2), edges can be categorized in two sets: inner and outer edges. To locate the outer edges, we first need to know where the object and background pixels are. That is why we use nested integrators. The second integrator computes edges, uses the result of the first integrator and computes the edge pixels in a post-processing step.

5 Results

The method we introduced in this report has been applied to different scenes with different target values to obtain an optimal lighting configuration that corresponds to a desired aesthetics. This section highlights some of the results we obtained using this method.

For our scenes, we use spherical light sources. For a spherical light source, the luminous flux is:

$$\Phi = 4\pi^2 R^2 L \tag{17}$$

where R is the radius of the sphere and L is the emitted luminance of the source.

5.1 First scene

The first scene is a Cornell box with the Utah teapot at its center. The key-light is located behind the camera and the fill-light is located between the teapot and the back wall, as shown in figure (13).

Our lighting goal in this first example is to obtain a high-key image using this scene. High-key images are characterized by an abscence of dark shadows, and are very bright. However, in typical high-key images, there should be no loss of information on the main subject. Using all these characteristics, we can already set up the target values.

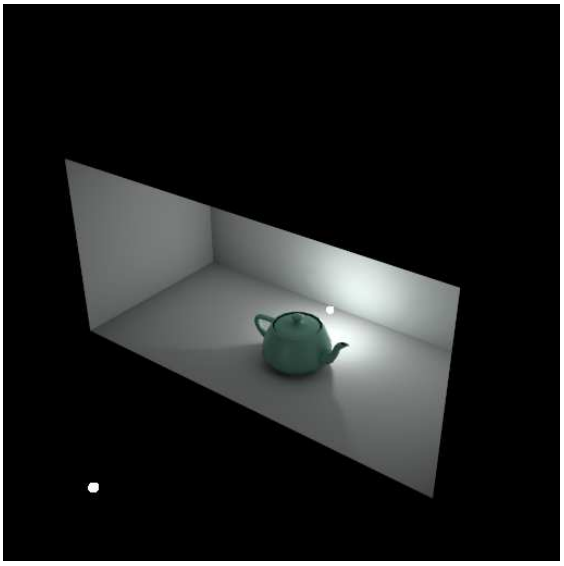


Figure 13: Light source positions in the first scene

function term	target value
$f_{meanObj}$	120
$f_{meanBack}$	200
f_{varObj}	60
$f_{varBack}$	45
f_{grad}	100
f_{hist}	high-key signature

Figure 14: Target values for the first scene

The target values presented in figure (14) have been chosen as follows: the mean luminance on the background's pixel should be high, because we want the background to be bright. The variance should also be lower on the background than on the object as we want a uniform background while still keeping detail on the object. For the histogram term, we use a high-key signature. This high-key signature is a luminance histogram obtained using supervised machine learning using a learning set of high-key images. The f_{hist} term ensures that the final image's luminance histogram is close to this target distribution.

The weights for each terms of the objective function are chosen as shown below:

The histogram term has the most important contribution, while the gradient term is considered the least important term. The free parameters are the key-light and fill-light radii, as well as the ratio between the key-light and the fill-light luminances (also called

function term	weighting factor
$f_{meanObj}$	0.4
$f_{meanBack}$	0.4
f_{varObj}	0.5
$f_{varBack}$	0.5
f_{grad}	0.2
f_{hist}	1.0

Figure 15: Weighting factors for the first scene

key-to-fill ratio).

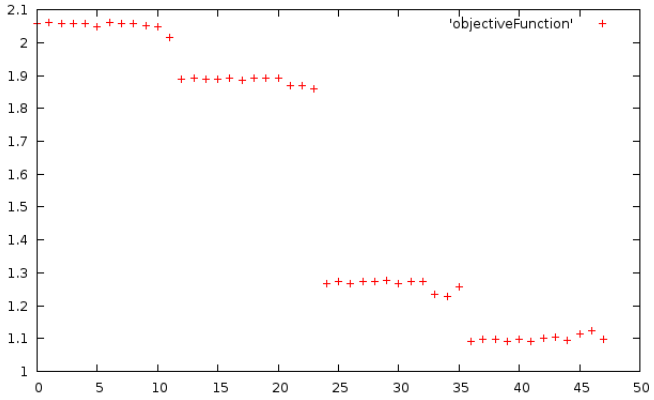


Figure 16: Values of the objective function f_q for each iteration for the first scene

Figure (16) shows the evolution of the objective function f_q during the optimization process. The abscissa of the graph corresponds to the number of evaluation of the objective function. Each step corresponds to an iteration. During each iteration, the objective function is evaluated to approximate the gradient in each optimization direction, (say for each parameters).

(a) Initial image of the first scene (b) Final image of the first scene

Figure 17: Results for the first scene

Figure (17a) shows the scene with the initial parameter values, before the optimization step. Figure (17b) shows the scene rendered with the optimal light parameters. As we can see, the background is very bright, and there is no information lost on the object. The teapot does not cast dark shadows on the scene. The parameter's initial and final values are presented in figure (18). The key-to-fill ratio and the fill-light's radius have increased significantly. We ascertain from eq.(17) that the luminous flux of the fill-light has increased. This larger and brighter fill-light is used to *fill* the dark shadows in the background.

parameter	initial value	final value
key-light radius	3.0	3.30
fill-light radius	3.0	7.66
key-to-fill ratio	0.1	6.91

Figure 18: Initial and resulting parameters for the first scene

Figure (19) shows the luminance, variance and gradient amplitude values computed on the final image. The mean luminance value on the background is higher than the mean luminance value on the object. The gradient value is also important as we wanted to keep details on the object.

function term	final value
$f_{meanObj}$	175
$f_{meanBack}$	245
f_{varObj}	43.01
$f_{varBack}$	43.95
f_{grad}	181.39

Figure 19: Values computed on the final image

5.2 Second scene

The second scene is a buddha model in front of a black plane. The key-light is located in front of the main object (the buddha statue) while the fill-light is located on the right of the scene. Figure(20) shows the light sources position in the scene from above. The key-light is at the bottom, the fill-light is on the right side and the object is in the middle.

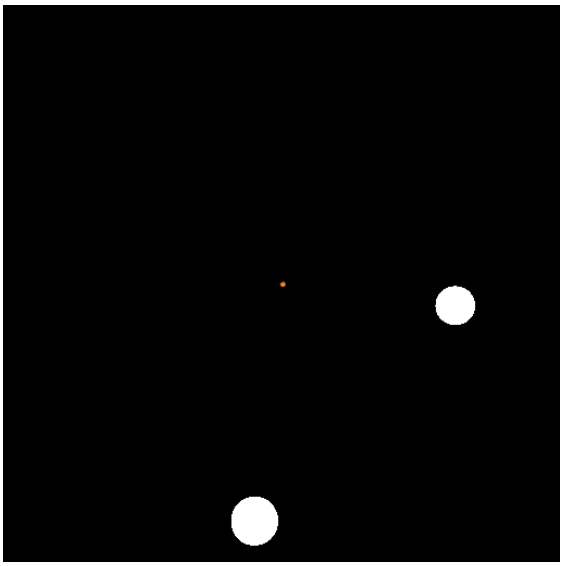


Figure 20: Light source positions in the second scene

Here, we want to convey an aesthetics different from the one of the first example. Our lighting goal is to obtain a low-key aesthetics. Target values are described in figure (21). Low-key images are characterized by a dark, uniform background, and high contrast on the main object. Here, we are using mean luminance and variance target values of 0 for the background. We also use an high value for gradient and variance on the object. Here again, we use a target histogram learned from examples, this time with low-key images.

function term	target value
$f_{meanObj}$	120
$f_{meanBack}$	0
f_{varObj}	80
$f_{varBack}$	0
f_{grad}	100
f_{hist}	low-key signature

Figure 21: Target values for the second scene

We use the same weighting factors as those of the first scene.

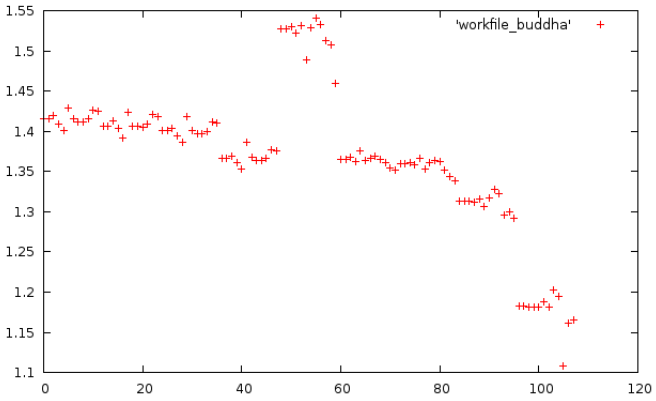
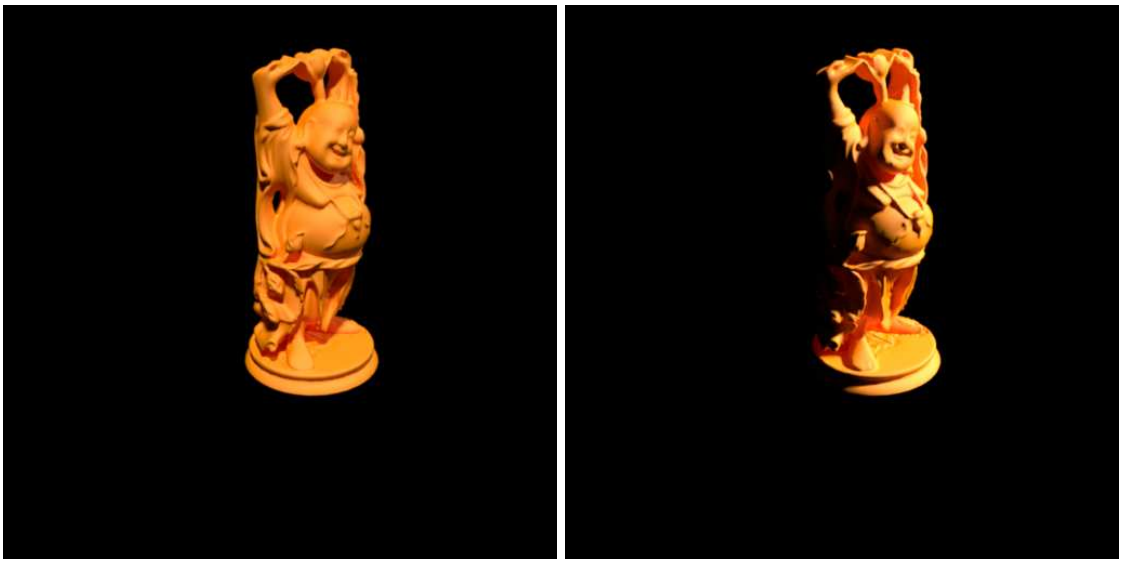


Figure 22: Values of the objective function f_q for each iteration for the second scene

Figure (22) shows the evolution of f_q during the optimization. The optimization process took 9 iterations to converge.



(a) Initial image of the second scene (b) Final image of the second scene

Figure 23: Results for the second scene

The initial configuration of the scene is shown in figure (23a). The final result shown in figure (23b) is consistent with the desired aesthetics specified using the target values. Figure (24) shows the initial and final values of the key-light radius, fill-light radius and key-to-fill ratio. The key-light radius is decreased to reduce front-facing lighting. The key-to-fill ratio is increased in order to increase the flux of the fill-light. The contrast in image (23b) is higher than in (23a) and the edges are visible. Figure (25) shows the mean luminance, variance and gradient amplitude values computed on the final image. If we compare them with the target values in figure (21), we can see that we have strong shading gradient, strong variance on the object and a mean luminance and variance of 0 on the background.

parameter	initial value	final value
key-light radius	3.0	0.1
fill-light radius	3.0	2.7
key-to-fill ratio	0.1	1.2

Figure 24: Initial and final parameters for the second scene

function term	final value
$f_{meanObj}$	104
$f_{meanBack}$	0
f_{varObj}	73.78
$f_{varBack}$	0
f_{grad}	128.19

Figure 25: Values computed on the final image

5.3 Third scene

The third scene represents a bowl of fruits on a piece of cloth. The main object is the bowl of fruit. The scene is rendered with its initial lighting parameters as shown in figure (26). The key-light is behind the camera, and the fill light is on the right side of the scene.

Figure 26: Initial configuration of the scene

With this scene, we want to obtain two different aesthetics using two different sets of target values, as shown in figures (27) and (28).

function term	target value
$f_{meanObj}$	120
$f_{meanBack}$	80
f_{varObj}	60
$f_{varBack}$	20
f_{grad}	10
f_{hist}	low-key signature

Figure 27: First set of target values

function term	target value
$f_{meanObj}$	150
$f_{meanBack}$	120
f_{varObj}	10
$f_{varBack}$	45
f_{grad}	150
f_{hist}	high-key signature

Figure 28: Second set of target values

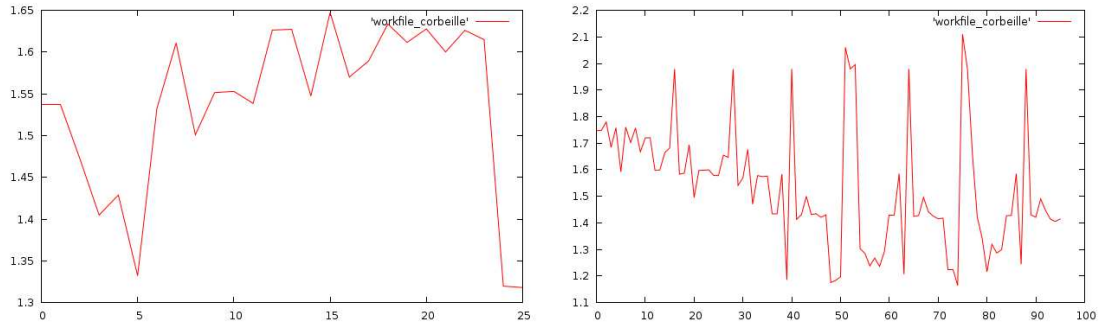
With the first set of values shown in figure (27), we want to maximize the contrast on the bowl of fruits. We want a high variance and a high mean luminance on the main object. Note that the mean luminance target on the background is not 0. Therefore, we are not aiming at a low-key aesthetics. With the second set of target values shown in figure (28), we want to have a high mean luminance on the fruits and on the background, and an important gradient amplitude.

The weighting factors used for this scene are as given in figure (29).

function term	weighting factor
$f_{meanObj}$	0.4
$f_{meanBack}$	0.4
f_{varObj}	0.5
$f_{varBack}$	0.5
f_{grad}	0.2
f_{hist}	0.5

Figure 29: Weighting factors for the third scene

Using the target values presented in figures (27) and (28), we obtain the values for f_q shown in figure (30).



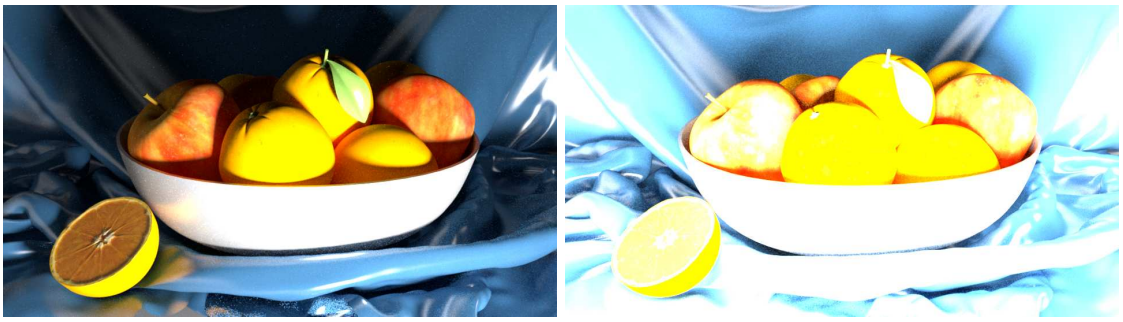
(a) Values of f_q for each iteration with the first set of parameters

(b) Evolution of f_q for each iteration with the second set of parameters

Figure 30: Evolution of the objective function f_q for the third scene with two different aesthetics

In figure (30a) and (30b) we can see that for the two aesthetics, the objective function contains many local minima. Specular highlights can appear on the bowl and on the piece of cloth. This type of material makes the optimization process more complex than with purely diffuse surfaces. However, a minimum is chosen when it meets one of two criteria: when the norm of the gradient of the objective function is below a threshold value (as we can see for the last iterations in figure (30a)) or when a certain number of function evaluation have been performed (as we can see in figure (30)).

After the optimization step, the scene can be rendered with an optimal set of parameters, as shown in Figure (31).



(a) Result for the third scene with the first set of parameters

(b) Result for the third scene with the second set of parameters

Figure 31: Results for the third scene

The final image obtained with the first set of target parameters defined in figure (27) is shown in figure (31a). The contrast and the mean luminance on the bowl of fruits

are high. Details are still visible in the image. The parameters resulting from the optimization process (light source radii and key-to-fill ratio) are shown in figure (32). The values computed on the final image are shown in figure (34).

The image obtained with the seciond set of target parameters defined in figure (28) is shown in figure (31b). The mean luminance is high on both the background and the fruits. However, the contrast on the fruits is low as the luminance seem uniform. The initial and final values for the parameters are shown in figure (33). The values computed on the final image are shown in figure (35).

parameter	initial value	final value
key-light radius	0.1	0.38
fill-light radius	0.1	0.1
key-to-fill ratio	0.1	1.3

Figure 32: Initial and resulting values of the parameters for the third scene with the first set of parameters

parameter	initial value	final value
key-light radius	0.1	1.51
fill-light radius	0.1	0.59
key-to-fill ratio	0.1	0.25

Figure 33: Initial and resulting values of the parameters for the third scene with the second set of parameters

function term	final value
$f_{meanObj}$	130
$f_{meanBack}$	102
f_{varObj}	79.28
$f_{varBack}$	74.89
f_{grad}	150

Figure 34: Values computed on the final image of the third scene with the first set of parameters

function term	final value
$f_{meanObj}$	239
$f_{meanBack}$	218
f_{varObj}	46.86
$f_{varBack}$	27.26
f_{grad}	240.8

Figure 35: Values computed on the final image of the third scene with the second set of parameters

In this scene, with the two different sets of target parameters, the objective function presents many local minima. However, the optimization process converges when the norm of the gradient of the objective function is below a certain threshold: it means that the change of parameters has a small impact on the final result. When the objective function has too many local minima and the gradient criterion is not met, the process is stopped after a fixed number of iterations.

6 Conclusion

In this report, we designed a platform for inverse rendering that accounts for target parameters specified by the user. These target values are used to configure an objective function that has to be minimized. We developed a `python` module that evaluates this objective function on an image of the scene and optimizes the free parameters to meet the user's intent. We considered three free parameters: the key-light and fill-light radii, as well as the key-to fill ratio. For our results, we addressed two image aesthetics used in film and photography (high-key and low-key images) and we considered three different scenes. The result we obtained using this method meet the criteria for the target aesthetics.

However, the metrics accounted for in the objective function are simple: pixel luminance, luminance variance, gradient amplitude, Kullback-Leibler divergence. More complex metrics that account for human visual perception, visual attention and color appearance could bring a better control of the final image's aesthetics. The metrics we used in the objective function are statistics on the pixels' luminance. Using higher-level metrics could yield a better correlation between the objective function's value and the perception of the user.

References

[1] Numpy 1.6.2. `http://www.numpy.org/`, 2013. [Online; accessed May 2013].

[2] Opencv 2.4.4. `http://www.opencv.org/`, 2013. [Online; accessed May 2013].

[3] Scipy 0.12.0. `http://www.scipy.org/`, 2013. [Online; accessed May 2013].

[4] Wenzel Jakob. Mitsuba renderer, 0.4.3. `http://mitsuba-renderer.org/`, 2013. [Online; accessed May 2013].

[5] James T. Kajiya. The rendering equation. In *Computer Graphics*, pages 143–150, 1986.

[6] 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, New York, NY, USA, 1993. ACM.

[7] William B. Kerr and Fabio Pellacini. Toward evaluating lighting design interface paradigms for novice users. *ACM Trans. Graph.*, 28(3):26:1–26:9, July 2009.

[8] Miguel Martin, Roland Fleming, Olga Sorkine, and Diego Gutierrez. Understanding exposure for reverse tone mapping. In *Congreso Espanol de Informática Gráfica*, pages 189–198, 2008.

[9] Panos Y Papalambros and Douglass J Wilde. *Principles of optimal design: modeling and computation*. Cambridge university press, 2000.

[10] Tobias Ritschel, Kaleigh Smith, Matthias Ihrke, Thorsten Grosch, Karol Myszkowski, and Hans-Peter Seidel. 3D Unsharp Masking for Scene Coherent Enhancement. *ACM Trans. Graph. (Proc. of SIGGRAPH 2008)*, 27(3), 2008.

[11] 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, New York, NY, USA, 1993. ACM.

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

[13] Baoyuan Wang, Yizhou Yu, and Ying-Qing Xu. Example-based image color and tone style enhancement. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 64:1–64:12, New York, NY, USA, 2011. ACM.

[14] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 23(4):550–560, December 1997.

[15] 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, New York, NY, USA, 2009. ACM.