Common Illumination between Real and Computer Generated Scenes

Alain Fournier
Atjeng S. Gunawan
Chris Romanzin

Department of Computer Science
University of British Columbia
Vancouver, BC, V6T 1Z2, Canada
{fournier|gunawan|romanzin}@cs.ubc.ca

Abstract

The ability to merge a real video image (RVI) with a computer-generated image (CGI) enhances the usefulness of both. To go beyond "cut and paste" and chroma-keying, and merge the two images successfully, one must solve the problems of common viewing parameters, common visibility and common illumination. The result can be dubbed *Computer Augmented Reality* (CAR).

We present in this paper techniques for approximating the common global illumination for RVIs and CGIs, assuming some elements of the scene geometry of the real world and common viewing parameters are known. Since the real image is a projection of the exact solution for the global illumination in the real world (done by nature), we approximate the global illumination of the merged image by making the RVI part of the solution to the common global illumination computation. The objects in the real scene are replaced by few boxes covering them; the image intensity of the RVI is used as the initial surface radiosity of the visible part of the boxes; the surface reflectance of the boxes is approximated by subtracting an estimate of the illuminant intensity based on the concept of ambient light; finally global illumination using a classic radiosity computation is used to render the surface of the CGIs with respect to their new environment and for calculating the amount of image intensity correction needed for surfaces of the real image.

An example animation testing these techniques has been produced. Most of the geometric problems have been solved in a relatively *ad hoc* manner. The viewing parameters were extracted by interactive matching of the synthetic scene with the RVIs. The visibility is determined by the relative position of the "blocks" representing the real objects and the computer generated objects, and a moving computer generated light has been inserted. The results of the merging are encouraging, and would be effective for many applications.

Résumé

La possibilité d'intègrer une image vidéo réelle (IVR) et une image de synthèse (IS) augmente considérablement l'utilité des deux. Pour pouvoir aller au-delà de "l'écran bleu" et de la simple composition, on doit résoudre les problèmes de paramètres de caméra, de visibilité, et surtout d'illumination commune. L'ensemble de ces techniques est appelé *réalité augmentée par images de synthèse (RAIS).*

Nous présentons dans cet article des techniques pour approximer l'illumination globale pour des IVR et des IS, en supposant quelques eléments connus pour la géomètrie de la scène et des paramêtres de caméra. L'illumination globale entre les vrais objets est bien sur déjà calculée, et nous utilisons ces résulats dans l'image finale. Les objets de la vraie scène sont remplacés à fin de calcul d'illumination et de visibilité par des "boites" qui les recouvrent. L'intensité obtenue de l'IVR est utilisée pour une première approximation de la radiosité des parties visibles de la boite d'un objet. La reflectance de la surface est estimée par approximation de la lumière ambiante, et finalement le calcul normal de radiosité est fait avec les objets de synthèse, l'approximation des vrais objets et les sources lumineuses synthèse et vraies si elles sont connues.

Une animation testant ces concepts a été réalisée. Nous avons inclus les mouvements de la caméra et le déplacement d'un objet réel. Une source lumineuse de synthèse se déplace également pendant l'animation. Les résultats sont encourageant, et sont utilisables pour des applications pratiques.

Keywords: global illumination, video images, shadows, viewing parameters, computer augmented reality.

1. Introduction

Advances in computer hardware and results in computer graphics made it possible to build graphics workstations which can produce real-time images of good resolution (about 1Kx1K or greater) and moderate complexity (several thousands polygons). In addition, specialized hardware allows users to have real-time video windows on the same workstations. While computer graphics has made great strides towards increased realism in modelling shape and light effects, neither the models nor the hardware is close to the point where it can give real-time realistic (that is good enough to "fool" us) images of our usual environment. It is also clear that in many, if not most, applications there is a real, existing environment within which the modelled objects should eventually be inserted.

The usefulness of the two sources of information, *real video images* (RVI) and *computer generated images* (CGI) can only be enhanced by the ability to merge them freely in real time on the workstation screen. By merging we mean ideally in a way that appears "seamless", where one cannot distinguish between the "real" part and the "synthetized" part. We call the ensemble of techniques to reach that goal *Computer Augmented Reality*. It is different from so-called virtual reality in that computer generated objects and effects are only part of the viewed scene. Of course the result can be viewed in a typical virtual reality display system, but it is not the only intended environment. We stress also that real-time is only a goal, and that for now we are quite content to investigate techniques that work, regardless of their current speed.

The main issues can be divided into *geometric* issues and *illumination* issues. The geometric issues in turn divide into *viewing parameters* and *visibility* problems. The viewing problem is to establish common viewing parameters between the RVI and the CGI. In most applications the goal is to extract the viewing parameters from the RVI and use them in the CGI to position the synthetic camera. We can distinguish between *active* methods, where the real camera is controlled and/or monitored to give the relevant data, and *passive* methods, where the viewing geometry is extracted from the images themselves. The visibility problem consists in resolving mutual priority while compositing the two images. In computer graphics the common solution is the so-called Z-buffer algorithm, and it is now usually implemented in real-time with hardware support. If (and it is a big if) a real-time *depth-map* of the RVI is provided, than the problem is essentially solved. Here again there are active (range finder, sonar, etc) and passive (stereo pairs) methods. It is important to note that the requirements on the precision at which the depth has to be determined varies considerably with the applications. In architecture, for instance, one can be satisfied with placing a building between objects in the background and objects in the foreground, with little or no intersection between these. In a forest environment, the precision required might be equivalent to the radius of a tree. In a mechanical application, a tolerance less than a small fraction of centimetres might be needed.

The illumination problem is to compute illumination (both *local* and *global*) of RVI from computer generated light sources, and of CGI from real light sources. The local illumination problem assume the local illumination conditions are known, and consists in computing the light reflected in the direction of the eye; the global illumination problem is to compute the local conditions for every visible point, given the scene description. Secondary illumination problems, such as shadows, reflections and transparencies, fall somewhere between these two categories. In local illumination, many models have been developed for computer graphics, giving reasonably realistic images. It is only recently that global illumination problems have been seriously investigated, and *radiosity*-based methods are the most popular ways used to solve these. The main new problems here are to identify the positions and characteristics of the lights in the RVI to illuminate the CGI, and to acquire enough knowledge about the geometry of the picture (*eg* getting the "shape from shading") to apply an illumination model and shade it according to the CGI lights. While the local illumination of CGI is rather straightforward after this, the global illumination of CGI is still rather costly. Computing global illumination on the RVI from computer generated lights is in a sense impossible, since it can be affected by surfaces not seen in the real images. One of the challenges is to develop heuristics to "infer" the hidden reflectors from the observed illumination of the real images.

Shadows present a particularly interesting challenge. We can note that in our context, the shadow problem, involving lights, objects and potential shadowing objects from two different origins, has eight "flavours", one of them "free" (when all are real), and some others well controlled (*eg* when all are computer generated). There is also the interesting problem of "reconstructing" the colour of a real surface when a computer generated light forces us to remove a real shadow.

The goal of this paper is to present some preliminary results on the *common global illumination* problem. The basic approach will be to represent objects in the real scene by relatively large blocks and merge these blocks with the computer generated objects and lights to compute global illumination with a classic "radiosity" computation. The results are then transferred on the real image to modify the initial radiosity accordingly. The blocks also help resolve common viewing parameters and visibility problems, though we do not consider this to be necessarily a practical solution.

2. Previous and Related Work

Surprisingly little research has been directed at the problems we just mentioned. In practice the few efforts mixing RVI (or filmed real images) and CGI have used *ad hoc* methods. For instance in the film *The Young Sherlock*

Holmes, a sequence mixes a real scene in a church and the computer animation of a stained-glass knight attacking a real character (the computer animation was done at *Pixar*). In video, DGP, the computer graphics lab at the University of Toronto, in collaboration with the CBC, produced *Le Game*, a mixture of RVI and CGI, with complex camera motion. The viewing problem was solved by using a robot arm holding the real camera, whose parameters were fed into the computer animation program. The compositing used ordinary colour-keying. Nakamae *et al* [Naka86a], are the closest to our goal, inserting computer generated buildings in a real landscape. They included shadow interaction, but the common illumination was approximated by simulating the real lights to render the computer models.

2.1. Compositing

The merging of two CGIs while resolving common visibility is usually done with a variant of classic compositing techniques [Port84a] using depth buffers, best explained by Duff [Duff85a]. While the α channel can help merge pixels with partial coverage, there is no effective way in simple compositing to reconcile illumination between the images. Similarly in merging RVI and CGI, the most effective method, video compositing by *chroma-keying* is of little help. Only a partial form of common visibility can be enforced, and only manually.

2.2. Global Illumination

The last few years have seen a blooming of results on global illumination computations, mostly based on linear systems involving radiosity; see [Fole90a] for a relatively recent survey. Some workers have addressed specifically the problem of adding new elements in a scene for which the global illumination has been already computed. George, Sillion and Greenberg [Geor90a] present a modification of *progressive radiosity* to allow faster radiosity computation for animation sequences where objects are added, moved or removed. To take into account the blocking of light caused by the new objects, they shoot negative energy towards the blocked elements. Bogart produced a video animation [Boga88a] which combined computer objects with a still background picture. Shadows from the computer objects on the real building were computed using ray-tracing and a model of the building.

2.3. Common Viewing and Visibility

Recent efforts in merging computer generated objects of one kind or another while solving the common viewpoint and visibility problems are described by Michael Deering, to accompany his results on virtual reality [Deer92a]. Bajura, Fuchs and Ohbuchi [Baju92a] who mix in real time objects generated from ultra-sounds superposed on images of the patient, and Gleicher and Witkin, [Glei92a], who illustrate their techniques to control the virtual camera by positioning computer generated objects on real objects in the environment. Again in none of these works are serious attempts made to provide a common illumination to the merged scene.

3. Conditions for a Complete Solution

It is pretty obvious that for a complete and accurate solution (at least accurate within common standards in computer graphics) one should be able to extract from the RVI a rather complete model of the objects and light sources, including the illumination models, and then render all of them as CGI. Since this involves solving a reconstructionist version of computer vision and then some, we will not wait for such a solution. We are, however, still working on different aspects of the problem, including retrieving light characteristics from the RVI [Guna91a], identifying highlights [Roma93a] and using depth information for visibility merging and local shadow and shading computations [Kryw93a].

Even once all the relevant information is gathered, re-rendering is still a challenge, since current global illumination algorithms usually do not deal well with specular reflection and scattering media, quite common in RVI. As usual, simplifications and heuristics will have to be used.

4. Approximation Strategies

The general strategy will be to generate a very simplified model of the real scene with few geometric primitives, and use this model mostly to compute approximate common global illumination, and additionally to retrieve viewing parameters and to determine visibility while re-rendering. Our restricted domain of application will be indoor scenes.

4.1. Global Scene Modelling

The scene from the RVI is modelled with few (10 to 100) *boxes* (parallelepipeds in our case, not necessarily aligned with the axes). These boxes are chosen with several purposes in mind. First they will be substitutes for the objects for the global illumination computation. Therefore they should be few for fast computation, but at the same time their sides should be relatively coincident with the big flat areas in the pictures. For indoor scenes, which is our example application, one box will always be chosen so that its sides are the floor, ceiling and main walls. In addition each large object that moves with respect to the room should have its box. Since the box will also be used in the global illumination for shadow computation, and will be used in re-rendering for visibility with respect to the computer generated objects, a way to enhance its usefulness without too much additional modelling effort is to use transparent texture mapping on the sides of the boxes.

Orthographic views of the object(s) enclosed in the box along the six (or less if not all needed) sides are digitized, the outline of the objects in each view is extracted (so far manually) and the texture is made transparent for the part which is not covered by the object. It is not of course equivalent to an accurate model of the object(s) in the box, but will produce more realistic form factors, shadows and visibility determination.

For each box a number of fiduciary points are chosen and measured with respect to the box frame of reference. If the box does not move, these points can be used to help retrieve the viewing parameters. If the box moves these points should be numerous enough to position the box accurately within the room frame of reference if their screen position within a frame is known. Four points at least are necessary, but more are used to compensate for hidden ones and for redundancy.

4.2. Estimation of Surface Reflectance

The relationship between the actual radiance (power/area*solid angle) or radiosity (power/area) and the pixel values are not known, since they depend in complex ways on the characteristics of the imaging and digitizing system. We are, however, only interested in generating CGI that are "matched" to the digitized versions of RVI, and therefore if we assume that the pixel values are proportional to the radiance (a big assumption for real imaging systems, but one that can be corrected for in precisely calibrated systems), then we only have to respect the same proportionality. At this stage we will treat the surfaces seen in the RVI as diffuse reflectors, and therefore for these the radiosity and the radiance is proportional as well. For a given pixel of value p_{xy}, given that the surface element S_i is seen at that pixel, we have:

$$B_i = K\ p_{xy} = E_i + \rho_i \sum_{all\ j} B_j\ F_{ij}$$

where B_i is the radiosity of element i, K is a constant of proportionality common to the whole image, E_i is the emission of element i (0 is only a reflecting surface), ρ_i the reflectivity of element i, and F_{ij} the form factor between j and i (fraction of energy leaving element j which is received by element i). Since we do not need K, we will just assume that in all the subsequent formulas the radiosities and the emissions have been divided by K, and therefore K is no longer appearing explicitly.

Even if we know that $E_i = 0$, we cannot on a single pixel separate the reflectivity (which we need for any global illumination computation) from the contribution of the other elements to the illumination of element i. We can, however, use a few heuristics to help us.

Following [Cohe88a] we can estimate the average form factor:

$$F_{*j} = \frac{A_j}{\sum_{all\ j} A_j}$$

where A_j is the area of element j. We can also estimate an overall interreflectivity factor as:

$$R = \frac{1}{1 - \hat{\rho}}$$

where $\hat{\rho}$ is the average reflectivity (average weighted by area). The latter is easily estimated for a given environment. Given this the *ambient* radiosity can be computed as:

$$B_A = \frac{R \sum_{all\ i} E_i A_i}{\sum_{all\ i} A_i}$$

On the other hand, the ambient radiosity can be estimated by the average radiosity of a pixel divided by the average reflectivity:

$$B_A = \frac{\sum_{all\ xy} p_{xy}}{N\ \hat{\rho}}$$

where N is the total number of pixels. This therefore gives us an estimate of the total power of the light sources present in the scene.

Surface elements are created from the sides of the boxes. The appropriate level of subdivision and how to compute it is currently an active subject of research. We will not address it here, and decide arbitrarily on a level of subdivision (rather low, since we are mainly dealing with correction to the illumination). To determine the radiosity from the real scene, ray-tracing is used to match pixels and surface elements. For each surface element which has visible pixels associated with it, its radiosity is assigned the average of all the pixels it contains. The reflectivity initially assigned to the element is the the average reflectivity, multiplied by the ratio between its radiosity and the the average radiosity of the neighbouring pixels (we take a neighbourhood that contains four times as many pixels as the element). The rational for this heuristic is that if the neighbourhood radiosity is the same, there is no reason to assume anything about the reflectivity of the element. If the neighbourhood is darker, that indicates (but does not prove, of course) that the reflectivity is likely to be higher than average, and similarly if it is brighter. Reflectivity is clamped at 1. The surface elements with no visible pixels are assigned the ambient radiosity and the average reflectivity.

4.3. Modelling of Lights

Often in the case of indoor scenes, the position of the lights, if not their intensity, is known. In this case they are modelled (usually as a collection of polygonal emitters). We can then compute a global radiosity computation with each of the light sources separately assigned a default emission (*wlg* $E_0 = 1$) and the model boxes with their assigned reflectivity. The solution assigns each element a radiosity used as a relative base value. At the end of these computations, for each element i we have:

$$B_i = \sum_{all\ lights\ k} \frac{E_k}{E_0} B_{ik}$$

where E_k is the emission of light k (unknown), B_i is the radiosity assigned to the element from the image, and B_{ik} is the radiosity computed for element i with light k at emission 1. Picking $m \geq k$ elements, we can compute the E_k which best fit our original estimates, and use these in the rest of the computation. Picking all of them for this computation provides a best fit for the distribution of power from the known light sources. Notice that we can constrain the sum to fit the estimate of total light intensity given by the ambient radiosity.

If nothing is known about the lights, then each element in the real scene is considered to be a light source with emission equal to its radiosity and its reflectivity is estimated as before.

The computer generated light sources are modelled as the other computer generated objects, and their emission is chosen depending on the application (but can be compared to the total emission of the real light estimated as given above).

4.4. Correction for Shadowing and Interreflections from CG Objects

The general attitude is to use what is already "computed" in the real scene, and compute only corrections for it. There are essentially two kinds of corrections: modifications due to the computer generated objects which block from the real lights or add interreflection from the real lights, and additional light from the computer generated light source(s). For the former, we deal with them differently depending on whether we have modelled the real light sources or not.

4.4.1. With Models of Real Light Sources

In this case we perform a global illumination computation with the models of the light sources (at the emission estimated for them according to the above section) and the models of the CG objects. The result gives us for each element a B_{i*}, the new radiosity with all the real lights and the CG objets. The ratio B_{i*} / B_i tells us how to modify the radiosity of each pixel which belongs to element i. Notice that a decrease means that CG objects are casting a shadow on the real object, an increase that they are adding interreflection.

4.4.2. Without Models of Real Light Sources

In this case we consider every element to be an emitter. To take shadowing into account, from each element i we shoot negative radiosity towards each other element j equal to the radiosity B_j of the target element. If the negative radiosity is not blocked, nothing happens, but if it is blocked this is subtracted from element i, as it means that the radiosity from j cannot reach i and i should be darker. Of course the form factor F_{ji} is used.

4.5. Global Illumination Computation

The stage is now set to compute the global illumination of the scene with models of the real objects, models of the real light sources (or the objects themselves as light sources if the real light sources are not modelled) and the added computer generated objects and light sources.

The particular method used is *progressive radiosity*, as described in [Cohe88a]. The form factors are currently computed as the analytical version of form factors of discs standing in for the surface elements (which can be parallelograms, or any n-sided regular polygon) as discussed in [Wall89a] and visibility among elements is determined by ray-casting.

The difference with normal CG radiosity computations is that for computer generated objects the whole radiosity is accumulated (which then include light reflected from all sources and interreflected from real and computer generated objects), but for the models of real objects only the additional radiosity ΔB_i (from the computer generated light sources, directly or indirectly) is stored separately.

4.6. Re-rendering

To re-render the scene we use ray-casting. For each ray R_{xy} at pixel xy which hits element i, we follow the following algorithm:

```
if i belongs to a computer
             generated object
    then
        p_xy = k × B_i × C_i × T_i
           + specular component
    else
        p_xy = old p_xy + k × ΔB_i × old p_xy
endif
```

where C_i is the color of the element, and T_i an optional texture value. In effect for RVI elements the old pixel value plays the role of the texture. Note that we compute the results in three separate colour channels (RGB), not because it's right, but because a spectral computation would not affect the steps of our computations.

5. A Commented Example

To illustrate the steps and the results, we have produced an animated sequence.

5.1. The Video Sequence

In the RVI the scene consists of the corner of a room in which a desk supports a workstation monitor, keyboard, mouse and soccer ball. Under the desk is the CPU of the workstation. In the middle of the room is a small square table with a book on it. In the corner of the room is a file cabinet, and on the left a small white box on the floor. The main lights illuminating the scene (out of view for all the frames) are a fluorescent light panel on the ceiling near the far corner, and an incandescent "luxo" style lamp pointing at the ceiling, roughly above the camera.

The whole video sequence (originally captured with an HI8 camera, and directly recorded with a S-video to RGB converter on a SONY LBR-5000 video disc) is about 700 frames of video. During about the first 200 frames the camera zooms out, and for the rest the camera slowly rotates from right to left (from the desk to the corner of the room). About 300 frames into the sequence the soccer ball starts rolling, falls off the desk, rolls on the floor, bumps into the white box on the floor and comes to rest by a leg of the little table. We therefore have both a moving camera and a moving object within the scene. Figure 1 shows a frame near the end of the original RVI.

5.2. Acquiring the Scene Information

After shooting the RVI scene, boxes were decided upon and their coordinates measured. There are 14 boxes for the scene in all, not all of them with 6 sides (for instance the bottom of the file cabinet is not included). The measurements were mostly within 1cm tolerance. The average reflectivity $\hat{\rho}$ was set at 0.7. The position of the fluorescent light overhead was determined, and it was modelled by a rectangle. The incandescent lamp is modelled by two rectangles oriented appropriately. In this test there was no attempt at modelling the colour of the sources, and they both were assigned white (at least the CG concept of white, R=G=B). The diameter of the soccer ball was determined as well, and it is modelled as a dodecahedron.

5.3. Retrieving Camera Positions

During the shooting we measured the position of the real camera for reference. Since the camera moves, however, and since we wanted to test various methods to determine the camera position, we implemented and tested two different methods.

The first method consists in identifying fiduciary points on the image, and solving by least square a system of equations relating the known 3D world coordinates of these point to their measured 2D screen coordinates. The points on the image were identified and positioned manually. This results in a 4x4 matrix that transforms fairly accurately the 3D coordinates into 2D coordinates. Then, using the method described by [Gana84a] we compute the viewing parameters in terms of camera position, look-at vector, up vector, look-at point on the screen and screen scaling factors. In practice this turned out to be not accurate and/or consistent enough to be used through the animation. It still might turn out to be the best approach, but for this example we used instead an interactive matching method. An interactive program displays the wireframe of the model of the real objects, superposed to the RVI. A viewer then manipulates the viewing parameters until a satisfactory match is obtained. This is quite difficult though it is facilitated by keeping constrained as many parameters as possible. In practice we used only about 25 frames of the RVI to be matched, and derived the other viewing parameters by interpolation. It is clear that a method such as described in [Glei92a] would be much better for this purpose. Figure 2 shows an example of an RVI together with the matched wireframe image.

Figure 2. Original frame and overlaid wire model.

5.4. The Computer Generated Objects

The computer generated objects consist of a book added on top of the real book on the small table and a spherical light hanging from the ceiling almost directly above the small table. At the end of the sequence a box comes out of the small box on the floor and wildly changes shape and colour. The light source turns on near the beginning of the sequence and starts swinging when the soccer ball starts rolling. The "morphing" of the small box starts when the ball hits it. The intensity of the additional light source has been chosen to be similar to the intensity of the lights present in the RVI. Figure 3 shows a wire frame of the scene including the computer generated elements.

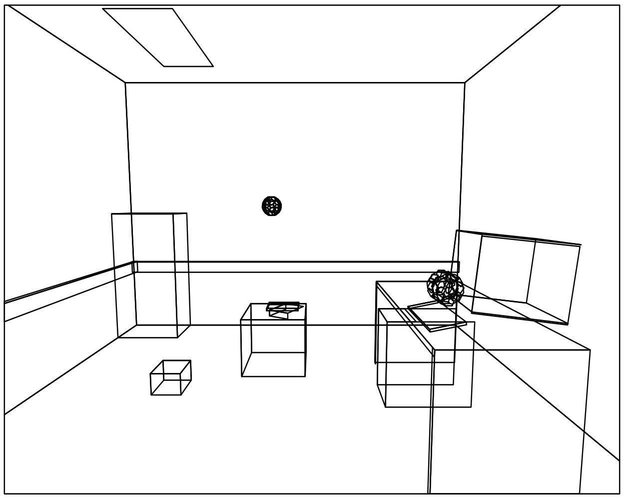

Figure 3. Wire model for the whole scene.

5.5. The Merging

Figure 4 shows the resulting images for the frame shown above.

Figure 3. Final merged image.

6. Results and Analysis

The goal of computing a satisfactory common illumination has been reached. Only by viewing the video can one be persuaded of its effectiveness. One can see from the animation (we did not try to hide the artefacts) that they are two categories of problems. One is the rough quantization imposed by the global illumination computation. This can be partly remedied by higher subdivision, but a more effective solution is to filter the variations in radiosity before applying them to the real image (in effect blurring them) and to the CG objects (which normally would have textures or high quality illuminaion models to add to the radiosity). The other kind of problems stems from the mis-registration of the CGI and the RVI. This results in shift of shadow lines, leaking of shadows and highlights to the wrong objects and appearance of discontinuities due to displaced edges. The method we used to determine viewing parameters is crude, time consuming, and not very accurate. This is potentially a serious problem that even instrumented cameras will not solve, since even a small error can be visible. Here again filtering can alleviate the symptoms, but not suppress them.

Real-time merging is still far away. Computation of the merged images took about 10mn per frame on a 30 MIPS machine (the frame resolution is 646x485). This time is reasonable enough, however, so that one could save rendering time if the alternative is to model the RVI in all its details (that is if our models are good enough).

7. Further Work and Conclusion

As mentioned above we are currently working on many other aspects of the common illumination problem. One can distinguish between "local" and "global" shadows, where local means that the objects are shadowing themselves or nearby objects, and global means that global illumination will take care of them. Of course there is no sharp transition between the two, but it is a useful distinction as the methods to compute them cam be quite different. In this work we ignored the local shadows (such as books on a shelf can cast on each other) since our very coarse modelling of the real scene does not get to that level of details. In this case a depth map can bring enough information to compute local shadows [Kryw93a].

Extracting the chromaticity of the real light source, which in turn can help locate and remove if necessary the specular highlights is also under investigation [Guna91a]. In this case the abundant literature from computer vision on the problem of determining *shape from shading* [Horn88a] will be of help. Exploration of the most relevant techniques is under way [Roma93a].

Acknowledgements

We gratefully acknowledge the support of NSERC through operating grants, an equipment grant and a Strategic grant which considerably facilitated the research described here. The support of the University of British Columbia in establishing a computer graphics laboratory in our department is greatly appreciated, as is the support of IBM Canada, which established GraFiC, an invaluable facility. Pierre Poulin went beyond the call of duty helping model, render and organize for the production of the video. Russ Krywolt provided considerable help in modelling for the sample animation, and several "volunteers", notably Chris Healey and Rob Scharein, also contributed.. We also thank all the members of Imager/GraFiC for their forbearance as their favourite toys were being taken away.

References

Baju92a.
M. Bajura, H. Fuchs, and R. Ohbuchi, ‘‘Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient,’’ *Computer Graphics (SIGGRAPH ’92 Proceedings)*, vol. 26, no. 2, pp. 203-210, held in Chicago, Illinois; 26-31 July 1992, July 1992.

Boga88a.
R.G. Bogart, ‘‘Key Change,’’ *ACM/SIGGRAPH Video Review*, no. 38, 1988.

Cohe88a.
M.F. Cohen, S.E. Chen, J.R. Wallace, and D.P. Greenberg, ‘‘A Progressive Refinement Approach to Fast Radiosity Image Generation,’’ *Computer Graphics (SIGGRAPH ’88 Proceedings)*, vol. 22, no. 4, pp. 75-84, held in Atlanta, Georgia, August 1988.

Deer92a.
M. Deering, ‘‘High Resolution Virtual Reality,’’ *Computer Graphics (SIGGRAPH ’92 Proceedings)*, vol. 26, no. 2, pp. 195-202, held in Chicago, Illinois, July 1992.

Duff85a.
T. Duff, ‘‘Compositing 3-D Rendered Images,’’ *Computer Graphics*, vol. 19, no. 3, pp. 41-44, July 1985.

Fole90a.
J.D. Foley, A. Van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice,* Addison-Wesley Publishing Co, Reading Mass, 1990.

Gana84a.
S. Ganapathy, ‘‘Decomposition of Transformation Matrices for Robot Vision,’’ *Pattern Recognition Letters*, vol. 2, no. 6, pp. 401-412, December 1984.

Geor90a.
D.W. George, F.X. Sillion, and D.P. Greenberg, ‘‘Radiosity Redistribution for Dynamic Environments,’’ *IEEE Computer Graphics and Applications*, vol. 10, no. 4, pp. 26-34, July 1990.

Glei92a.
M. Gleicher and A. Witkin, ‘‘Through-the-Lens Camera Control,’’ *Computer Graphics (SIGGRAPH ’92 Proceedings)*, vol. 26, no. 2, pp. 331--340, held in Chicago, Illinois, July 1992.

Guna91a.
A.S. Gunawan, ‘‘Estimating the Illuminant Color of a Scene from its Image Shading,’’ *Proceedings of the 1991 Western Computer Graphics Symposium*, pp. 29--30, held in Vernon, B.C., April 1991.

Horn88a.
B. K. P. Horn and M. J. Brooks, *Shape from Shading,* MIT Press, Cambridge, MA, 1988.

Kryw93a.
Russ Krywolt, *Production and Use of Depth Maps for Compositing,* Master Thesis, Department of Computer Science, University of British Columbia, May 1993 (estimated).

Naka86a.
E. Nakamae, K. Harada, T. Ishizaki, and T. Nishita, ‘‘A Montage Method: The Overlaying of the Computer Generated Images onto a Background Photograph,’’ *Computer Graphics*, vol. 20, no. 4, pp. 207-214, August 1986.

Port84a.
T. Porter and T. Duff, ‘‘Compositing Digital Images,’’ *Computer Graphics*, vol. 18, no. 3, pp. 253-259, July 1984.

Roma93a.
Chris Romanzin, *Extracting Highlights and Other Lighting Information from Video Images,* Master Thesis, Department of Computer Science, University of British Columbia., May 1993 (estimated).

Wall89a.
J. R. Wallace, K. A. Elmquist, and E. A. Haines, ‘‘A Ray Tracing Algorithm for Progressive Radiosity,’’ *Computer Graphics*, vol. 23, no. 3, pp. 315-324, July 1989.