EUROGRAPHICS 2002 / I. Navazo Alvaro and Ph. Slusallek
(Guest Editors)

On the Integration of Synthetic Objects with Real-World Scenes

F. Abad, E. Camahort, and R. Vivó

Dep. Sistemas Informáticos y Computación, Polytechnic University of Valencia, Valencia. Spain

Abstract

This paper presents a method for integrating synthetic objects in real-world photographs. We take a set of photographs of a real scene and build a simple image-based model. We use high dynamic range images to build an accurate representation of the lighting in the scene. Then we insert a synthetic object into the model and compute its illumination and shading using the lighting information. Illumination changes produced by the synthetic object are also applied to the background plane in the real-world photograph. We show how easy it is to achieve photorealistic results without specialized hardware. Our approach takes advantage of techniques like automatic camera calibration, high-dynamic range image capture and image-based lighting.
We show preliminary results obtained with our application. We also present two improvements that we are currently studying. They are aimed at improving the quality of the resulting images and decreasing the computational resources needed by the process.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Display algorithms I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture

1. Introduction

One of the main goals of computer graphics is the production of photorealistic renderings. A way of achieving this goal is by capturing, generating and storing sets of images. Computer graphics models made of images are called image-based models. One of their advantages is that they can be made of both synthetic images and real-world photographs. This paper is concerned with the integration of synthetic objects in real-world photographs.

Building models from photographs is critical to attain realistic renderings. However, many times it is also necessary to add synthetic objects to a real-world scene. We present in this paper a method that allows the addition of a synthetic object to a real-world photograph. We also show how the object can be illuminated with the light in the real-world scene and how it can affect the background object in the final image. Our method includes capturing multiple images of the real-world scene, building a synthetic object, and adding the object to the scene. The captured images are used for three purposes: camera calibration, building an image-based model of the lighting in the scene, and modeling the background of the scene.

Our method has multiple applications. It can be used in the production of images and animated sequences for cinema, television, advertising, virtual reality and augmented reality [1, 2]. Cinema and television have been the driving forces of the method we present. Our method addresses some of the problems that arise in augmented reality systems, used to improve the sensorial perception of human operators. Those problems are [24]:

- Geometry correspondence: the camera parameters of the virtual objects have to match the camera parameters of the real scene.
- Illumination correspondence: the virtual objects must be lit as if they were in the real scene; they also have to cast shadows.
- Temporal correspondence: the virtual object must move consistently with its environment.

The technique we present in this work shares most of these requirements of augmented reality. To simplify the problem, however, we focus on the generation of still images from static scenes. That is, we do not take into consideration the problem of guaranteeing temporal correspondence.

Our paper is organized as follows. First, we survey pre-

vious work in image-based models and their construction methods. Then, we present a few techniques that we use later as building blocks of our system. Afterwards, we give a step-by-step description of our method. We use a working example throughout the paper, and present several images of both partial and final results. We devote a section to explain our ideas to further improve the quality of the images produced by our system. Our last section presents some conclusions.

2. Previous Work

For many years the production of photorealistic images and animated sequences has been the driving force of computer graphics research [10]. Traditional models represent the objects in a scene using a description of their geometry. Recent models, however, employ images instead of geometry to accurately represent the light in a scene. Such models are known as image-based models, and several well-known computer graphics techniques are based on them: textures, warping, morphing, reflection mapping, and light fields.

Image-based models have the advantage that they can easily integrate synthetic models with images obtained from the real-world. Several authors have tried to achieve this goal [11, 9, 30]. However, they reconstruct the geometry of the real scene, instead of storing an image-based representation. Such approaches are expensive and require a lot of user interaction. For example, in [9] the geometry is reconstructed using a panoramic image of the environment, built from a number of photographs. The user has to define correspondence points between the images, and the topology of the polygons in the scene, assisted by a semi-automatic photogrammetric system.

Augmented Reality (AR) systems also require the integration of synthetic and real-world graphics data. The goal of AR is to improve the perception of reality of a human operator by attaching some information to an image of a real scene. Application areas of AR are telemedicine, microsurgery, entertainment and remote machine operation, among others [1, 2]. The work presented in [24] reconstructs the scene by means of an automatic method based on omnidirectional stereo images. Given two images with a known position, the method extracts some feature points from the image and recovers a rough calculation of the scene, in the form of a triangular mesh. The main problem of this method is that it requires a fisheye lens. A different method applied to augmented reality can be found in [16] and a method applied to video sequences in [13].

A problem in systems that integrate synthetic objects in real-world scenes is the accurate simulation of the lighting in the scene. One of the first papers that dealt with this problem was [20], where the user had to determine the position of the sun when the photograph was taken, in order to compute the shadows cast by the synthetic buildings. Some special effects professionals have developed ad hoc tools that capture light-source positions and model them in rendering systems [14]. Another method to simulate this effect projects the outline of the object along an arbitrary user-selected projection direction. Then, a percentage of the original pixel values in the photograph is subtracted for those pixels where the projection hits [29]. The success of this method relies exclusively on the skills of the user.

The work reported in [18] allows interactive synthetic relighting and remodeling of real environments. This includes removing real lights or objects, and adding virtual ones. They take two sets of photographs, one to recover the geometry of the scene using photogrammetric methods, and the other to estimate the reflectance of the surfaces of the scene. This second set of photographs is taken from the same viewpoint while moving a known light source around the scene. Then, they apply a radiosity algorithm in order to simulate light interactions between the surfaces. This method is difficult to apply to outdoor scenes.

Recently, Gibson [12] presented a system for integrating synthetic objects into photographs at interactive rates. He uses an Open GL rendering pipeline to simulate the shadows produced by the synthetic objects and to illuminate them with the light present in the real scene. The main problem of this solution is that it ignores inter-reflections between synthetic objects, and between these objects and the real-world scene.

The main problem of current systems is that they require a lot of user interaction. Typical approaches work by trial and error, thus requiring a lot of time and effort. An example is the definition of light-source parameters in the real scene, which is crucial for the final result. Also, users suffer from lack of help when building the geometric environment that models the scene where the synthetic objects are to be placed.

Our work tries to address most of these problems and provide sensible solutions to them. Our goal is a system that allows us to easily insert synthetic objects into a real scene. To achieve this goal we propose a set of steps that requires almost no specialized, expensive or hard to find equipment, like a special camera, a custom lens system, or a 3D scanner. Our methodology is based on the work presented in [7] by Debevec. Debevec's work provides the best ratio between quality and technical requirements. The drawback of his technique is that the synthetic objects cannot influence the distant scene, which is the part of the environment furthest from the objects. We propose in this paper a few ideas on how to solve this problem for both static scenes and animated sequences.

3. Background

Our work integrates several techniques into a single application. The application allows us to embed a synthetic object into a real-world scene without a lot of user interaction. This section presents a few basic components that build our

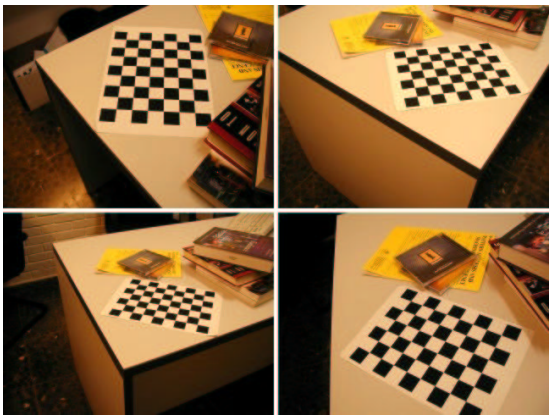

Figure 1: *Photographs used for calibration.*

application. Those components are useful to quickly obtain some very impressive results. Our method has two different phases: the capturing phase and the rendering phase. In the capturing phase most of the work is done by the user. The rendering phase, however, is almost completely automated.

3.1. Camera Calibration

One of the premises of having a photo-realistic result is to obtain a very precise model of the real scene. This model will later be used to integrate the synthetic object with the real scene, that is, to establish geometric correspondence. Camera calibration is very important for any Computer Vision technique that uses images to extract metric information. Camera calibration computes the parameters of the camera at the moment of taking the photograph. Recently, the Computer Vision community has produced several results in automatic camera calibration [31].

Figure 1 shows four photographs used to calibrate the camera we used to build our real-scene model. The calibration object is a 2D set of squares, printed with a laser printer on a piece of paper. Our application uses Intel's OpenCV library [22] to compute the camera parameters. The Open Source Computer Vision Library is an open library that provides basic functions for implementing computer vision applications.

The use of this library releases the user from the tasks of measuring the scene and marking the positions of the calibration object. The application finds the calibration object automatically, then returns the original camera position and the camera's internal parameters (focal distance, distortions of the lenses, etc).

3.2. High Dynamic Range Images

Conventional capture devices, such as film or digital cameras, present a non-linear behavior when digitizing full-range radiance values into pixel values. If the value of a pixel is double the value of another pixel, it is unlikely that the luminous energy of the scene measured at the first pixel will double the luminous energy measured at the second one [6].

On the other hand, the image captured by a photograph is a poor sample of the light present in the real scene. Phenomena like saturation, development and digitization alter the measure of the light in the scene, thus losing illumination information. Furthermore, the dynamic range (the maximum value divided by the minimum) is much lower in those devices than in a real scene.

In order to achieve photo-realism, we must precisely acquire the light in the real scene, specially near the future placement of the synthetic objects. Debevec presents a technique for capturing high dynamic range images using several photographs of a mirrored ball taken with different shutter speeds [6]. High dynamic range images capture the differences in the lighting of a scene more accurately than a single photograph, and thus are a better model for the lighting in the scene.

3.3. Radiance

The goal of photo-realism cannot be achieved if the synthetic objects do not look real, so a global illumination rendering system is required. A system that can handle high dynamic range images as input for lighting synthetic objects is also required. So we use the RADIANCE rendering system [28].

RADIANCE is a suite of programs for the analysis and visualization of lighting design. It is more than a rendering system, it is also a set of tools for simulating the behavior of the light. The results of the simulation are high dynamic radiance maps, but they may also be displayed as color images, numerical values or contour plots. RADIANCE is used by architects and engineers to predict illumination, visual quality and appearance of innovative design spaces. It is also used by researchers to evaluate new lighting and daylighting techniques. This tool is based on stochastic ray tracing, and it is licensed at no cost to users for non-commercial use.

4. Integrating Synthetic Objects in Real-World Scenes

This section briefly describes the techniques we use to render the images shown in Figures 8 and 9. Debevec [7] defines the process in terms of three components: a distant scene, a local scene and a set of synthetic objects. The distant scene sends light to the other two. It represents the real scene's illumination arriving at the future location of the synthetic objects. The local scene is the part of the real scene whose appearance is affected by the addition of the synthetic objects.

The user has to provide some information about the distant scene. She has to give an approximation of its geometry and she has to capture the light it radiates. This light will be

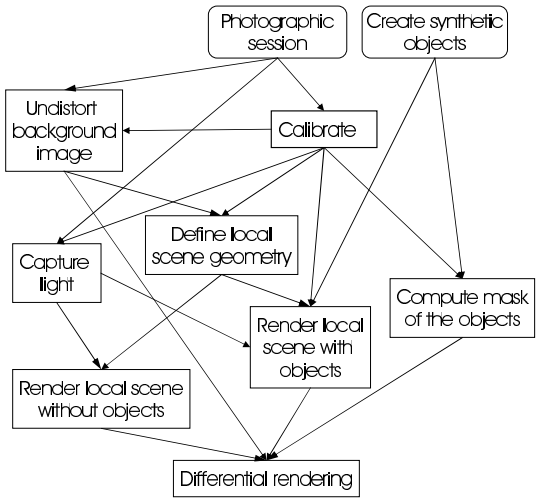

Figure 2: *Information flow in our system.*

represented by means of a high dynamic range image. The method of capturing the light in the scene is studied in the next section. It is not necessary to model the effects of light on the distant scene because the distance scene is not modified. However, we need both the geometry and the material properties of the local scene in order to render it after adding the synthetic objects. The local scene is usually reduced to the plane that supports the synthetic objects.

Figure 2 shows the steps we follow to integrate synthetic objects and real scenes using our system. Every step of the process is briefly explained in the following sections. The incoming arrows in a process define its information inputs and, therefore, impose a temporal order on the execution of the tasks. The user must provide two initial inputs to the application: a set of photographs and the model and materials of the synthetic objects. The photographs must be shot according to the guidelines proposed in the following section.

4.1. Photographic Session and Calibration

This step has a direct impact on the quality of the final image, so it is worth planning the session carefully. The goals of the photographic session are: (i) obtaining the background image that will be used for digital compositing, (ii) acquiring the images needed in order to recover the positions of the camera, and (iii) taking the photographs to build the distant scene. To achieve them we use a camera (digital, if possible), a tripod, a mirrored ball and a calibration template.

We start by taking several photographs of the calibration template from different positions and orientations (see Figure 1). The purpose of this photographs is to calibrate the camera, i.e., to compute the position and intrinsic parameters of the camera. To achieve this goal, the photographs must be as clear as possible, so we use a tripod and/or a fast shutter speed. It is also advised not to change the zoom of the camera in the process. A number of photographs greater than five is a good idea, with the purpose of reducing the estimation errors. The last two photographs have to be taken from the same viewpoint, so we use a tripod. Before shooting the second photo, the calibration template must be removed from the scene. This second photograph will be used as the background image, and it will be composed with the synthetic objects.

Now, we capture the light arriving at the future location of the synthetic objects in the real scene. Following the steps given in [6, 7], we shoot several photographs of a mirrored sphere. We use two directions ninety degrees apart to delete the camera (and the photographer) from the final image. The result of these photographs will be a light probe, which is a high dynamic range image that maps each incoming direction from the scene to a pixel in the image. That is, the light probe stores a value of irradiation for each incoming direction from the real scene into the center of the sphere.

In order to achieve the quality required in this phase, we have found that the following advice is useful for capturing the light probe:

- Both positions of the camera have to be equidistant to the sphere.
- One of those positions must be near the viewing direction of the background photograph.
- The mirrored ball should not visibly affect the scene (e.g., casting shadows, etc).
- A large zoom should be used to reduce the perspective distortion.
- The camera must not move during the session.
- The parameters of the camera, shutter speed, aperture, etc, should be recorded for each shot of the session.

There are a number of problems that can arise after the construction of the high dynamic range image. For instance, the borders of the image may look blurry. This may happen because the camera moved while taking the photographs. If the photographic session can not be repeated, then we need to register the images, in order to make them match. Ideally we would like to use an automatic registration algorithm. However, we find that existing algorithms fail when registering high shutter speed images (too dark) or low shutter speed images (too white). Another problem is that strange shadows may appear in the image. This typically happens when some of the images used to compile the high dynamic range image contain an object that does not appear in the others. This shadow can be deleted using the process explained later for removing the camera from the images. Finally, unexpected colors may appear in the image. If those colors appear near very bright areas (i.e. windows) the problem is likely to be a highest shutter speed too slow for the detail of such areas. The only solution to this problem is to repeat the session.

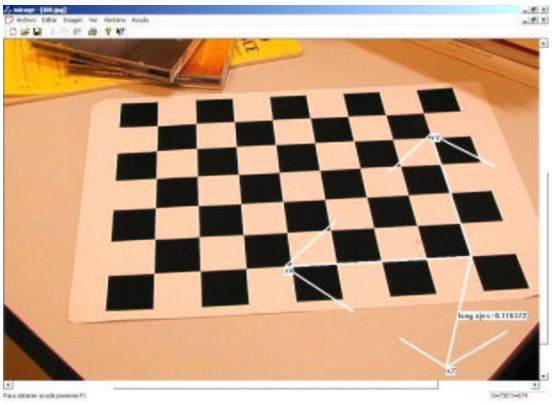

Figure 3: *Result of the calibration step.*

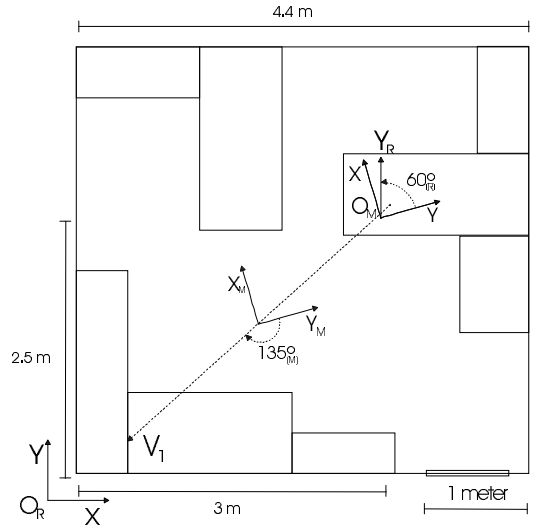

Figure 4: *Parameters for image registration.*

Once the photographs have been shot, and the two high dynamic range images have been built, the user must remove the camera and any other artifacts from the images. To do so she must follow these steps: (i) rotate the image furthest from the viewing direction of the background image until both images match; (ii) replace the camera in the image that has not been rotated by the same part of the rotated image (the camera is not there); and (iii) apply a panoramic transformation in order to create a light probe from a mirrored ball image. All of these steps can be carried out using HDRShop, a public domain application that can be obtained from [8].

An estimation of the geometry of the real scene is used later. This geometry can be as simple as a box of the size of the scene, so it is necessary to note down that information during the photographic session. Then, our application creates a 3D global reference system from the photographs taken during the calibration phase. The user only has to provide the number and size of the squares in the calibration template. Figure 3 shows the coordinate system imposed on a photograph. The position of the synthetic objects and the local scene must be later defined in that coordinate system. Once the calibration has been done, the background photograph has to be corrected in order to remove the effects of the lens distortion.

4.2. Distant Scene

The goal of this step is to create a light-based model of the surroundings of the scene. This light model is used later by RADIANCE to illuminate the local scene and the synthetic objects. In order to define a mapping between the light probe and the scene's geometry, the user has to provide a set of parameters and a rough estimation of the geometry of the scene.

Figure 4 shows the parameters that define the distant scene geometry, and the registration of the light probe to the geometry. O_M is the origin defined in the calibration process (see Figure 3) and O_R is the origin of the world of RADIANCE, where the geometry is defined. The user has to provide

- the volume of the environment (height, width and depth),
- the position of the calibration origin, O_M, with respect to O_R,
- the angle between the Y-axis defined in the calibration process (origin at O_M) and the Y-axis defined at O_R, and
- the angle between the Y-axis, defined by the calibration process, and V_1. V_1 is the direction defined by joining the center of the mirrored ball with the position of the camera.

With all this information RADIANCE can properly shade virtual objects located near the center of the light probe. Note that the parameters used in this process do not need to be too accurate.

4.3. Local Scene

The user has to define the local scene, the part of the real scene that the synthetic objects modify. Our application lets the user define a flat polygon on the $Z = 0$ plane by *drawing* its vertices on the background photograph (see Figure 5). Given the intrinsic and extrinsic matrices that define the projection of the real world onto the background image, we can reconstruct the 3D point that intersects the $Z = 0$ plane with any pixel.

Figure 6 shows the process, where

- p is the pixel selected by the user,
- P is the 3D point to be reconstructed on the $Z = 0$ plane,
- ?CS represent the World, Image and Camera Coordinate Systems,

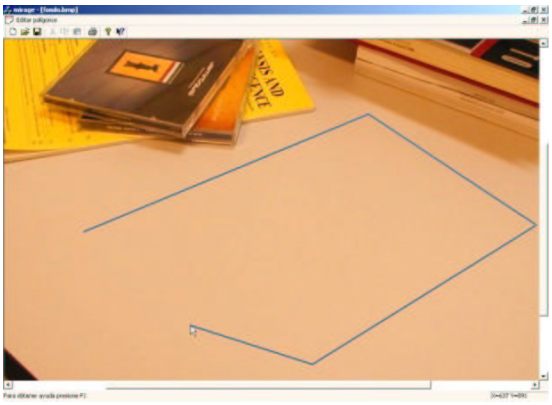

Figure 5: *Drawing a 3D polygon.*

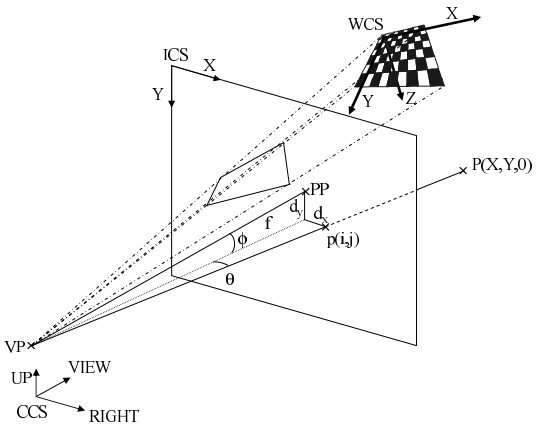

Figure 6: *Reconstruction of a 3D point from a pixel.*

- VP is the viewing position,
- PP is the principal point on the image,
- f is the focal length,
- d_x, d_y are the horizontal and vertical distances from PP to p, and
- θ, ϕ are the angular distances from PP to p.

The extrinsic matrix computed during the calibration process for the background image defines the position and orientation of the camera with respect to the WCS. Those parameters convert the WCS into the CCS. That way we can compute the position of the objects in the scene with respect to the camera.

The user has to define the material properties of the elements of the local scene. In our current implementation, we approximate the color of the local scene using the background image. Later, this color value is used as an initial value to compose the final image. We have found this value to be an approximation good enough for the final image. The user has to decide the type of material of the local scene from those materials defined by RADIANCE (plastic, metal, etc) [28] and its parameters (specularity, roughness). Most surfaces can be described by means of RADIANCE's *plastic* material.

4.4. Synthetic Objects

The user can design the synthetic objects using any 3D design package. The only requirement is that the output format be recognized by RADIANCE (i.e. 3DStudio .3ds, .dxf, etc). A mask of the objects will be needed later.

4.5. Final Rendering

The product of this step is the final image. To compose it we use the background image, a mask and a rendering of the synthetic objects, and a rendering of the synthetic objects in the local scene. The two renderings are produced using RADIANCE with the geometry information defined in the preceding steps, lit by the light probe and the distant scene defined in Sections 4.1 and 4.2.

Due to the way RADIANCE renders images, it is possible to obtain a final image with a uniform surface showing an unexpected textured pattern. This happens when the distant scene contains high intensity light sources. RADIANCE computes the light arriving at a point by casting a few random ambient rays. Since the random rays are typically different from one pixel to the next, it is possible that two neighboring pixels be lit differently, depending on their sets of random ambient rays. Since the lighting defined for the scene includes no light sources, but only ambient light, we have to force RADIANCE to take into account the light sources that appear in the light probe.

RADIANCE lets us define secondary light sources in a scene. A secondary light source is an impostor made of a set of polygons that simulate the light distribution produced by an object. Such impostors are always used in the lighting and shading computations of RADIANCE. Therefore, we resolve the above problem by defining a secondary light source for every bright area in the distance scene.

If the part of the background image occupied by the synthetic object and the local scene were substituted directly with their renderings, the resulting image would not be credible. The limit between the real part of the scene and the rendered objects would be visible. Instead of using traditional digital compositing, we add to the background image the *changes* produced by the synthetic objects. To do so we define a differential rendering algorithm as follows:

```
INPUT:
mask: each pixel is 1 if there is a syn-
thetic object, 0 otherwise
local: local scene lit by distant scene
objects: local scene + objects lit by dis-
tant scene
```

Figure 7: *Components of the differential rendering.*

```
OUTPUT:
final: image

for each pixel (x,y) do
  if mask(x,y) then
     final(x,y)=objects(x,y)
  else
     final(x,y)=background(x,y)+
                objects(x,y)-local(x,y)
  end if
end for
```

Figure 7 shows the components used by the differential renderer to generate the image in Figure 8. From left to right and top to bottom, the images are:

1. the local scene,
2. the local scene and synthetic objects,
3. the mask of the synthetic objects,
4. the differences between 1 and 2, and
5. the background image.

We have scaled the fifth image in order to show more detail. Note that our differential renderer adds to the background image the synthetic objects and the shadows and highlights they produce in the real scene. In Figure 9 we show a different image obtained with our method.

5. Improvements on our System

We have presented a method that adds synthetic objects to real-world photographs, and we have shown that we can produce realistic images. Now, we want to obtain better quality images, as well as animation sequences for video and film production. In this section, we propose some improvements on our system aimed at providing better image quality and simplifying the production of animated image sequences.

Figure 8: *Synthetic objects in a real scene (objects obtained from [8]).*

5.1. Inter-reflections Between Real and Synthetic Objects

One of the most difficult problems we find in building hybrid synthetic and real-world models is how to represent the geometry of the real-world scene. Current systems require that the user be the one modeling the local scene. In those systems the local scene is assumed to be made of simple shapes, like planes, spheres and boxes. These shapes are useful when defining real objects that are geometrically simple. There are alternative photogrammetric techniques that can be used to build a semi-automatic application to help the user define the geometric shape of these objects. Simple objects, such as books, CD cases, balls, pens, etc, can be easily modeled with tools like Debevec's Façade [5].

In order to recover the material properties of the object, a texture can also be extracted from the input photographs. Note that we know the precise position of the objects with respect to the camera. So we can use techniques like 3D warping to extract textures from the photograph and use them to model the local scene. This approach is only feasible if the scene objects are simple. If not, recovering the shape of a real object can be a very complex task, especially since Computer Vision techniques for shape recovery are not very robust, yet. For example, we may need to use specialized hardware, like a 3D scanner. Instead, we propose the use of impostors to include complex objects in the local scene. An important problem of this approach is how to build an impostor with a shape close enough to the shape of the real-world object. To solve this problem we may use some automatic segmentation techniques.

Figure 9: *Synthetic object in a real scene.*

Automatic segmentation in non restricted environments is a problem still unresolved, so some kind of user assistance is required. In order to build the impostor, we propose a technique like the one proposed in [19]. The technique computes the visual hull of an object from the intersection of the silhouette cones defined by the limits of the object in each frame and the projection center of the frame. As the authors state, the visual hull has several desirable properties: it contains the actual object and it has consistent silhouettes. The work presented in [19] uses the visual hull to render the modeled object from other viewpoints. In [17] a technique like visual hulls is used to solve the occlusion problem in Augmented Reality.

We propose using impostors to effectively model the real-world objects. Then, we can modify their appearance after inserting the synthetic objects. The impostors should also allow us to model the influence of the real-world objects on the synthetic ones.

5.2. Animating the Camera Viewpoint

Animating scenes with real and synthetic objects is a problem that has not received much attention in the literature. It is not complex to render animations where the synthetic objects move in the scene, as long as the real scene stays static. The technique shown in this paper produces accurate images, which can show, for example, a synthetic ball moving on a table. The images thus generated contain visual cues, like shadows and inter-reflections [27], needed to achieve a photo-realistic integration of the synthetic and the real-world objects. Nevertheless, an animation made of raw frames produced by our method would be “too perfect”. One of the problems of computer graphics animation is that it produces very accurate, clean and sharp images that look like a not-so-realistic metallic animation. To reduce this problem, a motion blur filter must be applied to the frames in the animation. In a real camera, motion occurring while the camera shutter is open produces a blurred image. Seminal work related to this effect can be found in [15, 23]. Cook [4] developed a technique that simulates the motion blur by distributing rays in the rendering phase for each pixel. If motion blur is needed after the rendering phase, we can use a technique like the one presented in [3].

A more complex problem arises when the real camera is moving. The problem is that the background image changes and, therefore, the calibration step must be repeated for every frame. As we show in Section 4.1, an accurate estimation of the intrinsic parameters of the camera, its location, and its orientation can be computed by using a calibration template. We take two photographs from the same location, one with the calibration template, and the other without it. We use the first one to compute the parameters of the camera, and the second one to obtain the background image. Unfortunately, this process is not possible when the camera is moving.

Note that maintaining temporal correspondence [24] is critical to achieve realistic animations. The synthetic objects must follow the movement of the camera. This problem has been studied in Augmented Reality systems [1, 2]. The problem of registration in AR consists of tracking the user’s view direction in order to refresh the perspective of the synthetic objects. Moreover, this view refreshing must be done in real time. This problem, in its unrestricted form, is still unresolved. In fact, there are hybrid systems that combine several techniques of tracking in order to compensate for the weaknesses of each separate technique. For example, magnetic sensors and video, or accelerometers and video tracking are being used in different projects. The best results are obtained in restricted environments, where fiducials have been placed in order to help detection. This technique, applied to our application, would require modifying the real scene by placing the fiducials and then, in a postproduction step, removing those fiducials digitally. This technique was used, for example, in [26]. In uncontrolled exterior scenes, other tools, such as GPS, a compass or a gyroscope can be used. With those techniques we must set important restrictions on the actions of the user in order to guarantee some robustness of the system.

This problem also appears when designing special effects for the film industry. The first motion picture that used an early form of motion control was “2001: A Space Odyssey” (1968). Motion control uses a computer to control the articulated arm where the camera is mounted. This allows us to film several elements of a complex scene exactly from the same perspective. As each element is aligned properly, the digital compositing is more credible. If the computer-operated system is unable to provide the camera parameters, a process like the one we used in Section 4.1 can be used. That is, a first shot of the real scene with the calibration template is taken by the moving camera. Afterwards, once the template is removed from the scene, a second shot of the scene is taken following the previous path. During this sec-

ond pass all the frames are taken at exactly the same positions as before. This process allows us to recover the parameters of the camera for each frame.

We have outlined the solutions used in Augmented Reality and film production to solve the temporal correspondence problem. The first solution has the most inexpensive implementation in terms of hardware requirements. Following our philosophy of avoiding specialized hardware, we prefer calibrating the camera using fiducials placed in the real-world scene. Once the camera has been calibrated for all the frames of the animation sequence, we can start thinking about the rendering phase.

In the rendering phase, a great deal of computational power is needed to generate the animation frame by frame. To alleviate this problem, we reduce the number of images used in one or all of the phases. For example, we can acquire a reduced number of real-world photographs. Then, if necessary, we can interpolate between them using a technique called view morphing [25]. View morphing has been shown to produce good results when rendering architectural environments [5, 21]. Additionally, we can render a reduced set of images of the synthetic objects. Those images can also be interpolated before compositing with the background images. RADIANCE [28] supports image interpolation between keyframes. It also takes advantage of the depth information stored in the Z-buffer to obtain better interpolation results. Once both sets of images have been obtained and, possibly, interpolated, we compose the two sets and produce the final set of rendered images. A final interpolation step may then be necessary if the set of composed images is smaller than the set of final images.

In the general case, image sets of different sizes can be captured, rendered and composited to obtain the final animation sequence. It remains to be determined what combination of which sets of images produces the best results. In any case, we expect interpolation to be less expensive than rendering. Therefore, we expect to substantially reduce production time if reduced sets of images can be used in the intermediate phases.

6. Conclusions

We have presented an application that produces photorealistic images by compositing rendered virtual objects with photographs of a real scene. This compositing simulates the effects of the real scene on the synthetic objects by rendering them under the same lighting conditions as the real scene. Furthermore, we also simulate the presence of the synthetic objects by computing their interactions, that is, shadows and inter-reflections, with the real scene. We present the methodology we follow to obtain the composited images. Our methodology is based on principles of camera calibration, high dynamic-range imaging and image-based lighting.

We also introduce some new ideas on how to enhance the quality of our images by properly modeling the interactions between the synthetic objects and the parts of the real scene closest to them. This is a difficult task: if a complex real object needs to be modeled to be included in the rendering phase, a great deal of manual work may be necessary. In this paper we outline some possible solutions to this problem. We also address the issue of generating animated sequences instead of single images. We discuss a couple of solutions to the problem of temporal correspondence, and suggest the use of interpolation to reduce the time needed to produce animation sequences.

Acknowledgements

This work was partially funded by project TIC 1999-0510-C02-01 of the Spanish Ministry of Science and Technology. Additional support was provided by the “Programa de Incentivo a la Investigación” of the Polythecnic University of Valencia..

References

1. R.Azuma, “A Survey of Augmented Reality”, *Presence:Teleoperators and Virtual Environments*, **6**(4):355-385, 1987. 1, 2, 8

2. R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, B. MacIntyre, “Recent Advances in Augmented Reality”, *IEEE Computer Graphics and Applications*, **21**(6):34-47, 2001. 1, 2, 8

3. G. J. Brostow, I. Essa, “Image-Based Motion Blur for Stop Motion Animation”, *Computer Graphics (Proceedings of SIGGRAPH 2001)*, pp. 561-566, 2001. 8

4. R. L. Cook, T. Porter, L. Carpenter, “Distributed Ray Tracing”. *Computer Graphics (Proceedings of SIGGRAPH 84)*, **18**(3):137-145, 1984. 8

5. P. E. Debevec, C. J. Taylor, J. Malik, “Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach”, *Computer Graphics (Proceedings of SIGGRAPH 96)*, **30**:11-20, 1996. 7, 9

6. P. E. Debevec, J. Malik, “Recovering High Dynamic Range Radiance Maps from Photographs”, *Computer Graphics (Proceedings of SIGGRAPH 97)*, pp. 369-378, 1997. 3, 4

7. P. E. Debevec, “Rendering Synthetic Objects Into Real Scenes: Bridging Traditional and Image-Based Graphics With Global Illumination and High Dynamic Range Photography”, *Computer Graphics (Proceedings of SIGGRAPH 98)*, pp. 189-198, 1998. 2, 3, 4

8. P. E. Debevec, Home Page at `http://www.debevec.org` 5, 7

9. G. Drettakis, L. Robert, S. Bougnoux, "Interactive common illumination for Computer Augmented Reality". *Eurographics Rendering Workshop* pp. 45-56, 1997 2

10. J. Foley, A. van Dam, S. Feiner, J. Hugues, and R. Phillips, *Introduction to Computer Graphics*. Addison Wesley, 1993. 2

11. A. Fournier, A. Gunawan, C. Romanzin, "Common illumination between real and computer generated scenes", *Proc. Graphics Interface'93*, 1993. 2

12. S. Gibson, A. Murta, "Interactive Rendering with real world illumination", *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*, pp. 365-376, 2000. 2

13. P. Jancène, F. Neyret, X. Provot, J.-P. Tarel, J.-M. Vézien, C. Meilhac, A. Verroust, "RES : computing the interactions between real and virtual objects in video sequences", *IEEE Workshop on Networked Realities*. 1995. 2

14. D. Kelly. *Digital compositing in depth*. The Coriolis Group. 2000. 2

15. J. Korein, N. Badler, "Temporal Anti-Aliasing in Computer Generated Animation", *Computer Graphics (Proceedings of SIGGRAPH 83)*, **17**(3):377-388, 1983. 8

16. K. Kutulakos, J. Vallino. "Calibration-Free Augmented Reality", *IEEE Trans. on Visualization and Computer Graphics*, **4**(1), 1998. 2

17. V. Lepetit and M.-O. Berger, "Handling Occlusions in Augmented Reality Systems : A Semi-Automatic Method", *IEEE and ACM International Symposyum on Augmented Reality*, Munich, Germany, pp. 137-146, 2000. 8

18. C. Loscos, M.-C. Frasson, G. Drettakis, B. Walter, X. Granier, P. Poulin. "Interactive Virtual Relighting and Remodeling of Real Scenes". *Eurographics Rendering Workshop*, pp. 329-340, 1999. 2

19. W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, L. McMillan, "Image-Based Visual Hulls". *Computer Graphics (Proceedings of SIGGRAPH 2000)*, pp. 369-374, 2000. 8

20. E. Nakamae, K. Harada, T. Ishizaki, T. Nishita, "A montage method: the overlaying of the computer generated images onto a background photograph", *Proc. 13th annual conference on Computer graphics and interactive techniques*, pp. 207-214, 1986. 2

21. M. M. Oliveira, G. Bishop, D. McAllister, "Relief Texture Mapping". *Computer Graphics (Proceedings of SIGGRAPH 2000)*, pp. 359-368, 2000. 9

22. Open Source Computer Vision Library. `http://intel.com/research/mrl/research/opencv` 3

23. M. Potmesil, I. Chakravarty, "Modeling motion blur in computer-generated images", *Computer Graphics (Proceedings of SIGGRAPH 83)*, **17**(3):389-399, 1983. 8

24. I. Sato, Y. Sato, K. Ikeuchi. "Acquiring a Radiance Distribution to Superimpose Virtual Objects onto a Real Scene", *IEEE Trans. On Visualization and Computer Graphics*, **5**(1), March 1999. 1, 2, 8

25. S. M. Seitz, C. R. Dyer, "View Morphing". *Computer Graphics (Proceedings of SIGGRAPH 96)*, **30**:21-30, 1996. 9

26. A. State, G. Hirota, D. T. Chen, W. F. Garrett, M. A. Livingston, "Superior augmented reality registration by integrating landmark tracking and magnetic tracking", *Computer Graphics (Proceedings of SIGGRAPH 96)*, **30**:429-438. 1996 8

27. W. B. Thompson, P. Shirley, and B. Smits, "Visual Glue", *University of Utah Technical Report UUCS-98-007* (March 1998). 8

28. G. W. Larson, R. Shakespeare. *Rendering with Radiance: The Art and Science of Lighting Visualization*. Morgan Kaufmann Publishers. 1998. 3, 6, 9

29. T. Wittenburg. *Photo-based 3D graphics in C++: compositing, warping, morphing and other digital effects*. John Wiley and Sons. 1995. 2

30. Y. Yu, P. Debevec, J. Malik, T. Hawkins. "Inverse Global Illumination: Recovering Reflectance Models of Real Scenes from Photographs" *Computer Graphics (Proceedings of SIGGRAPH 99)*, pp. 215-224, 1999. 2

31. Z. Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(11):1330-1334, 2000. 3