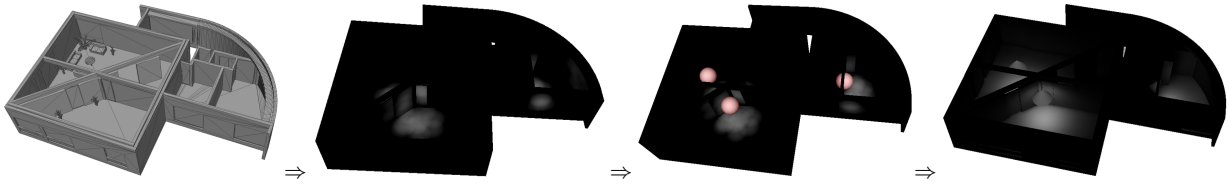
Eurographics Italian Chapter Conference (2010)
E. Puppo, A. Brogni, and L. De Floriani (Editors)

Towards a Voting Scheme for Calculating Light Source Positions from a given Target Illumination

René Zmugg[1], Sven Havemann[1] and Dieter W. Fellner[1,2]

[1]Institut für ComputerGraphik & Wissensvisualisierung, TU Graz, Austria
[2]Fraunhofer IGD & GRIS, TU Darmstadt, Germany



Teaser: The goal is to let the user specify the desired illumination by painting on the surface. Our method computes positions of standard area light sources that produce the desired illumination when later simulated using normal (forward) radiosity.

Abstract

Lighting conditions can make the difference between success or failure of an architectural space. The vision of space-light co-design is that architects can control the impression of an illuminated space already at an early design stage, instead of first designing spaces and then searching for a good lighting setup. As a first step towards this vision we propose a novel method to calculate potential light source positions from a given user defined target illumination. The method is independent of the tessellation of the scene and assumes a homogeneous diffuse Lambertian material. This allows using a voting system that determines potential positions for standard light sources with chosen size and brightness. Votes are cast from an illuminated surface point to all potential positions of a light source that would yield this illumination. Vote clusters consequently indicate a more probable light source position. With a slight extension the method can also identify mid-air light source positions.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—: Radiosity; Color, shading, shadowing, and texture

1. Introduction

We present first preliminary results of a project pursuing the vision of *space-light co-design*. The creation of architectural spaces traditionally proceeds by first designing buildings, and only then the illumination of the buildings. The vision of *space-light co-design* is to design architectural spaces and illumination in one process together. The advantage is that the emotional and artistic impact of the illuminated designed spaces can be judged already in very early design stages. Our vision is that the combination can be edited in a more targeted way to achieve better results faster than before.

Taking the illumination into consideration has become much more complicated today than it was in earlier periods. Of course there are prime historical exemplars of highly sophisticated illumination design, only to mention the *Hagia Sophia* in Istanbul, erected from 532 AD on, sometimes referred to as *the perfect space* in terms of illumination; and also Gothic architecture, which spread over whole Europe from 12th century on. The 'dissolution' of the walls, which no longer had to serve as supporting structure, let enter more sunlight ever before into a building. Mediaeval believers took coloured light as a manifestation of god. This stresses the importance of good lighting design.

In technical terms, sophisticated space-light co-design is easier when assuming a single directional light, i.e., sunlight. When one factor (light) is constant, it is easier to op-

DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2010/041-048

timize the other (space). Today, the situation has radically changed: (a) artificial (electric) lighting allows even greater design freedom, as the lighting industry is extremely innovative today. Thus, optimization obviously becomes more complex. Furthermore, (b) the design space of architectural buildings is currently being dramatically widened by the advent of freeform architecture. Buildings, e.g., from Zaha Hadid or Frank Gehry, are composed of individually designed elements fabricated by CNC methods.

The combination of sophisticated, advanced lighting technology and the possibilities of fancy non-standard buildings together make designing optimal illumination setups tedious and difficult. Maybe a slight change in the building geometry would allow for a drastically better illumination?

1.1. Breakdown into manageable tasks

The guiding vision of space-light co-design must be broken down into manageable sub-tasks. We decided that our first step should be a method for computing the lighting setup from a given user-defined illumination. This is an inverse problem: the radiosity method [Gla94] proceeds just the other way around ('forward radiosity'): First the *lighting setup* is defined (set of area light sources), then the global (non-specular) illumination is computed. The light designer compares the simulation result to the desired outcome, then modifies the lighting setup by experience, and simulates again. This trial-and error workflow is iterated until the result is acceptable.

We want to let the architect specify the *outcome* of a *lighting setup* directly, from which the system should compute the lighting setup (or maybe different alternatives) as fast as possible to allow for short turn-around times when changing the geometry, ideally at interactive rates.

This "inverse illumination" is an ill-posed problem: A light source is defined by *shape*, *area*, *position*, *orientation* and *intensity* of emittance. Furthermore, real light sources do not emit light as uniformly as the standard 'Lambertian' radiosity lights. For many commercial light sources the light distribution is available via internet and can be loaded into a renderer to create much more accurate lighting simulations. Another source of ambiguity is that the specified illumination may be incomplete and even contradictory when light spots are just painted on walls. The method should still be robust enough to compute a *reasonable* lighting setup. This is more important than exact reproduction of the input.

1.2. Solution approach

We make a number of radical simplifications to make the task solvable. Our first very crude assumption is that all objects of the whole environment have the same perfectly diffuse white material, there is no variation in color or brightness. Secondly, we assume a standard light source with arbitrary but fixed area and intensity. Since the computation depends only on the product of area and intensity, we can also think of a point light source with given radiant intensity. The following **core idea** leads to our voting method:

> Let **s** be a not-so-bright surface point on the wall, E_s its brightness, and assume it is known that the room is illuminated by a single bright 100 watts light bulb B that is attached to a wall. Then the potential positions **c** of B are those that are in the right distance from **s** such that they produce exactly a brightness of E_s at **s**.

Note that the brightness received from a light source also depends from the orientation of the receiver patch (cosine law). Therefore the set C of all potential positions **c** is *not* a sphere, but instead a set of ellipse-like shapes, one for each plane in the scene. the method

1.3. Contribution and benefit

In this paper we present preliminary results from our project. We want to propose and discuss and approach for disambiguating the ill-posed inverse illumination problem. Although we present only the core idea and more work is needed to make it practical, we believe the idea is still worth publishing since it may also be useful in other contexts.

Our contribution is (i) the idea from the previous subsection, which to the best of our knowledge is new, (ii) its development into a voting method in the next sections, and (iii) the exploration of its strengths and weaknesses.

The benefit of our desired method is that architects can directly specify the outcome of a lighting setup, instead of indirectly, with tedious simulate-compare-improve optimization cycles. The goal is faster turnaround times, so that more space-light variants can be tested in shorter time, thus hopefully improving the overall design quality of space and light.

2. Related Work

Radiosity and other illumination methods are well covered in literature, but there is not much literature on the inverse illumination problem. There are different ways to formulate and solve the problem.

One formulation takes light source positions as input and calculates the light parameters with respect to the desired target illumination. Schoeneman et al. [SDS*93] introduced a user interface for painting a desired illumination on all Lambertian surfaces in the scene. Their method computes the color and intensities of a fixed number of light sources with given position. These methods require knowing the light source positions. Our method computes these positions, but assumes a standard light source. Other methods use local illumination features and geometry to determine possible light source positions. Poulin et al. [PF92, PRJ97] proposed

calculating the positions from highlight and umbra information. The main problem is that they use view dependent highlights; multiple light reflections are ignored. Our geometrical approach uses local information as well, but no view dependent information.

The third class of methods tries to solve the inverse problem by optimizing some quadratic error. Tena et al. [TG97] proposed as measure the difference between target illumination and illumination produced by the lighting setup. They use a genetic algorithm to optimize a population of lighting configurations with respect to this error. Number, positions and intensities of the light sources are automatically computed. Drawbacks are that the non-deterministic algorithm leads to high execution times, and the method is dependent on the tessellation; our method is not. The method of Costa et al. [ISC*99] can compute position, direction and angle of light sources. It is based on general reflectance functions and uses simulated annealing for optimizing light parameters. The drawbacks are the risk of non-convergence and the high computational time of the stochastic method. The advantage of this method, as well as of ours, is that light sources placed in midair can be found.

Contensin et al. [Con02] proposed a method where the user defines the target illumination of some patches and also specifies which patches should *not* be considered as possible light sources. The radiosity equation is solved using a pseudo-inverse, which leads to a great number of light sources, and even patches that absorb light. Since this is physically not possible, they minimize the number of light sources and eliminate the absorbing patches in a post-process. Their method obtains good results but depends on the tessellation and is not capable of finding midair light source positions. Ours is faster and (currently) less accurate.

Pellacini et al. [PBMF07] offered a method for achieving the desired lighting for 3D scenes used in computer cinematography. Their optimization approach places point lights, which illuminate the scene as the user defined by painting the desired lighting on the geometry beforehand. Their light placement is not as constrained as ours because obviously they focus on good visual effects rather than physical correctness. Their approach for global illumination purposes provides good results, but needs long execution times.

3. Brightness iso-values as deformed ellipse

3.1. Observations and assumptions

Each illuminated point, whether illuminated directly or indirectly, has to be visible to the light source illuminating it. We exploit this to formulate a voting system that leads to potential light source positions: Each illuminated surface point **s** votes for all potential positions **c** that could be the center of a light source that yields the illumination specified at **s**. It depends on the form factor between both patches, as it determines how much energy can be transferred from one point or patch to another. The form factor $F_{s,c}$ from the area light source at **c** to surface point **s** can be approximated as:

$$F_{s,c} = \frac{\cos(\Theta_c) \cdot \cos(\Theta_s)}{\pi \cdot \|\mathbf{r}\|^2} \cdot A_c \qquad (1)$$

As shown in Fig. 1, Θ_s and Θ_c are the angles between the surface normal vectors and the vector $\mathbf{r} = \mathbf{c} - \mathbf{s}$, the connection of **s** and **c**, and A_c is the area of the light source. The form factor is in fact an integral over the area of the light source. To keep things simple we assume that the form factor is constant over the area when seen from **s**. This assumption holds only if the light source is comparably small or far away from **s**.

Furthermore, we also assume a standard light source with known intensity I. The actual amount of energy transferred from the area light source to **s** is the integral over the area A_c of I times (differential) form factor. Again assuming that the form factor is approximately constant we can replace the integral simply by the product $A_c \cdot I$ of area size times standard intensity. This product we denote I_0, the *radiant intensity* of our standard light source. It is the only parameter needed to describe it. Another assumption we make here is that the whole area of the light source is visible from **s**. Hence, the irradiance resulting at the receiver point **s** from the area light source at **c** can be expressed as:

$$E_{s,c} = \frac{\cos(\Theta_c) \cdot \cos(\Theta_s)}{\pi \cdot \|\mathbf{r}\|^2} \cdot I_0 \qquad (2)$$

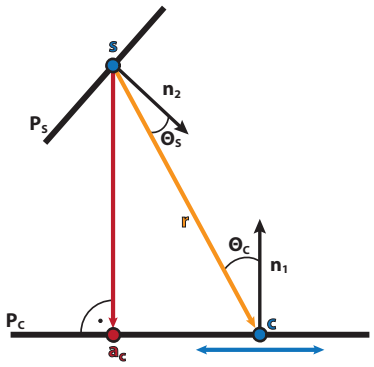


Figure 1: *Arrangement of two planes used for observing form factor distributions.*

Now we made the following experiment: Let the angle between planes P_s and P_c containing points **s** and **c** be 90 degrees, assume **c** is one meter away from P_s, and the light source has unit intensity. Fig. 2 shows the (color coded) brightness that arrives at the points of P_s. Every contour line is a set of points receiving the same amount of irradiance. The symmetry of equation 2 suggests that this works also the other way around, i.e., for a point with given brightness the potential light source positions form exactly the same kind of ellipse-like shape.

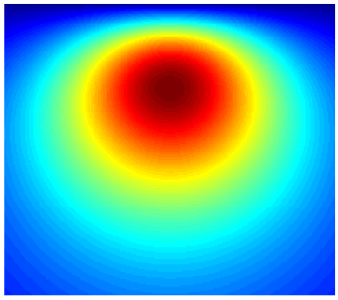


Figure 2: *Form factor distribution for an angle of $\frac{\pi}{2}$ between the normals of the two planes.*

3.2. The deformed ellipse in explicit and implicit form

Our first challenge is that we need these iso-contour lines for each combination of distance and plane angle. In order to trade time for space we require an explicit parametric form of the ellipse-like shapes from Fig. 2 for sampling the contours. Retrieving an well defined explicit representation of equation 2 is hard to achieve. The main problems are that the resulting formulation contains separate formulas of degree higher than four for separate parts of the ellipse-like shape.

Consequently we tried another approach. The shape resembles an ellipse with a deformation along the minor axis, but for $\Theta_s = \Theta_c = 0$ it is a circle. After some experimentation we found that adding to the minor axis a cosine function with double frequency is quite suitable:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_m \\ y_m \end{pmatrix} + \begin{pmatrix} a \cdot \sin(t) \\ b(\cos(t) + d \cdot \cos(2t)) \end{pmatrix} \tag{3}$$

Here, a and b are the major and minor apex lengths, d is the deformation factor, and (x_m, y_m) is the midpoint of the ellipse. This formula allows representing the contour lines occurring in the form factor distribution with only a small error of about 4% of the length of the major axis for typical cases. Ellipses with apex lengths a and b are obtained with $d = 0$, and circles by setting $a = b$ and $d = 0$. Since the method in the next section requires an implicit formulation, we have derived the following formula:

$$b\left(\sqrt{1 - \frac{(x-x_m)^2}{a^2}} + d\left(1 - 2\frac{(x-x_m)^2}{a^2}\right)\right) - (y - y_m) = 0 \tag{4}$$

3.3. Ellipse fitting

The next step is to determine, for a given concrete setting as in Fig. 1, the parameters for equation 3 of an optimally deformed ellipse E_c on plane P_c. Let I_s be the illumination that was specified for the query point $\mathbf{s}$, and let C_c be the set of all points on P_c that are potential positions for a standard light source to cast I_s on $\mathbf{s}$. So E_c is supposed to approximate C_c. The key observation is: Because of the reversibility of light paths, and the symmetry of equation 2, C_c is equally the set of all points on P_c that receive I_s from a standard light source that is placed at $\mathbf{s}$ on P_s ("*reversed setting*").

First of all we need to determine whether C_c is empty: If I_s is very bright, but P_c is far away, it may be that no point on P_c is close enough to cast enough light on s. Therefore the first optimization yields the point of maximum intensity in the reversed setting. Let $\mathbf{a_c}$ be the foot of the perpendicular from $\mathbf{s}$ to P_c (1). It serves as origin for a two-dimensional (x,y) coordinate system on P_c. The unit basis vectors $\mathbf{b_1}$ and $\mathbf{b_2}$ of this coordinate system are:

$$\begin{aligned} \text{major axis direction} &: \mathbf{b_1} = \mathbf{n_1} \times \mathbf{n_2} \\ \text{minor axis direction} &: \mathbf{b_2} = \mathbf{n_1} \times \mathbf{b_1} \end{aligned} \tag{5}$$

These vectors also serve as the axes of the underlying form factor distribution. The point of maximum intensity $\mathbf{q}$ has the coordinate $(0, y_q)$, and y_q can be determined by searching for the largest value of eq. 2 in positive y-direction, starting from $\mathbf{a_c}$. Equation 2 must therefore be expressed in terms of (x,y)-coordinates. This is done by replacing in $\mathbf{r} = \mathbf{c} - \mathbf{s}$ the point $\mathbf{c}$ by a general point $\mathbf{a_c} + x \cdot \mathbf{b_1} + y \cdot \mathbf{b_2}$, thus obtaining

$$\mathbf{r}(x,y) = \mathbf{a_c} + x \cdot \mathbf{b_1} + y \cdot \mathbf{b_2} - \mathbf{s} \tag{6}$$

The form factor intensity equation 2 can then be rewritten as

$$I_s(x,y) = \frac{\langle \mathbf{n_1}, -\mathbf{r}(x,y) \rangle \cdot \langle \mathbf{n_2}, \mathbf{r}(x,y) \rangle}{\pi \cdot \|\mathbf{r}(x,y)\|^4} \cdot I_0 \tag{7}$$

Note that $\mathbf{r}(x,y)$ is not normalized, so it is necessary to divide by the length of $\mathbf{r}(x,y)$ twice more. The optimization $y_q = \text{argmax}_y(I_s(0,y))$ is simplified by the fact that I_s is a convex, symmetric function, i.e., binary line search can be used.

If $I_s(0,y_q) > I_s$ then the set C_c is not empty. Now a second iterative procedure yields sample points from C_c: Starting from the point $\mathbf{q} = (0,y_q)$ of maximum intensity another k line searches in different directions yield k points $(x_i, y_i), i = 1 \ldots k$, on C_c (see Fig. 3, with $k = 20$ for several different intensities). A third iterative procedure fits a deformed ellipse to these points. Let $E(a,b,d,x_m,y_m,x,y)$ be the left hand side of the implicit ellipse in equation 4; it is zero for points on the ellipse. We then can formulate a mean squared error function:

$$\text{mse}(a,b,d,x_m,y_m) = \frac{1}{k}\sum_{i=1}^{k} E(a,b,d,x_m,y_m,x_i,y_i)^2 \tag{8}$$

This error function needs to be minimized in terms of the ellipse parameters (a,b,d,x_m,y_m). Note that x_m is always zero, and in general $y_m \neq y_q$. The resulting average error of fitting an ellipse to the iso-contour is about 3.67% of the length of the major axis, which justifies using equation 3.

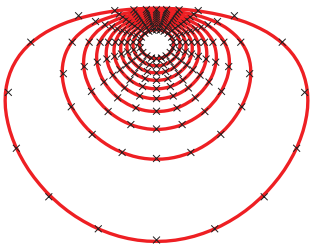


Figure 3: *The crosses are samples of C_c, the set of points with intensity I_c; the red curves are the deformed ellipses (equation 3) fitted to these points.*

4. The voting system

The voting procedure processes a selected set of illuminated sample points drawn from all surfaces in the scene (*vote points*). The vote point density is defined by the number of samples per m^2. Since regular sampling leads to distracting aliasing artifacts, Poisson disc sampling is used.

4.1. Voting procedure

Each plane in the scene has an attached *voting bitmap* with texels of approximately the same size (e.g., 10×10 cm). It is initialized by zero (white). Given a vote point s on a plane P_s with its specified illumination I_s, all planes visible from s are processed. P_s itself is not processed (no self illumination), nor are planes located behind P_s.

For each plane (partly) visible from s the parameters of the deformed ellipse are computed (sec. 3.3). The ellipse is rendered into the voting bitmap using the explicit parametric formulation (eq. 3). Occlusions are handled correctly by checking for blocking geometry for each pixel to be drawn. So in case of occlusion the ellipse is drawn only partly.

As explained, the deformed ellipse is calculated such that when placing the center of an area light source on it, exactly illumination I_s is produced in s. Light sources placed *inside* the ellipse produce larger, and placed *outside* produce smaller illumination. So votes should be cast also to the outside to account for secondary (reflected) light, or for multiple light sources. However, in case of multiple light sources, the problem is to determine how much illumination is produced by which light source; there are infinite possibilities. The danger is a combinatorial explosion when searching through different possible combinations of multiple light sources. So for pragmatic reasons we chose to cast votes only on the ellipse itself. The coarse resolution of the voting bitmap acts like a low-pass filter, so that voting clusters still can emerge. To some extent this accounts also for secondary light.

4.2. Light source extraction

After accumulating all votes from all vote points a *vote distribution* is obtained for every plane in the scene. *Vote clusters* are accumulation points in the voting bitmap that indicate more probable potential light source positions. To enhance the local maxima in the resulting textures, the voting bitmap is first smoothed using a Laplace filter, followed by *non-maxima suppression* to reset pixels that cannot correspond to a local maximum. The last step thresholding: All pixels with a vote value greater than the given threshold are returned as potential light source positions.

To avoid clusters of several light source positions around a local maximum the size of the Laplace filter and of the non-maxima suppression can be varied. Furthermore, the threshold and the texture resolution can be adjusted as well.

4.3. Mid-air light sources from non-occluding planes

The vote distribution is calculated for each plane in the scene. Additional non-occluding planes can also be placed that do not block voting rays but have a voting bitmap attached. Vote clusters on these planes then correspond to light sources positioned in mid-air. With multiple such planes stacked over each other, interesting vote patterns appear that also show certain issues with our method (Figs. 4 and 5).

4.4. Interpreting the vote distributions

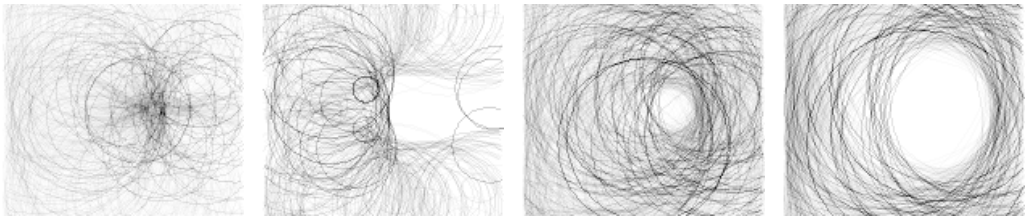
The intensity of the standard light source is chosen globally only once. This parameter is critical and can greatly influence the result (see Fig. 4). In 4(a) most ellipses intersect in a common point and a clear cluster emerges. In 4(b) the standard intensity is too small: The ellipses do not reach the center, the result is a hole surrounded by a front of ellipses.

Increasing the intensity means that the light source needs to be farther away from the vote point to achieve the same illumination. Since plane and vote point are fixed, the only way to enlarge the ellipses is to increase the intensity. In 4(c) the ellipses intersect but no clear cluster emerges. The ellipses are too large because the light source is too large or bright. Decreasing the intensity means decreasing the distance from light source to vote point, which shrinks the ellipses. The last image shows a far too large intensity.

Fig. 5 shows an interesting behaviour with a vote distribution for five consecutive non-occluding planes stacked over each other from ceiling downwards (left to right). The first image shows no clear accumulation point but a ring, as the ellipses do not intersect in a single point; the plane is too far from the vote point (or the intensity too low). In the second image, the ring is almost closed, in the third it is really closed. In image four, the cluster center disperses, and in image five it is completely "out of focus".

5. Optimizing performance

Immediate feedback and interactivity are crucial for unleashing the creativity of the architect. To find good de-



(a) correct (b) too small (c) too large (d) far too large

Figure 4: *Interpretation of vote distributions. In image (a) the vote distribution shows a clear cluster. In (b) the intensity is too small as non-intersecting ellipses form fronts on all sides. In (c) and (d) the standard intensity is too large. There are many ellipse intersections, but no clear cluster emerges. Larger encircled spaces are an indicator for an unsuitable standard intensity.*

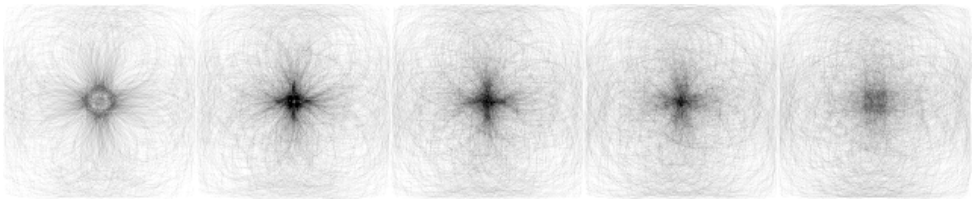


Figure 5: *Vote distribution for five 'non-occluding planes' stacked over each other, from ceiling downwards in 10 cm steps. In the middle, the cluster point is best "in focus".*

sign ideas, many alternatives must be tried out. The complexity of our algorithm is $O(\#\text{sample points} \cdot \#\text{planes}^2)$, where #sample points depends on the overall surface in the scene and the votes per m^2. Because of this complexity we have implemented several optimizations to judge whether the method might be realtime capable.

5.1. Pre-calculations

5.1.1. Poisson disc sampling

Poisson disc sampling both assures a uniform vote point density and eliminates aliasing effects. Each sample has distance $\leq r$ to other samples as it is surrounded by an empty disc with radius r, the *Poisson disc*. Poisson sampling is applied on a per-triangle basis; consequently, the empty disc property cannot be guaranteed along the borders of the triangles. The barycentric coordinates of the Poisson points are precalculated and saved as red, green and blue channels in a texture. So during execution a random texture lookup is necessary to obtain the Poisson samples for a given triangle.

5.1.2. Vote table

Vote processing is time intensive since each vote requires two different optimizations, to find the form factor maximum and to fit the respective iso-contour ellipse. Both optimizations are too time consuming to obtain interactive rates. Furthermore, all this has to be done for quite a large number of votes to guarantee a stable result.

Fortunately, both optimization can be pre-computed using a three-dimensional table: The sampling density of the deformed ellipse depends on the distance between vote point and plane, the angle between the normal vectors, and the distance of the observed illumination value from the form factor maximum. The speedup gained by the table lookup is beyond a factor of 200%. The drawback, however, is that the table contains only a sparse subset of the possible arrangements, so the table must have an appropriate size.

5.2. CUDA implementation

Parallel computing can be used both for distributing vote points via Poisson sampling and for vote processing. One CUDA-thread is started for each Poisson point. The number of CUDA-blocks for vote processing depends on both the number of Poisson points and the number of planes to vote to. As each Poisson point votes for each (visible) plane, the number of CUDA-blocks is the number of planes times the number of vote points. All CUDA-blocks contain a fixed number of CUDA-threads and are executed in parallel. Each thread is responsible for rendering one part of the ellipse into the voting bitmap. Moderately complex scenes that took over half an hour to compute on CPU can be performed in *less than one second* with the CUDA based implementation on a GeForce GTX 480. The scene from our teaser consists of 14612 triangles on 517 planes. Computing the vote distribution for 20 votes per m^2 takes 1.138 seconds. The large scene in Figure 8 consists of 46080 triangles on 649 planes. The computation for this scene takes 15.489 seconds with 20 votes per m^2.

6. Results and Discussion

For a ground truth test we have computed the global illumination (forward radiosity using Maya) for several test scenes with given (standard) area light sources. Our system had to return the known light positions (center of area) to prove its correctness. The example scene in Figure 6 (top) is intentionally similar to the scene used in [Con02]. The left image shows the scene geometry and the area light source positions used for the forward radiosity calculation. The resulting illumination (second image) was used as input for our algorithm. The third image shows that our algorithm finds the correct light positions (red spheres). The last image in this row shows the radiosity calculated using these positions, which is really identical to the result from the initial radiosity pass.

The second row shows our target use case, painted illumination. This time the positions of the light sources are not known, so there is no ground truth. The illumination was painted on floor and desk (second image). Note that this time the calculated positions (red spheres in third image) are in mid-air. The right picture shows the result of the forward radiosity using these light source positions. It yields in fact the desired illumination. However, we have to admit that for obtaining this kind of results some parameter fine tuning was necessary (mainly intensity and threshold).

6.1. Limitations

One inherent problem of our approach occurs with smooth illumination from many light sources. Without clear clusters it is not possible to decide how many light sources exist. Consequently, our approach works best when there are many points for which most of the illumination comes from a single light source. Figure 7 illustrates the effect of four light sources getting closer and closer. When the cluster centers are very close, the vote distribution becomes more ambiguous, revealing in fact a circle of possible light source positions. This vote distribution, however, could also be interpreted as the intensity being chosen too small. With a four times increase of the standard intensity, a single cluster center emerges as the position of a $4\times$ stronger light source.

Another important limitation is that our method is much better suited for polygonal scenes than for scenes with many curved free-form objects.

6.1.1. Approaching the remaining issues

In order to deal with indirect illumination, one idea is to use *multiple levels of indirection*: A first ellipse is drawn to account for the direct light, which is a fraction of the total illumination at the vote point. Then taking the points on this ellipse as vote points, a second generation of ellipses is drawn to account for the indirect illumination. Clusters of the secondary votes yield the light positions for the important case of 1-level indirect illumination. However, this approach requires an estimate of the fraction of indirect illumination.

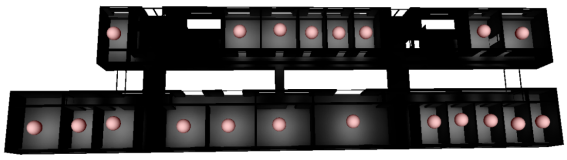


Figure 8: *Successful extraction of* 20 *distinct light source positions with a iterative approach in a large scene resembling a floor in a building of the TU Graz.*

Another possible approach to deal with indirect illumination is a multi-pass technique: After calculating potential light source positions (reverse pass) the scene is illuminated (forward pass), and the forward light is subtracted from the input light. This might in fact yield surface points with negative illumination, which indicates a substantial fraction of indirect illumination. By reducing the intensity, the indirect illumination can be isolated. It is then dealt with in the second pass, thus resulting in a method that works in a ping-pong fashion, iterating reverse and forward calculations. Figure 8 shows a successful extraction of 20 distinct light source positions in a large scene using a variation of this idea.

7. Conclusion and Future Work

We have presented a novel idea for approaching the ill-posed inverse lighting problem. Our method can handle various configurations and proposes in most cases a reasonable lighting setup. Our method is not very mature and still has a number of limitations, but we think it has potential for generalization in various ways. The first thing to do is to remove the simplifications of a single standard light source (see Fig. 8), and to allow the materials of the surfaces to vary in terms of reflectance. Also darker materials should be capable of casting votes, which will probably need to be amplified to account for light damping.

The other direction of research is to work towards more sophisticated lighting setups, which are very relevant for high-end interior design. In particular indirect illumination is very important, which will possibly require a multi-pass approach to account for secondary and tertiary illumination. Great potential has the iteration of inverse radiosity followed by forward radiosity using the light positions computed in the reverse stage. Subtracting the resulting forward illumination from the initial input illumination might allow computing a much more accurate solution, either by modification of the lighting setup, or by adding several smaller lights that cover larger area light sources.

Currently the process also needs fine tuning some parameters such as texel size and vote density. We also anticipate great potential in interpreting the different typical vote dis-

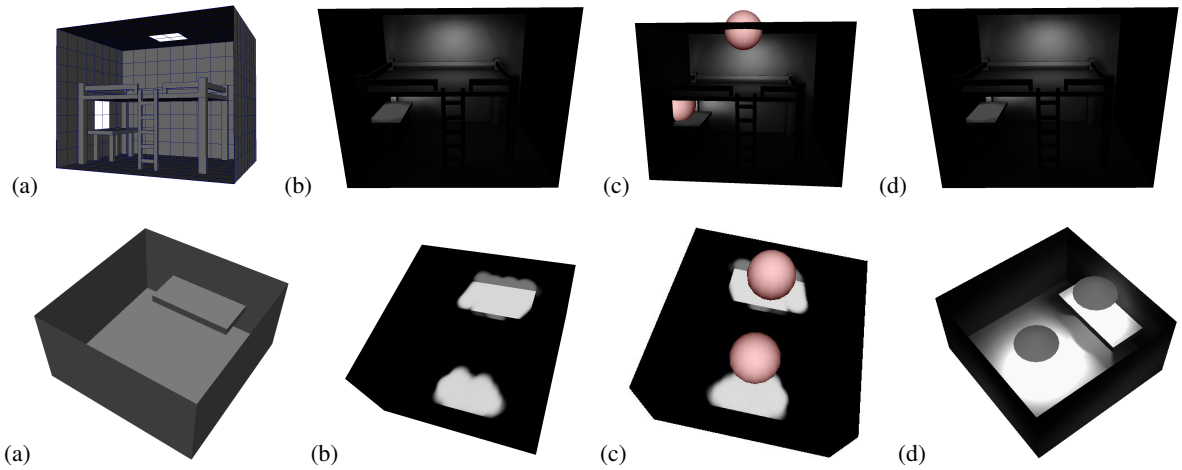


Figure 6: *Ground truth comparison and painted illumination use case. The four columns (left to right) show (a) the scene geometry and known light source positions, (b) the input illumination for our algorithm, (c) the calculated light source positions (red balls), and (c) the forward radiosity obtained using the calculated positions. The ground truth (top) is exactly reproduced despite the scene complexity, the painted illumination (bottom) is indeed achieved.*

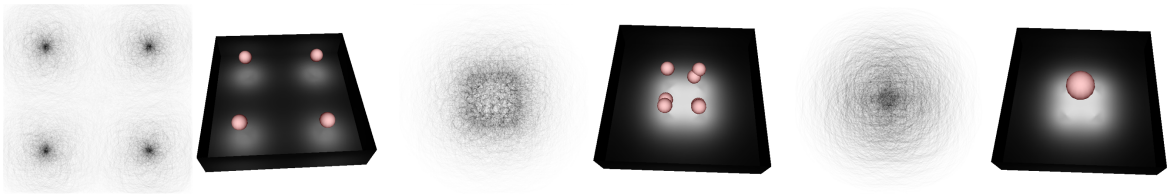


Figure 7: *The overlap problem. When the four light sources (left) are moved closer together (middle), the illuminated regions overlap in such a way that no clear vote cluster emerges anymore. In fact there is a ring of points with approximately the same vote strength. This looks just as if the standard light source was too weak. With a stronger standard intensity, a much clearer cluster center is found from the same input illumination (right).*

tribution patterns (e.g., focus processing of vote clusters). Finally, more work needs to be done to obtain an integrated user interface where forward and backward radiosity are integrated, and which produces results fast enough to support interactive 3D modeling with lighting calculations carried out in parallel, together with a more flexible placement of non-occluding planes for mid-air light sources.

References

[Con02] CONTENSIN M.: Inverse lighting problem in radiosity. *Inverse Problems in Engineering 10* (1 January 2002), 131–152(22). 3, 7

[Gla94] GLASSNER A. S.: *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994. 2

[ISC*99] II I., SOUSA A. A., COSTA A. C., FERREIRA F. N., III F.: Lighting design: A goal based approach using optimisation, 1999. 3

[PBMF07] PELLACINI F., BATTAGLIA F., MORLEY K., FINKELSTEIN A.: Lighting with paint. *ACM Transactions on Graphics 26*, 2 (June 2007), Article 9. 3

[PF92] POULIN P., FOURNIER A.: Lights from highlights and shadows. In *SI3D '92: Proceedings of the 1992 symposium on Interactive 3D graphics* (New York, NY, USA, 1992), ACM, pp. 31–38. 2

[PRJ97] POULIN P., RATIB K., JACQUES M.: Sketching shadows and highlights to position lights. In *CGI '97: Proceedings of the 1997 Conference on Computer Graphics International* (Washington, DC, USA, 1997), IEEE Computer Society, p. 56. 2

[SDS*93] SCHOENEMAN C., DORSEY J., SMITS B., ARVO J., GREENBERG D.: Painting with light. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1993), ACM, pp. 143–146. 2

[TG97] TENA J. E., GOLDBERG I. R.: An interactive system for solving inverse illumination problems using genetic algorithms, 1997. 3