Fitting Complex Materials From A Single Image

Samuel Boivin[*]

University of Toronto
Department of Computer Science
Dynamic Graphics Project (DGP)

1 Introduction

1.1 Role of these course notes

These notes explain in detail material that do not appear in the original paper [3]. Therefore, even if the reader will find some overlap with the paper, it is fundamental to read the original paper [3] to understand the following notes.

1.2 Overview of the problem

Since its origin, Computer Graphics has aimed to simulate an illusion of reality. Rendering algorithms have been developed specifically to generate near-perfect images under realistic illumination conditions. It is often difficult to say if such images are realistic or not because there is no real reference such as a photograph. Moreover, the application may need to create novel viewpoints and/or novel illumination conditions from a sparse set of photographs. This is difficult to achieve without using image-based modeling and rendering algorithms. For example, suppose we want to insert a new synthetic object on top of a real anisotropic mirror inside a real scene. This operation clearly needs to take into account the interaction between the new object and its environment (especially this mirror). This is impossible to do, if we do not have an approximation of the reflectance properties of the real surfaces in the image. Therefore specific algorithms are necessary to recover these reflectance properties from the real images.

Many authors have contributed to the resolution of this problem [13, 16, 24, 23, 25, 17, 18, 28, 7, 29, 14, 15, 22, 21, 10, 9, 20]. The algorithms that they have produced vary greatly and not all can be re-used for our applications. Considerable work has been done for the reflectance estimation of an isolated object in particular illumination conditions [13, 16, 24, 23, 25, 17, 18] . Although these techniques often bring very detailed reflectance information (i.e. a full BRDF in some cases), their goal is more to replace the use of an expensive gonioreflectometer rather than to be able to change the viewpoint and/or the illumination. Recently, several methods have been developed to take into account the interaction between objects inside a real scene, from a sparse set of photographs [7, 29, 14, 15]. Fournier [10] proposed a different approach but with the use of a single image. However, his technique was limited to perfect diffuse environments and was not able to take into account specular surfaces. Our method has the similar ambition to recover an approximation of the BRDF of the surfaces from a single image, including the processing of specular, isotropic or anisotropic surfaces. This is extremely difficult to achieve because it is not possible to compute a full BRDF correctly without having several images, except for trivial cases
We propose a hierarchical and iterative technique that computes the best possible approximation of a real image, using the error computed between the rerendered image and the real one. Each of the new images is generated by making more and more complex assumptions about the reflectance properties of the real surfaces. It is rendered by a global illumination software that takes into account these reflectance changes. The main advantages of our approach are: it does not need any special device to capture the real image (a classical camera is enough), and it estimates the reflectances of all types of surfaces (including anisotropic mirrors) from a single image without any particular constraint

[*]email: boivin@dgp.toronto.edu

for the viewpoint position, the light sources[1] or the orientation of the objects . The goal of our method is to recover an approximation of the BRDF of the surfaces, and to compute the best synthetic image preserving the real properties of the scene (a real mirror has to be simulated as a specular surface and not as a textured surface for example).

2 Background and Related Work

All the techniques and ideas in this paper have been made possible by works about photorealistic rendering including global illumination and ray tracing, image-based modeling and BRDF modeling. However, the most relevant domains deal with *inverse rendering*, *image-based rendering* and *reflectance recovery*. We can split the *reflectance recovery* algorithms into three groups: direct measure of reflectances on the object using a specific device [26, 12, 1, 6], the extraction of reflectances from a set of images [13, 16, 24, 23, 25, 17, 18, 28, 7, 29, 14, 15], and the extraction of reflectances from a single image [22, 21, 10, 9, 20]. The last two parts may be subdivided into two categories, depending on whether the method takes into account energetic interreflections (using a global illumination algorithm for example) or not. A complete overview of these techniques can be found in our paper [3]. We only include here related techniques that use a single image and global illumination for reflectance recovery.

A pioneering work in reflectance recovery from a single image and using global illumination was completed by Fournier et al. [10] in 1993. He proposed to rerender an original image using a 3D representation of the scene (including the positions of the light source and the camera parameters) and a single image of this scene. All the surfaces are considered as perfect diffuse, and they used their reprojection on the real image to estimate their reflectances. A radiosity-based algorithm then computes an image applying these reflectances to a progressive radiosity technique [5] to obtain a new synthetic image.

An extension of the previous method was developed by Drettakis et al. [9]. They proposed an interactive version of the initial paper and added a vision algorithm for the camera calibration and the automatic positioning of the 3D geometrical model. They described a slightly different technique for the estimation of the reflectances of the surfaces and they used a hierarchical radiosity algorithm [11] to compute a new synthetic image close to the real one.

An approach similar to Fournier et al.'s was chosen by Gagalowicz [20]. It included a feedback that compares the real image to the synthetic one. He described a technique to generate a new synthetic image from a single one (except the 3D geometrical model built from two stereo images) using an iterative method that minimizes the error between the real image and the synthetic one. However, this technique is limited to a pure Lambertian approximation of the surface reflectances.

3 Elements of Reflectance Recovery

3.1 The Notion of Group

In our paper [3], we describe a new concept that we called *groups* to solve the problem of not directly seen objects. When we only have one single image, it is very common that several objects are not directly visible in the image from the original viewpoint, or their projection area is too small to be usable for reflectance analysis. It is then almost impossible to compute their reflectance because no information is available in the image. Therefore, every object in the scene is described as a part of a *group* of objects. In this group, at least one object is directly visible in the image and it is assumed to exactly have the same reflectance as the other objects of its group. So, when the reflectance for this object (and/or all other directly seen objects) has been computed, it is propagated to all other objects of the same group.

3.2 Reflectance Model and Data Description

In this technique, we used the Ward's BRDF model [26] for several reasons. First of all, it is a very simple model, and it gives you the opportunity to generate anisotropic surfaces. Secondly, is has been validated on real materials using an experimental device to compute the parameters used in this model. However, it is possible to use any other BRDF model if you can determine a hierarchy of BRDFs for the implementation: the hierarchy should be built following the

[1] In fact, the emittances of the light sources are supposed to be known. However, if it is not the case then there is a method that can recover them automatically as proposed by Fournier et al. [10].

number and the complexity of the parameters used in the model. For example, if you look at the Ward's BRDF model and then at our hierarchy, you see that the hierarchy has been built following theses rules (perfectly diffuse: *natural* case with one parameter, perfectly specular: simplest case with one parameter, non-perfectly specular: enhanced specular case with one iteratively computed parameter, ..., isotropic: three parameters, ...).

A 3D geometrical model is **indispensable** to our technique. It does not matter how this 3D model has been obtained, but its projection on the image must match the scene, and the data must be usable by your rendering algorithm[2]. Some proposals to obtain such models are given in [8] for example. The technique that we used is described in the paper.

3.3 Accuracy of the geometrical model and extraction of the pixel intensities

Since we have the geometrical model, the camera parameters and the original image, we are able to compute *index buffers* using a famous technique called *offscreen rendering* [19]. The index buffers produce a new image that gives the number of the group for every pixel it belongs to. So, when we compute the reflectance we just traverse the index buffer and the original image to sum these pixel intensities [3] The precision required by the inverse algorithm for the positioning of the geometrical model tolerates several pixels of difference between the projection of the model and the real objects in the image. The acceptable number of misclassified pixels depends on the size of the projected object in the original image. For example, if the projection of all objects belonging to the same group has a total number of ten visible pixels, then the inverse algorithm will compute the wrong BRDF when at least about three or four of the ten pixels do not belong to the currently analyzed objects. We use very classical filtering methods, such as edge detectors, edge removal filters and a planar approximation, to reduce inconsistencies with the geometrical model by minimizing the number of pixels assigned to a wrong object.

4 Inverse Rendering from a Single Image

4.1 Overview of the Algorithm

Here, we want to remind you of the main characteristic of our inverse rendering algorithm. The following paragraph is a part of our SIGGRAPH 2001 paper.

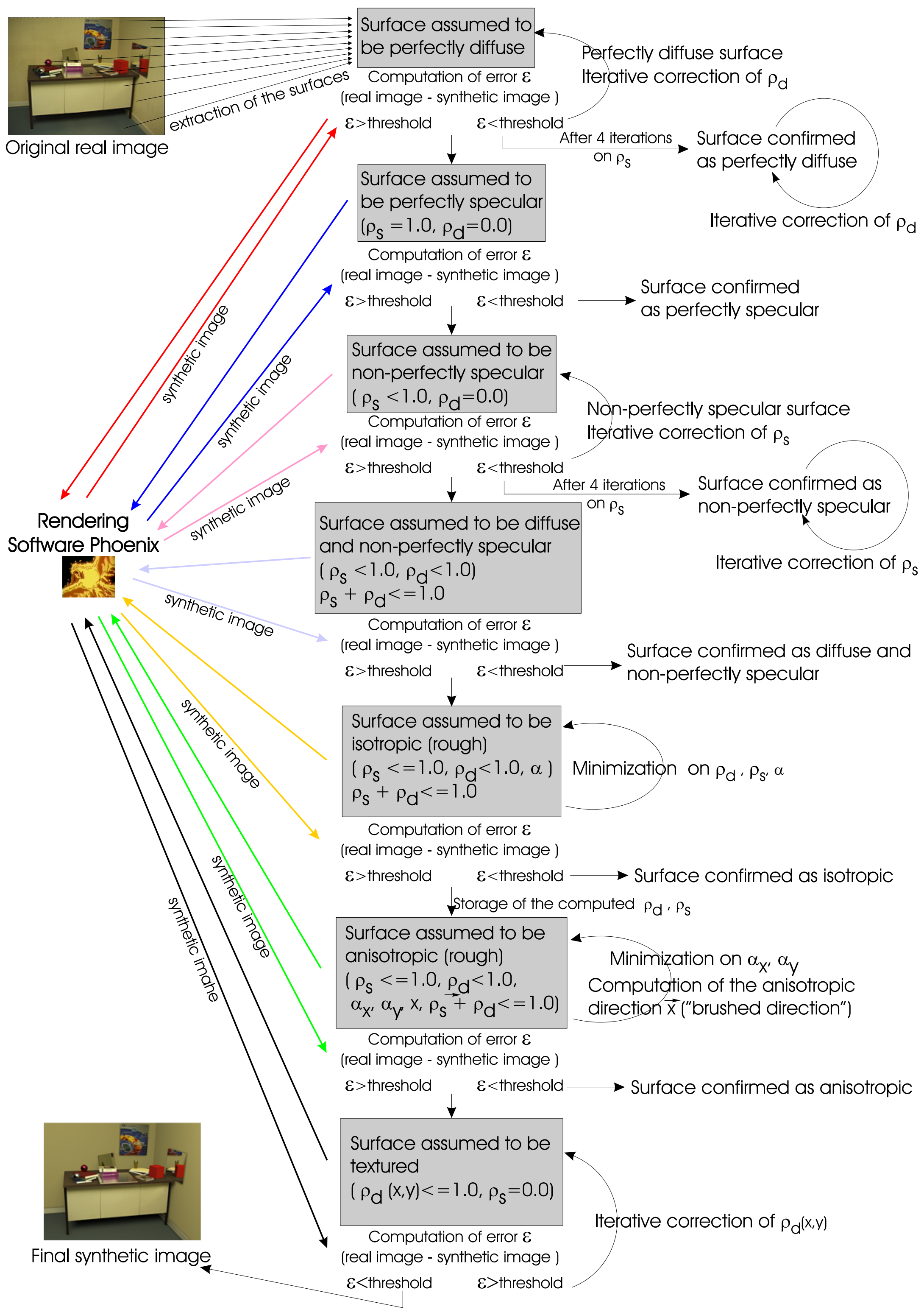
The core of our technique is incremental and hierarchical (see figure 1). It is incremental because the surface reflectances are going to evolve to their optimum value. It is hierarchical because the general algorithm forces the surfaces BRDF to be more and more complex if the error between the real and the synthetic image does not decrease for these surfaces. This algorithm is iterative and will proceed to successive corrections of the surface reflectances by minimizing the error between the real and the synthetic image. Indeed, each computed error for a group of objects having the same photometric properties drives the correction of their reflectance. Our technique successively applies the selected assumption on the group reflectances until the error becomes smaller than a user-defined threshold. The notion of threshold and how to fix its value is discussed in [3, 2]

We start the algorithm with the *perfect diffuse* case without considering texture (the diffuse reflectance of a group is computed by averaging the radiances covered by its projection in the real image). All the surfaces are then considered as perfectly Lambertian, and the rendering software (*Phoenix* in this case[4]) computes a new approximation of the image. If the difference between the real and the synthetic image for a group is greater than a fixed threshold on all the group projection, then the reflectance of this group is considered as *perfectly specular* for the next rerendering iteration. If, after *Phoenix* has recomputed a new image using the new assumption, the error for this group remains big, then its reflectance is simulated as *non-perfectly specular.* We apply the same principle again to change the group reflectance to a *diffuse and specular* one. Until then, all the surfaces were considered with no roughness term (only a ρ_d and a ρ_s were estimated). In the next assumption, if the difference between the two images still produces big

[2]Remember that you must have a rendering software to produce a photorealistic image. This image is supposed to be compared to the original one. So, the choice of the technique is *critical* to our inverse rendering algorithm because on one hand it must provide a **photorealistic** image, and on the other hand this image must be computed on a **reasonably short time** (less than a few minutes).

[3]In fact, the pixel intensities are first converted into radiances. See [3, 2] to understand why and how.

[4]It is possible to use any other global illumination rendering software, such as *Radiance* for example.



Original real image
extraction of the surfaces
Surface assumed to be perfectly diffuse
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Perfectly diffuse surface
Iterative correction of ρd
After 4 iterations on ρs
Surface confirmed as perfectly diffuse
Iterative correction of ρd
Surface assumed to be perfectly specular (ρs =1.0, ρd=0.0)
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Surface confirmed as perfectly specular
Surface assumed to be non-perfectly specular (ρs <1.0, ρd=0.0)
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Non-perfectly specular surface
Iterative correction of ρs
After 4 iterations on ρs
Surface confirmed as non-perfectly specular
Iterative correction of ρs
Rendering Software Phoenix
synthetic image
synthetic image
synthetic image
synthetic image
Surface assumed to be diffuse and non-perfectly specular (ρs <1.0, ρd<1.0) ρs + ρd<=1.0
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Surface confirmed as diffuse and non-perfectly specular
synthetic image
Surface assumed to be isotropic (rough) (ρs <=1.0, ρd<1.0, α) ρs + ρd<=1.0
Minimization on ρd , ρs, α
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Surface confirmed as isotropic
Storage of the computed ρd , ρs
synthetic image
Surface assumed to be anisotropic (rough) (ρs <=1.0, ρd<1.0, αx, αy, x, ρs + ρd<=1.0)
Minimization on αx, αy
Computation of the anisotropic direction x ("brushed direction")
Computation of error ε
(real image - synthetic image)
ε>threshold
ε<threshold
Surface confirmed as anisotropic
synthetic imahe
Surface assumed to be textured (ρd (x,y)<=1.0, ρs=0.0)
Iterative correction of ρd(x,y)
Final synthetic image
Computation of error ε
(real image - synthetic image)
ε<threshold
ε>threshold

Figure 1: General iterative and hierarchical algorithm for reflectance recovery. Each surface of the scene is analyzed separately, depending on the assumption about its reflectance (perfectly diffuse, perfectly specular, etc.). If the assumption is false (the error between the real and the synthetic image is large), then the surface reflectance is assumed to be more complex (hierarchical principle). If the assumption is correct then the surface reflectance is modified accordingly in order to minimize the error between the two images (iterative principle). During each global rerendering iteration, the reflectances of all surfaces are then continuously updated, to take into account the incident energy coming from any surface for which the BRDF has changed (a diffuse surface which became *perfectly specular* for example).

errors, they are considered as isotropic and a roughness factor (α) has to be evaluated. This assumption is extended to anisotropic properties if the user-defined threshold for the error has not been reached. If all assumptions have failed, the group is supposed to be highly textured. Since only a single image is available, it is extremely difficult and sometimes impossible to create a combination between this texture and other reflectance properties (a glossy textured surface for example). This situation is discussed in paragraph 5.2.

5 Implementation details

We present here details regarding particular cases and implementation issues.

5.1 The problem of multiple specular reflections

This is a very common problem in our technique and no details have been given in our paper on the method to solve it. Let us consider the following case: given a real image containing two non-perfect mirrors (A and B) facing each other. So, in the view of A in the original image, B is seen through the specular reflection and vice-versa. When our algorithm attempts to compute the BRDF of A, it is biased by the BRDF of B and vice-versa. You could think that this is not a very common case, but it is. Due to the hierarchical principle of our technique, this case is very common. Indeed, every time that the algorithm needs to go further than the perfectly specular case, it tries the non-perfectly specular case. So when you have two textured books facing each other, they will be considered as non-perfectly specular at the same time in the hierarchy. Another problem is the case where a textured book (A) has a part reflecting in a specular object (B). In this case, when the hierarchical algorithm decide that both of them are being assumed as specular surfaces, the problem is that the part of A reflecting in B prevents our algorithm of finding the properties of B: all the pixels of the projection of A reflecting in B bias the reflectance computation of B.
Of course, our algorithm takes into account all these cases by doing a simple processing. When the algorithm detects a surface as specular (let us call it now A), it first analyzes the content of the indirectly seen object (let say for simplicity purposes that there is only one object seen through the mirror, and let us call it B). We compute an index buffer of the objects seen through the mirror. For each of these objects, we look if their reflectance is being analyzed or if it has already been *confirmed* (see figure 1). If it has already been confirmed, there is no problem and its reflectance will not bias the specular surface. If it is being analyzed then our algorithm proceeds as follows: the object (B) seen through the specular surface is considered as textured. We directly extract its texture from the image regardless of its properties, and no illumination is taken into account for the next rendering step on this *pseudo-textured* object. Now, when the algorithm looks at the properties of the mirror (A), the part of the object (B) seen through the mirror can not bias the reflectance of the mirror because it is considered as a texture. The inverse rendering algorithm then computes the properties of the specular surface (A). The error between the real and the synthetic image determine if the specular assumption is true or not and this error is stored for later use. Then, the algorithm tries the opposite assumption: A is considered as textured and B is a specular object. A new error between the real and the synthetic image is computed and it is stored for later use. Finally, using the **previously stored specular properties** of A and B, the rendering algorithm computes a new synthetic image assuming that both A and B are specular objects. A new error is computed between the real and the synthetic image. The inverse rendering algorithm then compares the three computed errors and chooses the hypothesis which has the smallest one. However, if this error concerns a case where an object has been assumed textured, then **only the specular object** has its properties confirmed. The object assumed textured is still being furthermore investigated to compute its real properties (it could be an isotropic one for example).

5.2 Textured surfaces

When the simulation of a surface as anisotropic still produces big errors in the difference image, we proceed to texture extraction.

Extracting the texture from the real image is an easy task that can be realized using the technique proposed in [27] for example (see figure 2). However, we have to extract this texture while taking into account the fact that it already has received the energy from the light sources, and that the pixels covered by its projection in the real image contain this information. Otherwise, if we send the energy of the light sources to these textures again, they will be over-illuminated. Therefore, we introduce here a notion called *radiosity texture* that balances the extracted texture with an intermediate texture in order to minimize the error between the real and the synthetic image. As for the perfectly diffuse reflectance case, this intermediate texture is computed by an iterative method (see algorithms 1 and 2). All fully detailed algorithms can be found in [2].

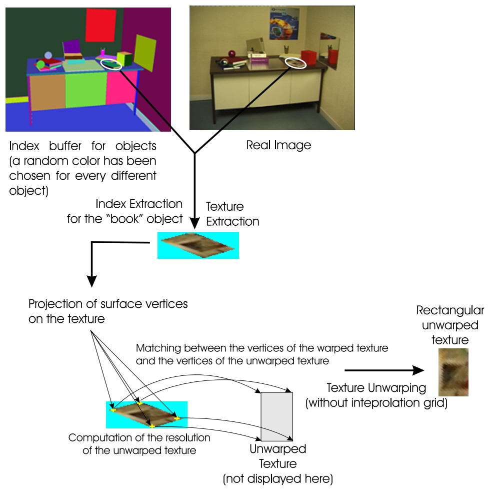

Index buffer for objects (a random color has been chosen for every different object)
Real Image
Index Extraction for the "book" object
Texture Extraction
Projection of surface vertices on the texture
Rectangular unwarped texture
Matching between the vertices of the warped texture and the vertices of the unwarped texture
Texture Unwarping (without inteprolation grid)
Computation of the resolution of the unwarped texture
Unwarped Texture (not displayed here)

Figure 2: Extracting and Unwarping Textures

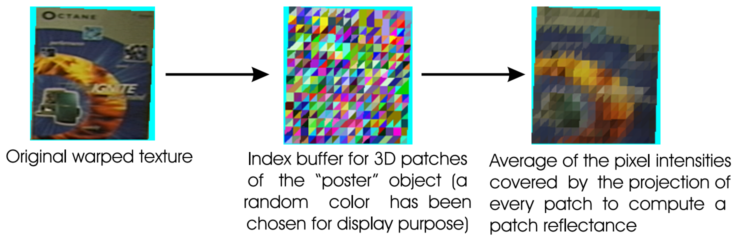

Original warped texture
Index buffer for 3D patches of the "poster" object (a random color has been chosen for display purpose)
Average of the pixel intensities covered by the projection of every patch to compute a patch reflectance

Figure 3: Computation of the diffuse reflectances for a texture in a real image. Here, the subdivision level has been voluntarily increased for clarity purposes.

At the first iteration, the texture used to rerender the image is the texture directly extracted from the real image. At the second iteration, the texture used to obtain the resulting synthetic image is multiplied by the ratio between the newly extracted texture of this synthetic image and the texture of the real image (see algorithm 2). This iterative process stops when the user-defined threshold for textured surfaces has been reached. The textures of the poster and the books in the rerendered images of section 6.2 have been obtained using this technique. The problem of this method is that it computes a texture including the shadows, the specular reflections and the highlights. Typically, suppose that we have a marbled floor on which a sphere is reflected. The texture of this floor in the real image then includes the marble characteristics, its reflectance properties and the sphere reflection including its own reflectance properties. How does one extract the marble characteristics only and independently of the rest of the scene ? This is an extremely hard problem, and according to Y.Sato et al. [25] no algorithm has been proposed yet to solve it using a single image.

6 Results

6.1 Experimental Validation on a synthetic scene

In this section, we present the values obtained for the recovered BRDF of a computer-generated scene (see left image of Figure 5). We compare them to the original known values used to render the original image with *Phoenix* [3, 2].

The first level of the hierarchy in the inverse rendering process computes all the parameters of the surfaces in a straightforward manner. However, the error remains large for the floor and the next levels are tested for this object.

Algorithm 1 Function replacing the anisotropic properties by the texture properties

```
Function Change_Anisotropic_Group_By_Texture(group)
// Is the error for the group bigger than a user-defined threshold ?
if group.$\varepsilon$ $\geq$ anisotropic threshold then
    group.type = texture;
    // The surface is now considered as a textured surface. We extract the object textures
    // and we unwarp them to make them usable by the following function.
    for all objects $j$ do
        Compute_Texture(group.object[$j$], real_img);
    end for
end if
```

Algorithm 2 Pseudo-Algorithm for the computation and the iterative correction of texture reflectances

```
Function Compute_Texture(object, real_img)
 The patch reflectances are computed using the warped original texture
 otherwise the unwarped texture could bias the computation of $\rho_d$
object−>texture = texture extracted from real image using index buffer;
for all patchs $k$ do
    Compute the average diffuse reflectance of patch k (see figure 3) using offscreen-rendering
end for
// Texture unwarping for final rendering
Compute vertex coordinates for all object facets projected on the image;
Unwarp object−>texture using previous coordinates to generate a rectangular image (see figure 2);

Function Modify_Texture(object, real_img, syn_img)
// Computation of the error image
tmp_texture = texture extracted from the the synthetic image;
texture_$\varepsilon$ = $\frac{\text{texture(object,real\_img)}}{\text{tmp\_texture}}$;
// Compute new texture
new_texture = object−> texture $\times$ texture_$\varepsilon$;
// Computation of patch reflectances for the radiosity equation (see [4] for more details)
for all patchs $k$ projecting on new_text do
    Compute $\rho_{d_k}$ as the average of the pixel intensities
    covered by the projection of $k$;
end for
// Unwarping of the new texture for the next rendering
Compute vertex coordinates for all object facets projected on the image;
Unwarp new_texture using previous coordinates to generate a rectangular image (see figure 2);
```

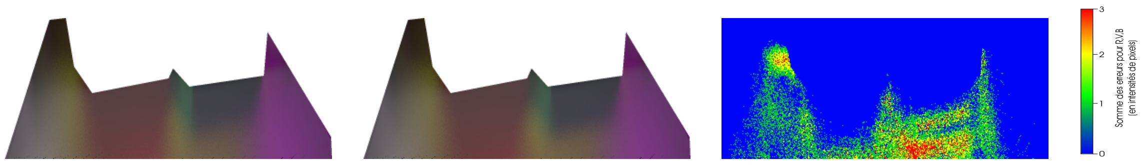


Figure 4: From left to right: original anisotropic floor, floor simulated as an isotropic object, and the error image between the original and the rerendered images.

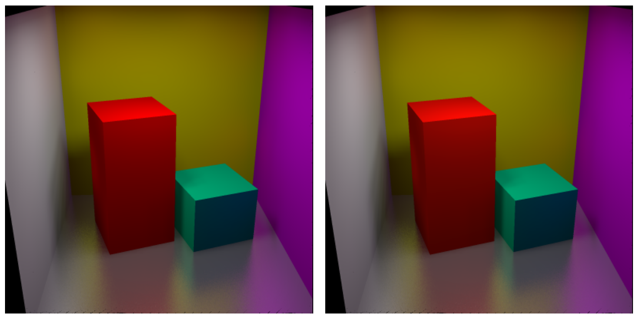


Figure 5: Left: the original computer-generated image. Right: the new synthetic image produced by our inverse rendering technique.

The specular assumptions (perfectly specular, non-perfectly specular, both diffuse and specular) produced large errors forcing the algorithm to choose the isotropy hypothesis. During the isotropy case, a global minimum has been found for ρ_d, ρ_s and α, and the synthetic image is visually very close to the original as shown by Figure 4. However, as we only set 1% for the maximum tolerated error to switch from the isotropy hypothesis to the anisotropy, our method tries to simulate the floor as an anisotropic object.

Surface	Parameter	Real value	Computed value	Comment
Left wall	ρ_d	(0.66, 0.66, 0.66)	(0.65916, 0.66075, 0.66037)	
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Right wall	ρ_d	(0.69, 0, 0.95)	(0.69002, 0.0, 0.95901)	
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Back wall	ρ_d	(0.65, 0.65, 0.0)	(0.64997, 0.65067, $4 \cdot 10^{-7}$)	
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Ceiling	ρ_d	(1.0, 1.0, 1.0)	(1.0, 1.0, 1.0)	See footnote [5].
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Big block	ρ_d	(0.77, 0.0, 0.0)	(0.77002, $2 \cdot 10^{-4}$, $3 \cdot 10^{-6}$)	
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Small block	ρ_d	(0.0, 0.76, 0.26)	(0.0, 0.75802, 0.25912)	
	ρ_s	(0.0, 0.0, 0.0)	(0.0, 0.0, 0.0)	
Floor	ρ_d	(0.1, 0.1, 0.1)	(0.10013, 0.10045, 0.09981)	
	ρ_s	(0.9, 0.9, 0.9)	(0.89909, 0.90102, 0.89903)	
	θ	0.0^o	2.8^o	The total error for
	α_x	0.07	0.06999	θ is 0.77%.
	α_y	0.11	0.1101	

Figure 6: Comparison between the recovered reflectance parameters and their original values.

[5]The ceiling is not directly visible in the original image. When this happens, the algorithm considered this object as a perfect diffuse white object. In practice, if such a case happens, the user should find an object whose photometric properties are close to the ceiling. The ceiling will then be declared in the same group as this object.

6.2 Inverse Rendering

All the following synthetic images have been generated using *Phoenix* as rendering and inverse rendering software. The first synthetic image at the top right of figure 8 has been generated in 37 minutes using the hierarchical algorithm, from the left real photograph. Two specular surfaces have been recovered and simulated as non-perfect mirrors. Neither isotropic nor anisotropic hypotheses have been done thanks to the optimization technique described in our paper [3], and 14 rerendering iterations were necessary to generate the final image.

The inverse algorithm took 4 hours and 40 minutes to produce the image at the bottom right of figure 8. Roughly 4 hours of this time were necessary to recover the anisotropic BRDF of the aluminum surface. The final rendering stage took 32 minutes to render the final image (100 bounced rays have been used for the anisotropic surface).

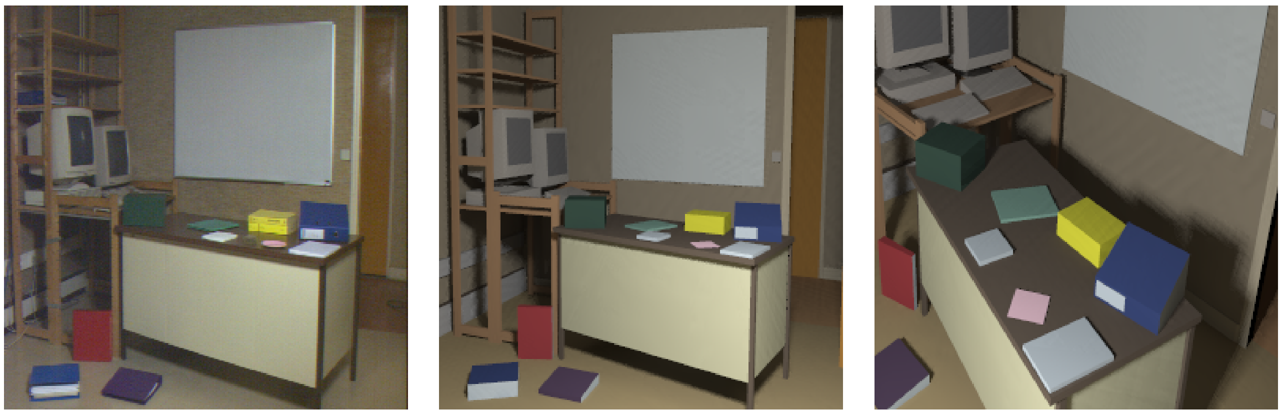


Figure 7: Example of a pure diffuse approximation of a whole 3D scene. From left to right: the original image captured with a camera, the synthetic image and a synthetic image generated under a new viewpoint. The perfectly diffuse assumption is realistic enough for many surfaces, except the computer monitor and the door. Moreover, even if the desk is a real anisotropic surface, a pure diffuse approximation produces a realistic enough result for this object. Note that a book on the left bookshelf has not been modeled. Due to the filtering step and the principle of the method, this does not disturb the inverse rendering case. However, this remains true only for small objects that do not interact much with the real environment. A “very“ large error in the modeling step would definitely produce wrong results.

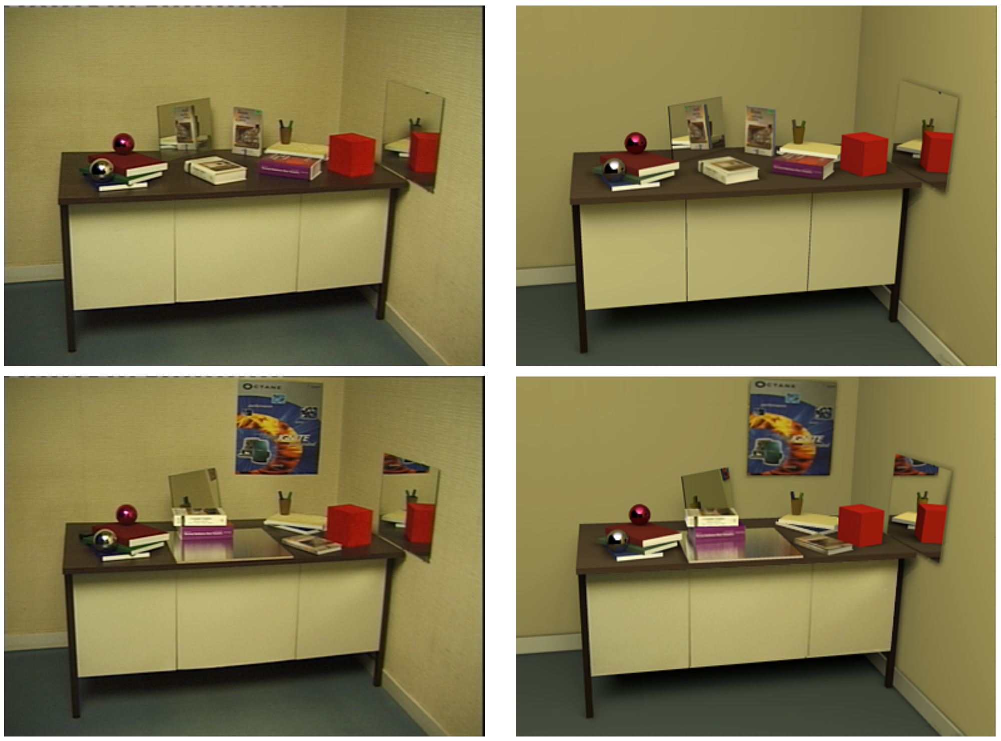


Figure 8: Two different examples of synthetic images (right) rerendered from a single real image (left). We remark that the perfectly diffuse assumption is realistic enough for many surfaces (including the walls, the floor, the desk, etc.).

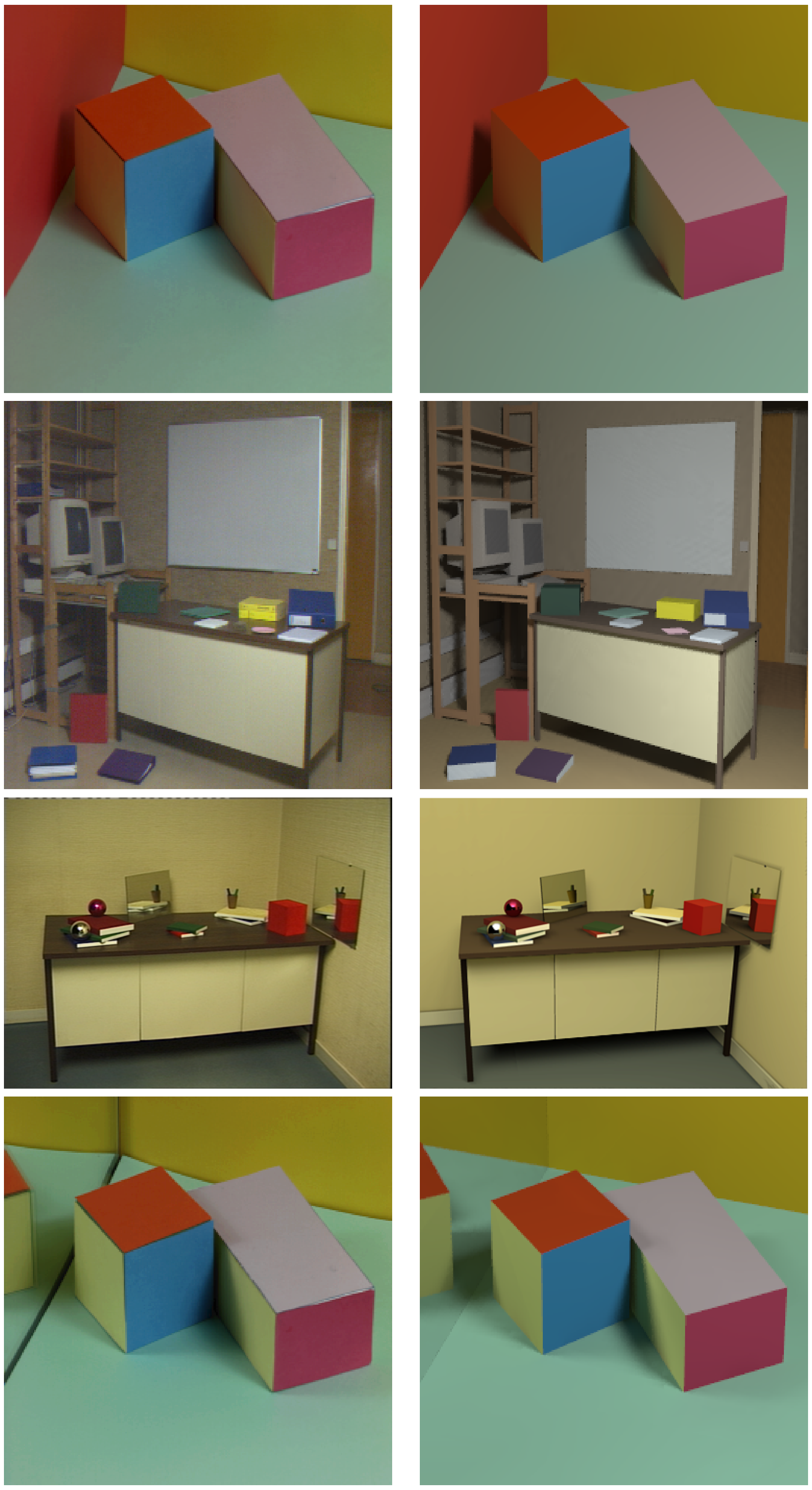


Figure 9: Some examples of rerendered scenes (right column) including perfectly diffuse and specular surfaces compared to the original image (left column)

7 Augmented Reality Applications

The images of figure 10 show examples of applications in augmented reality. Some synthetic objects have been added such as a small robot and a luxo-like desk lamp. It is also possible to modify the reflectances easily too. New viewpoints can be generated and new illumination conditions can be created as well.

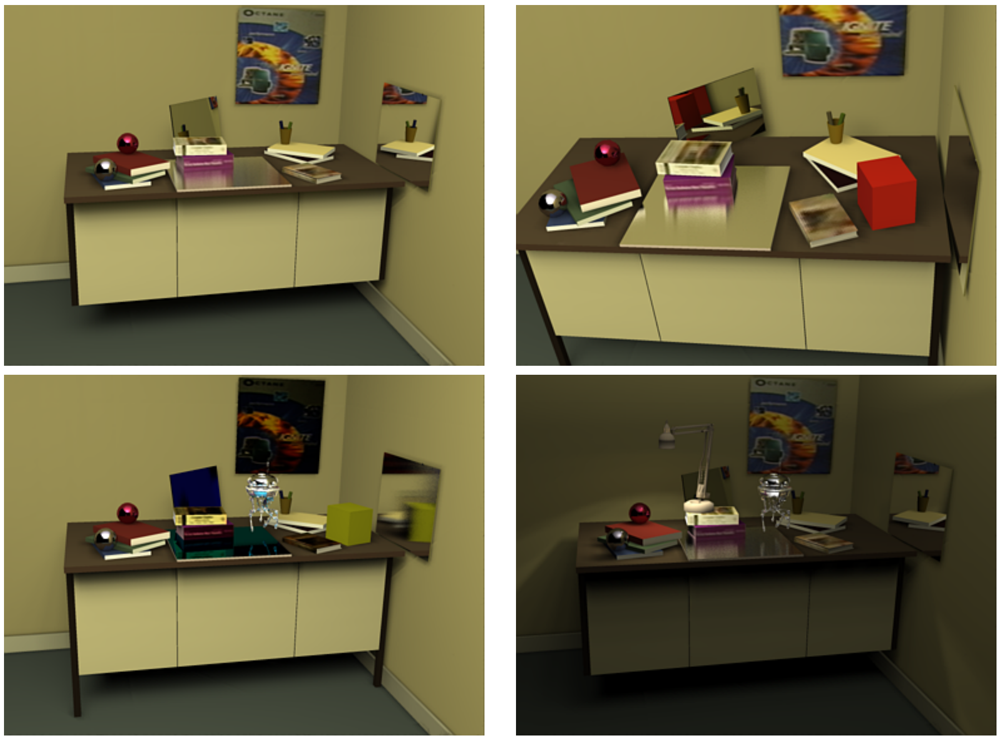


Figure 10: Examples of several augmented reality applications. All these new images were rendered using our global illumination software *Phoenix*, which first recovered the surface reflectances from the bottom left image of figure 8. The top left image shows the original scene removing some objects (the feet of the desk and the red cube). Note that the right mirror has taken into account the modification. The right top image shows the original scene rendered under a novel viewpoint. The bottom left image shows the scene with modified photometric properties, and the addition of an object (a small robot). The bottom right image presents the scene under novel illumination conditions with the addition and deletion of objects.

8 Conclusion and Future Work

In this course, we have presented a new technique that determine an approximation of the reflectance properties of the surfaces of a 3D scene [3, 2]. An incremental and hierarchical algorithm iteratively estimates the diffuse, specular, isotropic and anisotropic reflectance parameters. In a final step, the textured surfaces are considered as a special case of reflectances to be simulated. The method takes as input a single photograph of the scene taken under known illumination conditions as well as a 3D geometric model of the scene. The result is a complete description of the photometric properties of the scene which may be used to produce a photorealistic synthetic image very similar to the real one. We showed that the method is robust and allows for the visualization the original scene from any new angle, with any illumination conditions and with the addition, removal and modification of objects.

Our work has currently some limitations, especially regarding textured surfaces. Until now, we are not able to discriminate the shadows or highlights from an assumed textured surface. In this regard, it will be interesting to extend our method to these cases, although we think that this is a very difficult problem, if one sticks to the single image assumption.

While many challenges remain, we believe that algorithms for recovering an approximation of the reflectances inside a real scene are an important direction of research for both the Computer Vision and the Computer Graphics communities. In Computer Vision, it could be possible for example to use our method to enhance the positioning of mirrors using a minimization algorithm between the real and the synthetic image. Regarding Computer Graphics, we may extend the reflectance recovery algorithm to objects that have more complex photometric properties such as light beam, small fires, caustics, etc. The hierarchical property of our technique offers many possible extensions.

Aknowledgments

The author would like to thank Glenn Tsang for his helpful suggestions regarding the writing of these course notes.

References

[1] R. Baribeau, M. Rioux, and G. Godin. Color reflectance modeling using a polychromatic laser range sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):263–269, February 1992.

[2] Samuel Boivin. *Simulation Photoréaliste de Scènes d'Intérieur à partir d'Images Réelles*. PhD thesis, Spécialité Informatique, École Polytechnique, Palaiseau, January 2001.

[3] Samuel Boivin and André Gagalowicz. Image-based rendering of diffuse, specular and glossy surfaces from a single image. *Proceedings of SIGGRAPH 2001*, pages 107–116, August 2001. ISBN 1-58113-292-1.

[4] M. F. Cohen, D. P. Greenberg, D. S. Immel, and P. J. Brock. An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, 6(3):26–35, March 1986.

[5] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 75–84, August 1988.

[6] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 18(1):1–34, January 1999. ISSN 0730-0301.

[7] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In Michael Cohen, editor, *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 189–198. Addison Wesley, July 1998.

[8] Paul Ernest Debevec. *Modeling and Rendering Architecture from Photographs*. PhD thesis, University of California, Berkeley, 1996.

[9] George Drettakis, Luc Robert, and Sylvain Bougnoux. Interactive common illumination for computer augmented reality. In Julie Dorsey and Philipp Slusallek, editors, *Eurographics Rendering Workshop 1997*, pages 45–56. Springer Wien, June 1997.

[10] Alain Fournier, Atjeng S. Gunawan, and Chris Romanzin. Common illumination between real and computer generated scenes. In *Graphics Interface '93*, pages 254–262. Canadian Information Processing Society, May 1993. Held in Toronto, Ontario, Canada.

[11] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):197–206, July 1991.

[12] Konrad F. Karner, Heinz Mayer, and Michael Gervautz. An image based measurement system for anisotropic reflection. *Computer Graphics Forum*, 15(3):119–128, August 1996.

[13] G. Kay and T. Caelli. Inverting an illumination model from range and intensity maps. *CGVIP: Image Understanding*, 59:183–201, 1994.

[14] C. Loscos, M. C. Frasson, G. Drettakis, B. Walter, X. Grainer, and P. Poulin. Interactive virtual relighting and remodeling of real scenes. Available from www.imagis.imag.fr/Publications RT-0230, Institut National de Recherche en Informatique en Automatique (INRIA), Grenoble, France, April 1999.

[15] Celine Loscos, George Drettakis, and Luc Robert. Interactive virtual relighting of real scenes. *IEEE Transactions on Visualization and Computer Graphics*, 6(3):289–305, 2000.

[16] J. Lu and J. Little. Reflectance function estimation and shape recovery from image sequence of rotating object. In *International Conference on Computer Vision*, pages 80–86, June 1995.

[17] Stephen R. Marschner and Donald P. Greenberg. Inverse lighting for photography. In *Proceedings of the Fifth Color Imaging Conference*. Society for Imaging Science and Technology, November 1997.

[18] Stephen R. Marschner, Stephen H. Westin, Eric P. F. Lafortune, Kenneth E. Torrance, and Donald P.Greenberg. Image-based brdf measurement including human skin. In Dani Lischinski and Greg Ward Larson, editors, *Eurographics Rendering Workshop 1999*. Eurographics, June 1999.

[19] J. Neider, T. Davis, and M. Woo. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Addison Wesley, New York, 1995.

[20] A. Rosenblum. *Data Visualization*, chapter Modeling Complex indoor scenes using an analysis/synthesis framework (André Gagalowicz). Academic Press, 1994.

[21] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Illumination distribution from brightness in shadows: Adaptive extimation of illumination distribution with unknown reflectance properties in shadow regions. In *Proceedings of IEEE ICCC'99*, pages 875–882, September 1999.

[22] Kosuke Sato and Katsushi Ikeuchi. Determining reflectance properties of an object using range and brightness images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1139–1153, 1991.

[23] Yoichi Sato and Katsushi Ikeuchi. Temporal-color space analysis of reflection. *Journal of Optical Society of America*, 11(11):2990–3002, November 1994.

[24] Yoichi Sato and Katsushi Ikeuchi. Reflectance analysis for 3d computer graphics model generation. *Graphical Models and Image Processing*, 58(5):437–451, 1996.

[25] Yoichi Sato, Mark D. Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In Turner Whitted, editor, *Computer Graphics, Proceedings of SIGGRAPH 97*, pages 379–388. Addison Wesley, August 1997.

[26] Gregory J. Ward. Measuring and modeling anisotropic reflection. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 265–272. ACM Press, July 1992.

[27] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos, 1990.

[28] Tien-Tsin Wong, Pheng-Ann Heng, Siu-Hang Or, and Wai-Yin Ng. Image-based rendering with controllable illumination. In Julie Dorsey and Phillip Slusallek, editors, *Rendering Techniques '97 (Proceedings of the Eighth Eurographics Workshop on Rendering)*, pages 13–22, New York,, 1997. Springer Wien. ISBN 3-211-83001-4.

[29] Y. Yu, P. Debevec, J. Malik, and T. Hawkins. Inverse global illumination : Recovering reflectance models of real scenes from photographs. In A. Rockwood, editor, *Computer Graphics (SIGGRAPH '99 Proceedings)*, volume 19, pages 215–224. Addison Wesley Longman, August 1999.