Inverse Lighting for Photography

Stephen R. Marschner
Donald P. Greenberg
Cornell University Program of Computer Graphics
Ithaca, New York

Abstract

We introduce a technique for improving photographs using *inverse lighting*, a new process based on algorithms developed in computer graphics for computing the reflection of light in 3D space. From a photograph and a 3D surface model for the object pictured, inverse lighting estimates the directional distribution of the incident light. We then use this information to process the photograph digitally to alter the lighting on the object.

Inverse lighting is a specific example of the general idea of *inverse rendering*. This refers to the practice of using the methods of computer graphics, which normally are used to render images from scene information, to infer scene information from images. Our system uses physically based rendering technology to construct a linear least squares system that we solve to find the lighting. As an application, the results are then used to simulate a change in the incident light in the photograph.

An implementation is described that uses 3D models from a laser range scanner and photographs from a high-resolution color CCD camera. We demonstrate the system on a simple test object and a human face.

1. Introduction

Photographers have perfected the techniques for developing and printing film with a variety of effects. They commonly manipulate individual parts of a photograph, removing shadows, highlighting features, softening backgrounds, and sharpening detail by optically focusing portions of the image in different ways or exposing regions of the picture in different amounts. These time-consuming manual operations require great artistic and technical skill.

Many of these same operations are now performed using digital filters. This technology is widely available to consumers through commercial software, which performs two-dimensional operations on two-dimensional pixel arrays. Although results can be impressive, some results, such as removing shadows, changing lighting conditions, or modifying the shading on continuous surfaces, are difficult to achieve.

What if these programs had more information about the 3D geometry? In the field of computer graphics, the study of realistic image synthesis has led to physically accurate simulations of image formation. These are normally used to render images from geometric models:

$$\text{model} + \text{lighting} + \text{camera} \rightarrow \text{image}.$$

If the model and camera information were known, and an image was available, one could use the same models inversely to determine the lighting:

$$\text{model} + \text{camera} + \text{image} \rightarrow \text{lighting}.$$

This *inverse rendering* method would yield an approximation of the original 3D lighting conditions. The lighting could then be manipulated in 3D space, aiding in realistic manipulations such as removing shadows or highlights.

This paper presents a technique for using 3D information to help change the distribution of light in a photograph. We begin with a photograph, a 3D model from a laser range camera, and information about the 3D camera position. The computation starts by solving a least-squares system to find the distribution of light incident on the object in the photograph. We then use the result in a forward rendering algorithm to compute the modification required to change the lighting to a new, user-specified configuration. We finish by applying this modification to the original photograph, producing an enhanced image that appears as if it was taken under the desired lighting (Figure 2).

Solving for lighting in an inverse system is not new to computer graphics, although it may be new to photography. One application is lighting design [10, 7], in which a configuration of lights is computed from a specification of desired illumination. Other uses of the linearity of rendering include representing the phases of sunlight using basis images [8] and work involving lighting design for opera [3] and real time building walkthroughs [1].

2. Inverse Lighting

The problem of inverse lighting can be stated as follows: given a photograph of an object, a 3D model of that object (including its reflectance), and a description of the camera, determine the incident light distribution. We simplify the task by assuming that the object is illuminated only by distant light sources. The incident light distribution describes how much light arrives at the object from every direction—this includes light reflected from the environment as well as direct light. We can think of this light as being emitted by a large sphere surrounding the object. The mathematical form of the answer is a function $L : S^2 \rightarrow \mathbb{R}$.

We approach inverse lighting using the same framework many have used for physically based rendering. A rendering algorithm [4] is a program for generating synthetic images from scene descriptions. Such a program takes three inputs: a description of the relevant light-reflecting surfaces, a description of the light that illuminates the surfaces, and a description of a camera. The computation simulates the physical processes of light reflection and image formation as they would occur in the real scene, and the output is a synthetic photograph. If a renderer uses mathematical models that closely approximate the physics of light transport (as opposed to only generating a visually plausible image), we call it a physically based renderer. We encapsulate a rendering algorithm in a linear operator $\mathcal{R}$ that maps a lighting configuration to an image[1].

We invert $\mathcal{R}$ by taking advantage of its linearity, approximating it by a finite-dimensional linear system. To build this system, we model $\mathcal{R}$'s input—a light distribution—as a sum of basis functions and measure its output—an image—with a finite set of pixels. Let $L_1, \ldots, L_n : S^2 \rightarrow \mathbb{R}$ be a set of n functions on the sphere that will serve as basis functions to represent lighting. A representable lighting configuration is then $L = \sum_{i=1}^n \alpha_i L_i$. The output of $\mathcal{R}$ on this input is the image

$$I = \mathcal{R}(L) = \sum_{i=1}^{n} \alpha_i \mathcal{R}(L_i). \tag{1}$$

Let $v_1, \ldots, v_m$ be the values of the m pixels in the image. Each pixel value is a linear function K_j of I: $v_j = K_j(I)$. Substituting this relationship into (1),

$$v_j = \sum_{i=1}^{n} \alpha_i K_j(\mathcal{R}(L_i)).$$

The equation is now in the form of an m by n linear system; the number of rows is equal to the number of pixels in the image being fit, and the number of columns is equal to the number of basis lights. In matrix form,

$$\begin{bmatrix} \vdots \\ v_j \\ \vdots \end{bmatrix} = \begin{bmatrix} & \vdots & \\ \cdots & K_j(\mathcal{R}(L_i)) & \cdots \\ & \vdots & \end{bmatrix} \begin{bmatrix} \vdots \\ \alpha_i \\ \vdots \end{bmatrix}. \tag{2}$$

Or more compactly, $\hat{I} = \hat{\mathcal{R}}\hat{L}$. Note that column i of $\hat{\mathcal{R}}$ contains the pixel values of an image lit by the i^{th} basis function alone. Solving (2) amounts to finding a linear combination of these n basis images that matches the photograph; the coefficients $\alpha_1, \ldots, \alpha_n$ then describe the incoming light in terms of the basis $L_1, \ldots, L_n$.

[1]That $\mathcal{R}$ is linear simply means that a photograph taken using several light sources is the sum of photographs taken with each light source individually.

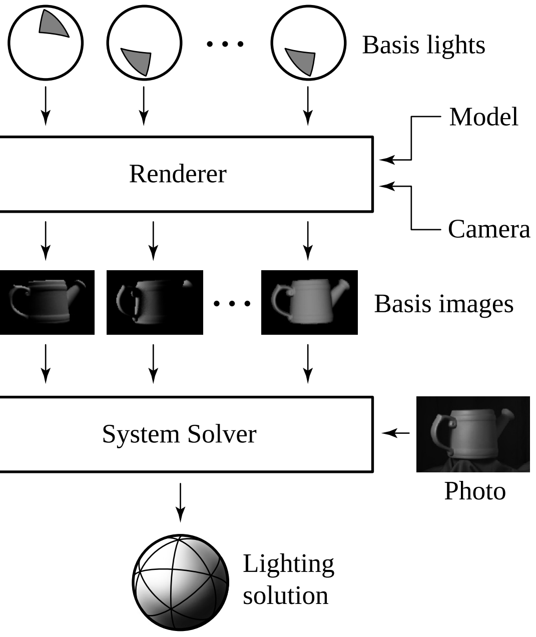

Figure 1: *The data flow in the inverse lighting algorithm.*

In this paper, there are always many more pixels than light basis functions, so the system is over-determined. Further, $\hat{\mathcal{R}}$ is usually ill-conditioned relative to the measurement accuracy of $\hat{I}$ and $\hat{\mathcal{R}}$, especially if the object's BRDF is fairly smooth, since a smooth BRDF serves to blur the information about the incident light distribution. We used a linear least squares algorithm with first-order linear regularization [9], and an optimization using the generalized singular value decomposition [5] to allow the regularization parameter to be adjusted interactively.

The process of inverse lighting is summarized in Figure 1. First, the renderer is given the camera, the 3D model, and each of the basis lights, and it produces a set of basis images. The linear system inversion method is then used to find a linear combination of these basis images that matches the photograph. The coefficients in this linear combination are the lighting solution.

3. Re-lighting

Once we have computed the existing light distribution, we can use that information to modify the lighting in the photograph. Again, we assume that we have a 3D model approximating the object's surface geometry and reflectance, but we expect the photograph to contain details that are missing from the model. We want to preserve these details

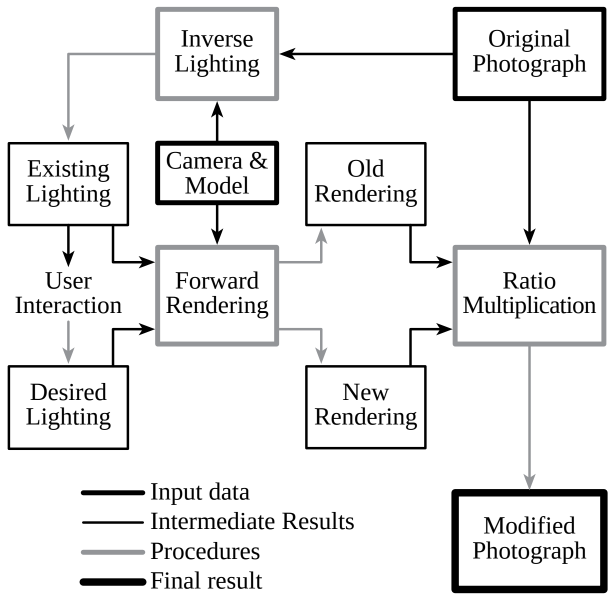

Figure 2: *The data flow in the re-lighting system.*

while changing the lighting on the object.

The renderer, given a description of the existing and desired lighting, can predict the required change in the image by producing two rendered images: one under the existing lighting (the “old rendering”) and one under the desired lighting (the “new rendering”). The old rendering should look like the photograph but lack detail, and the new rendering shows how the modified photograph should look, again without the detail. We use ratios to do the modification—we set the ratio of the modified to existing photograph equal to the ratio of the new to old rendering. That is, we multiply the photograph, pixel by pixel, by the ratio of the new to old renderings, resulting in the modified photograph. Some filtering is required to prevent problems with zero or near-zero pixels in the old rendering.

The complete process of re-lighting a photograph by using inverse lighting is diagrammed in Figure 2. First, inverse lighting is used to compute the existing lighting from the photograph, given the 3D model and the camera parameters. The user modifies this lighting solution to form a new, desired lighting configuration. The rendering system is then used in its normal forward mode to create the new and old renderings from the desired and existing lighting, again given the model and camera. The photograph is then multiplied by the ratio of these renderings to produce the result, a modified photograph.

4. Results

4.1. A rigid object

The first demonstration of the re-lighting algorithm is to modify a photograph of a fairly diffuse, rigid object (a watering can painted with flat paint). We used a Cyberware 3030 range scanner and a Photometrics PXL 1300 CCD camera to obtain the data, and we assumed diffuse reflectance. Figure 3 shows the original photograph (a) and the modified photograph (b), along with the rendered images (c,d) used for re-lighting. The result is convincing save for minor misregistration artifacts and some subtle highlights that were not changed because the diffuse reflectance model did not account for them.

4.2. A human face

The second example, in which we illustrate an application of re-lighting to image compositing, is shown in Figure 3. We began with a snapshot of a person under office lighting, and we wished to paste in a photograph of a second person, taken under different lighting (e), so that the subjects would appear to be standing next to one another. Simply compositing the images would be unconvincing because of the mismatched lighting. Instead, we used inverse lighting to compute the light distribution in the snapshot, then used that information to re-light (f) the second photograph to match the lighting on the two faces before compositing the images (g).

We obtained the 3D models with the same scanner and continued to assume diffuse reflectance. This only approximates the geometry and reflectance of the real subjects, because skin is not a diffuse reflector, and a face is not rigid—the shape changes with facial expression. To accommodate the difference in geometry, we applied an image warp to the rendered images [2]; this lets us take advantage of our 3D information while keeping the features in the model and photograph in correspondence. Also, we filtered the geometry to include only skin areas, since the scanner does not record useful data in the hair. We took the photographs with a Kodak DCS 420 CCD camera.

5. Conclusions and Future Work

We have presented a technique that uses physically based rendering techniques inversely to reconstruct lighting from a photograph and a 3D model of the pictured object. We have demonstrated the application of the resulting data to modifying lighting in digital photographs, both in controlled test cases and in a challenging case that is important to photography, a picture of a human face. We believe that the ability to calculate lighting modifications in 3D space offers far greater potential than 2D image alterations alone.

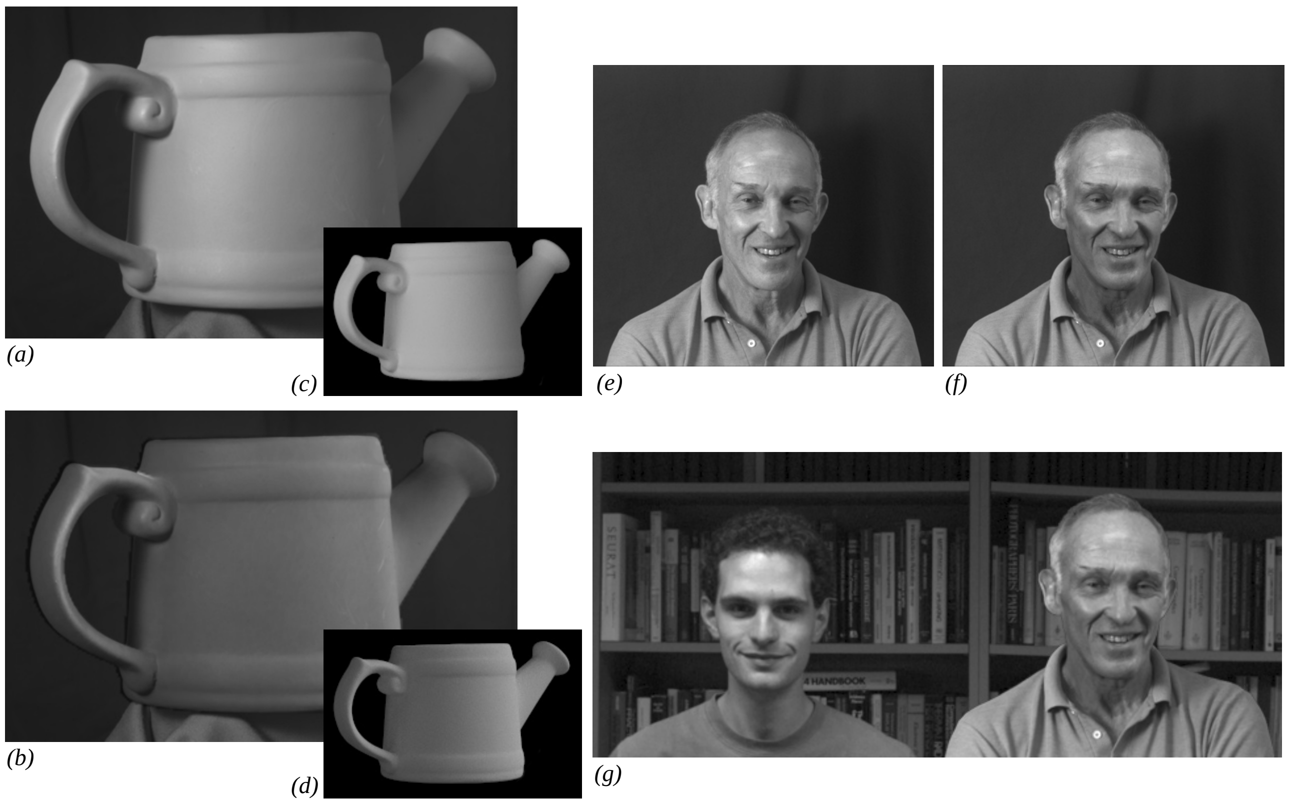

Figure 3: *Results of the re-lighting system. (a-d) Re-lighting a diffuse, rigid object; (e-g) re-lighting a face for a consistent composite.*

The lambertian model for surface reflectance that we used for this work is too simple; real surfaces, including the examples shown here, have more complex behavior. Many models for reflectance exist [4, 6], and our system can handle non-lambertian models without modification.

We used a very simple basis for L, with piecewise constant functions distributed uniformly. Smoother functions or specialized bases that concentrate detail in areas of importance might be more successful.

6. Acknowledgements

Thanks to Pete Shirley for creating `eon`, the rendering code that underlies our system. We gratefully acknowledge generous support of the Hewlett Packard Corporation, on whose workstations this work was carried out.

References

[1] J. M. Airey, J. H. Rohlf, and F. P. Brooks, Jr. Towards image realism with interactive update rates in complex virtual building environments. In *1990 Symposium on Interactive 3D Graphics*, pages 41–50. ACM SIGGRAPH, March 1990.

[2] T. Beier and S. Neely. Feature based image metamorphosis. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 35–42, July 1992.

[3] J. Dorsey, J. Arvo, and D. Greenberg. Interactive design of complex time dependent lighting. *IEEE Computer Graphics and Applications*, 15(2):26–36, March 1995.

[4] A. S. Glassner. *Principles of Digital Image Synthesis.* Morgan Kaufmann, San Francisco, 1995.

[5] G. Golub and C. Van Loan. *Matrix Computations (3rd ed.).* Johns Hopkins University Press, Baltimore, 1996.

[6] P. Hanrahan and W. Krueger. Reflection from layered surfaces due to subsurface scattering. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 165–174, August 1993.

[7] J. K. Kawai, J. S. Painter, and M. F. Cohen. Radioptimization—goal based rendering. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 147–154, August 1993.

[8] J. S. Nimeroff, E. Simoncelli, and J. Dorsey. Efficient re-rendering of naturally illuminated environments. In *Fifth Eurographics Workshop on Rendering*, pages 359–373, Darmstadt, Germany, June 1994.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (2nd ed.).* Cambridge University Press, Cambridge, 1992.

[10] C. Shoeneman, J. Dorsey, B. Smits, J. Arvo, and D. Greenberg. Painting with light. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 143–146, August 1994.