Virtual Reflections for Augmented Reality Environments

Timo Ropinski, Steffen Wachenfeld and Klaus Hinrichs
Institut für Informatik, Westfälische Wilhelms-Universität Münster, Germany
{ropinski, steffen.wachenfeld, khh}@math.uni-muenster.de

Abstract

For a photorealistic appearance of virtual objects rendered in augmented reality environments global illumination must be simulated. In this paper we present a real-time technique for generating reflections on virtual objects which are rendered into images or image sequences of real scenes. The basic idea of virtual reflections is to estimate the environment of a virtual object by extracting information from the image to be augmented. Based on this estimated environment, cube map textures are generated and applied by using environment mapping during rendering to simulate reflections on the virtual object. Although some work has been done to simulate global lighting conditions in real-time augmented reality systems, current approaches do not consider reflections of real world objects onto virtual objects. Virtual reflections are an easy to implement approximation of such reflections and can be combined with existing illumination techniques, such as the creation of shadows for augmented reality environments.

Key words: Augmented Reality, Environment Mapping, Virtual Reflections

1. Introduction

The goal to generate photorealistic images when integrating virtual objects into real scenes can only be achieved if light interactions between virtual and real objects are modeled. Therefore it is necessary to simulate global illumination in augmented reality environments. In this paper we present a real-time technique called *virtual reflections*, which is capable of simulating reflections of real world objects on virtual objects which are rendered into images or image sequences of real scenes. The proposed technique needs no additional information to calculate convincing reflections when a real scene image is augmented with one or multiple virtual objects.

The basic idea of virtual reflections is to estimate the environment of a virtual object by extracting information from the image to be augmented. Since in most augmented reality environments (e.g., the magic book [1]) the virtual objects have to be rendered in real-time this computation has to be performed on the fly. When using virtual reflections cube map textures are generated based on the estimated environment. These cube map textures are applied during rendering using environment mapping to simulate reflections on the virtual objects.

Although some work has been done to simulate global lighting conditions in real-time augmented reality systems, current approaches ([2], [3]) do not consider reflections of real world objects onto virtual objects. The technique we are going to propose can be easily integrated into existing rendering systems supporting other illumination techniques, such as the creation of shadows in augmented reality environments.

If more than one virtual object is being rendered into a virtual scene, our technique also allows the simulation of reflections of virtual objects on other virtual objects.

The next section discusses related work concerning environment mapping as well as the simulation of global lighting conditions for augmented reality environments. Section 3 introduces the virtual reflections technique and explains how to improve image quality and how to simulate reflections when a real scene is augmented with multiple objects. In section 4 some examples are shown. Section 5 discusses the frame rates achieved, when virtual reflections are applied under different conditions. Section 6 concludes the paper.

2. Related Work

This section discusses related work regarding both, the environment mapping technique as well as the simulation of global illumination in augmented reality environments.

To apply virtual reflections the construction of an environment map is necessary. Environment mapping has been introduced in 1976 by Blinn and Newell [4]. In their paper they introduce environment maps, which can be used to simulate reflections on arbitrarily shaped virtual objects. An environment map is a texture applied during rendering, which contains information about the environment, i.e., the surrounding of the reflecting object. During the environment mapping process appropriate texture coordinates associated with virtual objects are used to determine, which parts of the environment are reflected on a virtual object. Greene [5] extended the technique and gave examples of two types of environment textures, spherical as well as cubic ones.

He discussed how to construct both types either synthetically or by taking a picture with a fisheye lens. Spherical environment maps provide a projection of the environment onto a sphere, while the environment is mapped onto the six sides of a cube when using cube map textures. In his paper Greene gave examples for the use of environment maps to render an infinitely far away backdrop, and to simulate realistic reflections on shiny objects closer to the virtual camera. Greene preferred the use of cubic environment maps over that of spherical ones, because of the easier integration into 3D graphics hardware. Today the environment mapping technique is accelerated by graphics hardware and is extensively used in computer games and other interactive graphics applications to simulate reflections while preserving high frame rates.

Regarding augmented reality environments, some work has been done to investigate how virtual objects can be smoothly integrated into real scenes. Most of this work supporting the simulation of global lighting conditions refers to the simulation of shadowing, both shadows of virtual objects on real world objects and shadows of real world objects on virtual objects. Naemura et al. [6] describe the application of *virtual lights* to be placed within the environment to simulate shadows in augmented reality applications. But the technique requires information about the position of the real world objects for shadow simulation. A different approach has been proposed by Loscos et al. [7]. In their work, they reconstruct the lighting information of a scene as well as its geometry from a set of photographs. Since these photographs have to be taken under certain circumstances, the technique cannot be used for real-time augmentation. To allow shadows in real-time augmented reality environments, Haller et al. [3] have adapted the shadow volume technique used in real-time computer graphics. Using this technique, they are able to let virtual objects cast and receive shadows. To allow shadows as well as reflections in augmented reality environments, Debevec [8] has introduced a system classifying virtual objects with respect to their distance to the virtual camera. By only considering the objects closer to the camera a global illumination model can be used to simulate both shadows and reflections. Although only objects positioned near the virtual camera are considered, Debevec's approach does not allow real-time augmentation. To simulate reflections in virtual reality environments, Kautz et al. [9] use cubic environment maps. In contrast to the approach presented in this paper their technique focuses on filtering techniques rather than on the construction process of the environment map. The technique is not extendable to real-time augmented reality environments, because additional information is needed for generating the environment map. Agusanto et al. [2] also propose the usage of environment maps in augmented reality environments. In their approach, they improve realism for rendering virtual objects in augmented reality by using pre-filtered high-dynamic range environment maps of the real scene. Using these maps they can obtain diffuse as well as glossy light reflections on virtual objects by using a multi-pass rendering algorithm. In contrast to the technique presented in this paper, no specular reflections of real world objects are possible, since the information contributing to the cube map is blurred and can be used to simulate lighting conditions only.

3. Technique

In this section we describe our technique to simulate reflections on virtual objects which are rendered into a real world scene. Figure 1 shows the augmentation of such a real scene image, figure 1 (a) shows a rendered 3D model with textures and figure 1 (b) shows the same 3D model rendered using virtual reflections.

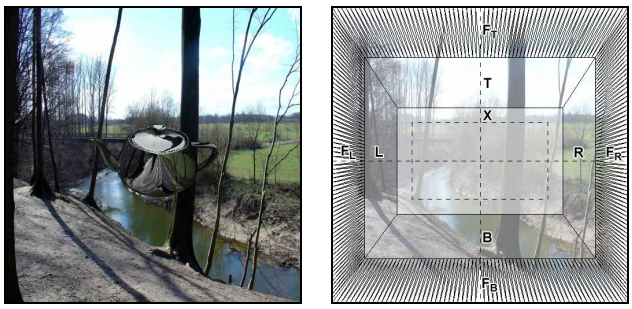
(a) no reflections (b) virtual reflections

Fig. 1 3D model rendered into a real scene
(model by Laurence Boissieux © INRIA 2003).

3.1 Automatic Environment Map Generation

Virtual reflections are achieved by applying cube map environment textures when rendering virtual objects. These cube map textures are obtained by projecting six regions *L*, *R*, *T*, *B*, *X* and *F* of the real scene image onto the six sides of a cube (see figure 2).

To ensure that real world objects located on one side of the virtual object are only reflected on that particular side, the image coordinates of the virtual object have to be taken into account during generation of the cube map textures. Therefore the six regions are segmented further, as is indicated by the two dashed lines in figure 2 (b). Using this segmentation the left and the right cube map texture are obtained by projecting the regions labeled with *L* and *R* onto the corresponding face of the cube, while the image information above/below the dashed line is projected onto the upper/lower half of these faces. To generate the cube map textures for the top and bottom face of the cube the image information of the regions *T* and *B* is used. The partition given by the dashed vertical line is used to project only parts of the image located left/right of the virtual object onto the particular half of the corresponding cube map texture.



(a) virtual reflections (b) segmentation of the scene image

(c) the generated cube map

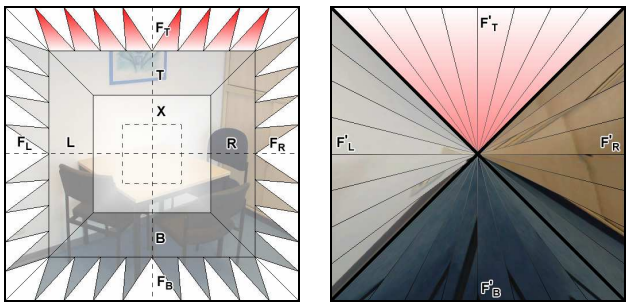
Fig. 2 Segmentation and projection of image data for the generation of the cube map.

To generate the cube map texture for the back face of the cube, the inner region *X* of the scene image is used. Using this segmentation we are able to construct five of the six needed cube map textures. Only for the front cube map texture the information is not contained in the real scene image. This texture is very important, because it is reflected by all object surfaces pointing towards the virtual camera. We approximate the information needed for the front cube map texture by projecting the outermost parts of the image onto the front face.

Figure 3 shows how this approximation is done by projecting the image information of the four outermost trapezoidal regions (F_T, F_L, F_R, F_B) onto four triangular areas which form the cube's front face (F'_T, F'_L, F'_R, F'_B). To do this we cut out a number of *n* triangles in the trapezoidal regions of the real scene image and use them with inverted orientation to assemble the cube map's front face. The red gradient in figure 3 visualizes how the orientation of those image regions has been changed. The dashed lines indicate which parts of the trapezoidal regions appear on which quarter of the front texture.

This way a cube map which has smooth edges between all adjacent cube faces can be constructed in real-time. Although there is a singularity in the middle of the front cube face we get visually convincing reflections especially on more complex objects or objects with curved surfaces (see section 4). Instead of cubic environment maps also spherical maps could be used. But the projection of the scene onto a spherical map would result in an unsmooth edge reaching from the map's top to the bottom at the middle of the map's front. Furthermore, when generating spherical maps it is much more complicated and time consuming to achieve reflections of real scene objects only onto the correct sides of virtual objects. The reason is that it would require the real-time adaptation of the spherical projection which is much more computational expensive than the adaptation of the cubic projection.

The described technique is capable of simulating glossy as well as diffuse reflections of the real scene on virtual objects. To achieve different degrees of reflection, various mechanisms can be used to combine the colors stored in the cube map textures with the virtual object's material. We have used different combination modes provided by the OpenGL API, on which our implementation is based. In figure 1 (b) the application of the replace mode is shown, i.e., the color of the virtual object is replaced by the corresponding color stored in the cube map. This combination leads to the visually convincing representation of an object made of chrome. Figure 7 (a) shows an image applying virtual reflections using the OpenGL modulate mode. In this mode the material properties of the virtual object are multiplied by the color stored in the cube map to get the final color for the virtual object.

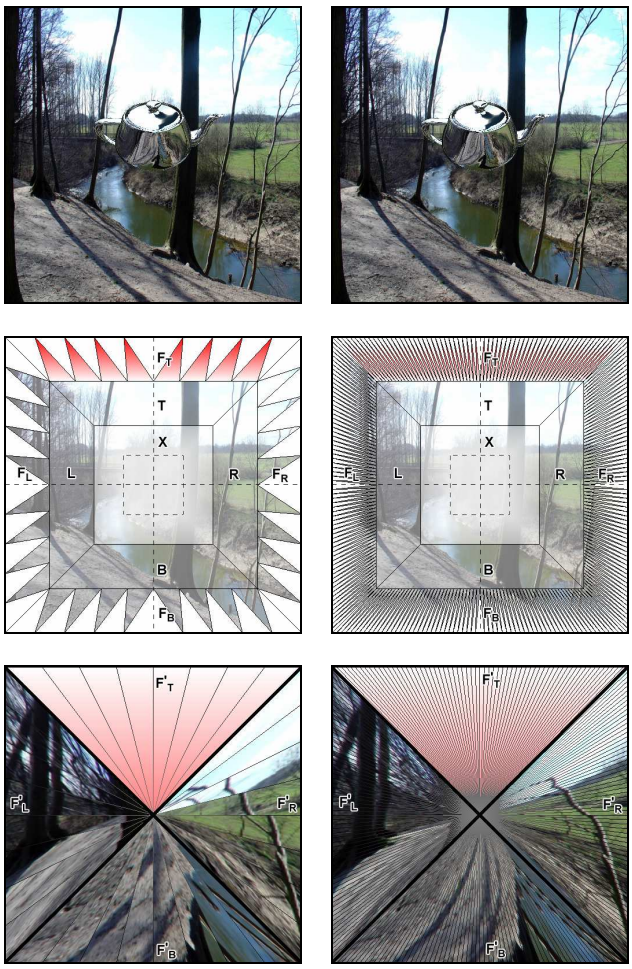


(a) segmentation of the scene image (b) cube map's front face

Fig. 3 Approximation of the cube's front face using *n*=32 triangles.

3.2 Improving Reflections

The constructed cube map textures always have smooth edges between all adjacent cube faces but the edges between the triangles used to compose the cube's front map show discontinuities. The quality of the cube's front texture and with it the reflection quality can be improved by adjusting the number *n* of triangles which are cut out of the outer image region and projected onto the front



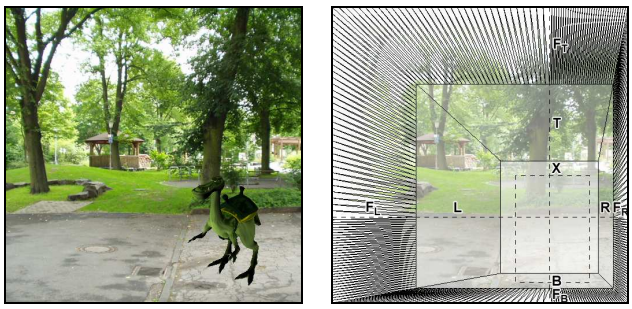
(a) level of approximation n=32 triangles

(b) level of approximation n=256 triangles

Fig. 4 Comparison of scene augmentation for different levels of approximation.

face. In figure 4 different levels of approximation for the cube map generation process are shown. The left column of images was created using n=32 triangles, while the right column was created using a higher approximation of n=256 triangles. From top to bottom the images show: the augmented scene, the segmentation of the real scene image and the resulting front face.

To achieve the most convincing reflections it has to be considered how to divide the image into the six areas *L*, *R*, *T*, *B*, *X* and *F*. The area *X* which will be projected onto the cubes back face should contain image parts that lie behind the virtual object. If no extra information about the scene is given, we process the bounding box of the virtual object in image coordinates (dotted line within *X*, see figure 3 (a)) and enlarge it to define the region *X*. This way *X* grows and moves when the virtual object is resized or moved. Experiments have shown good results using an enlargement by 5-10% of the image width/height while keeping a minimum distance to the



(a) augmented image

(b) corresponding segmentation

Fig. 5 Augmentation of a real scene image with an arbitrarily placed virtual object.

image border of 10-20% of the image width/height. In special cases, where the virtual object is positioned near the image border, keeping a specified minimum distance to the border can result in an area *X* which is smaller than the bounding box.

The other regions appear optimal in most cases if *F* is chosen in a way that it bisects the remaining distance between *X* and the image border. In figure 5 an augmentation of a real scene image with an arbitrarily positioned virtual object is shown. Figure 5 (a) shows the augmented scene, while figure 5 (b) shows the used segmentation of the real scene image. The image is segmented in a way that the cube map textures are generated, such that real world objects occurring in the image will only be reflected on the corresponding sides of a virtual object. It can be seen that *X* is chosen in a way that the minimum distance to the bottom border of the image is kept. Results can be further improved if additional information about the scene is available such as the location of walls in indoor scenes or the borders of streets in outdoor scenes. The extraction of such information from the scene image could be integrated using real-time capable computer vision algorithms, e.g., vanishing point detection.

(a) recursion depth r=2

(b) recursion depth r=6

Fig. 6 Multiple objects rendered into a real scene using different recursion depths.

(a) using virtual reflections

(b) no reflections

Fig. 7 Picture of Manhattan downtown augmented with a virtual car.

3.3 Multiple Objects

Virtual reflections can also be applied to a scene augmented with more than one virtual object when it is desired that the virtual objects reflect themselves on each other. These kinds of reflections can be achieved by using our technique performing multiple rendering passes. The number of required rendering passes depends on the number of desired recursive reflections *r*. Figure 6 shows an example of rendering two shiny objects with virtual reflections. It as well illustrates that higher numbers of recursive reflections will only have a small impact on the resulting image.

```
loadRealSceneImage(S);

for (int r=0;r<recursionDepth;r++) {
   for (int i=0;i<numObjects;i++) {
      calcBoundingBox(obj_i);
      createEnvMap(S, obj_i);
      renderWithVirtRefl(obj_i);
   }
   if (r != recursionDepth)
      S = captureScreen();
}
```

Listing 1 Pseudocode showing the application of virtual reflections to multiple objects.

To integrate multiple virtual objects into a real scene image, the technique has to be extended to a multi-pass rendering technique. This technique performs the steps shown in listing 1. Executing these steps once (r=1) will result in an image with virtual objects showing reflections of the scene but no reflections of the other present virtual objects. Executing these steps twice (r=2) results in virtual objects being reflected on the surfaces of other virtual objects (see figure 6 (a)). Two opposing mirrors rendered in this manner will each reflect the other mirror but will not reflect themselves in the mirror shown in their own reflection. To achieve a "hall of mirrors" effect more rendering passes are needed and the recursion depth *r* has to be set to the number of desired inter-reflections.

Since it is possible to construct and use a different cube map for each virtual object in the same rendering pass, the number of required rendering passes is independent from the number of objects to be rendered. Therefore the rendering complexity regarding the number of rendering passes is *r*, *r* being the desired recursion depth.

4. Examples

This section describes some examples showing the use of virtual reflections. Figure 7 (a) shows a picture taken in downtown Manhattan, which has been augmented with a car model in real-time. The same scene is shown in figure 7 (b) without using virtual reflections while rendering the car model. During rendering of the scene shown in figure 7 (a) the original material of the car, as shown in figure 7 (b), is modulated by the automatically generated cube map shown in figure 8. The cube map has been constructed by using n=256 triangles specifying the level of approximation used for the segmentation of the real scene image. The augmented image is an example where the region *X* is smaller than the bounding box in order to keep a minimum distance to the image border. Thus to include the information below the region *X* projected to the back of the cube map, the region *X* corresponding to the car model has been set such that the information for the bottom map and the lower part of the

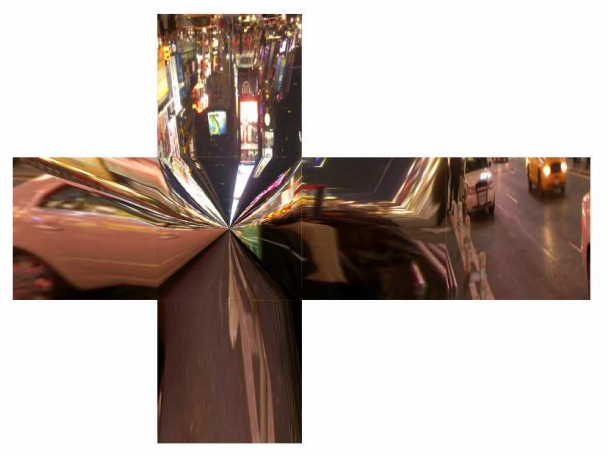


Fig. 8 The generated cube map.

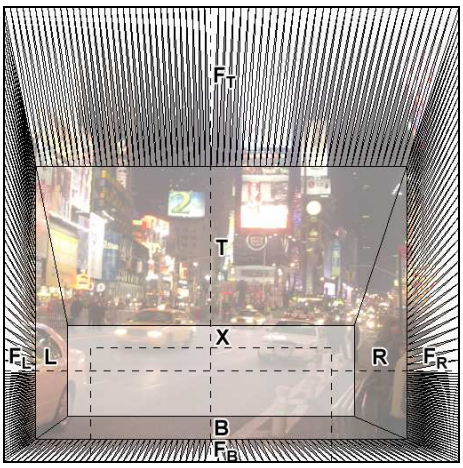


Fig. 9 Segmentation of the real scene image.

front map can be obtained from within the picture. This segmentation used for generating the cube map textures is shown in figure 9.

Virtual reflections are capable of showing approximations of the environmental reflections on virtual objects. Since the human visual perception system is not geared to distinguish physically correct reflections from approximated reflections, the user gets a convincing impression of reflections in most cases.

In Figure 10 virtual reflections generated from the same environment are applied to different virtual objects. Figure 10 (a) shows virtual reflections applied to a cube, while figure 10 (b) shows reflections on a sphere and figure 10 (c) illustrates the application of virtual reflections to a teapot model. The three images illustrate that the degree of realism depends on the curvature of the object's surface virtual reflections are applied to. When applying the created environment map to a planar surface as one of the cube faces shown in figure 10 (a) its structure is revealed. In particular when combined with a small *n* denoting a low approximation level, the different parts of the cube map may be identifiable. Further the singularity at the center of the cube map's front face is shown when projecting on a planar surface. Similar to the cube the sphere gives a good impression of the used cube map textures, but since the sphere's surface is not planar the viewer may not as easily identify the reflections as fake reflections. The most convincing reflection is created on the teapot shown in figure 10 (c), which is the model with the highest degree of curvature.

Figure 11 shows two more examples of the application of virtual reflections to virtual objects having a high degree of surface curvature. In Figure 11 (a) an object is rendered into an indoor scene, while in figure 11 (b) an object is rendered into an outdoor scene.

5. Performance

Because the cube map textures used by virtual reflections are constructed by exploiting hardware accelerated texture mapping functionality, the technique can be applied in real-time.

Table 1 shows frame rates measured during rendering of different objects using different levels *n* of approximation. Table 2 shows frame rates measured when applying virtual reflections to multiple objects. The achieved frame rates show that the level of approximation *n* has just a minor influence on the performance of the proposed technique. Furthermore the complexity of the virtual objects does not increase the overhead in rendering when using virtual reflections, because it is a single pass rendering technique and time needed for constructing the cube map textures is independent of the scene complexity.

Table 1. Measured frame rates for single objects using different approximations.

	no reflections	reflections (*n*=32)	reflections (*n*=128)
sphere (see figure 10 (b))	59.98 fps	59.92 fps	59.87 fps
teapot (see figure 10 (c))	59.92 fps	59.86 fps	59.84fps
car model (see figure 7 (a))	8.73 fps	8.55 fps	8.55 fps

Table 2. Measured frame rates for multiple objects using different approximations.

	reflections (*n*=32)	reflections (*n*=128)
sphere and torus r=2 (see figure 6 (a))	23.26 fps	21.49 fps
sphere and torus r=6 (see figure 6 (b))	10.17 fps	9.98 fps

(a) cube (b) sphere (c) teapot

Fig. 10 Examples showing that virtual reflections appear more realistic on objects with a higher degree of curvature.

Applying virtual reflections to scenes augmented with multiple virtual objects reflecting each other results in a significant decrease of performance. Because the recursion depth *r* determines the number of needed rendering passes (see section 3.3), high values of *r* further decrease performance as it can be seen in table 2.

For the performance tests a Pentium 4, 3 GHz system, running Windows XP Professional, equipped with 1 GB RAM and a NVidia GeForce FX5900XT graphics card with 128 MB RAM has been used.

6. Conclusion

In this paper we have presented a rendering technique for simulating reflections in augmented reality environments.

The proposed technique called virtual reflections is easy to implement and can be used in real-time augmented reality environments to generate convincing reflections on the fly for single as well as multiple virtual objects rendered into a real scene. Based on the described approach it is easy to generate cube map textures, which can be processed by current graphics systems and simulate reflections on virtual objects having materials with different degrees of reflectance.

(a) indoor scene (b) outdoor scene

Fig. 11 Examples showing reflections on different 3D models.

The application of virtual reflections and the possibility to combine them with existing approaches to simulate lighting and shadow casting results in an improved quality of real-time augmentation for still images as well as animations. Since virtual reflections can be applied in real-time the technique is adequate for the use in applications like the magic book [1] or augmented table environments [10], where the real world as seen by the user, usually by looking through head-mounted displays, is augmented in real-time.

References

1. M. Billinghurst, H. Kato, I. Poupyrev: “The Magic-Book - Moving Seamlessly between Reality and Virtuality”. In IEEE Computer Graphics and Applications , volume 21, no. 3, pages 6-8, May 2001.

2. K. Agusanto, L. Li, Z. Chuangui, N. Wan Sing, K. Chee Keong: “Photorealistic Rendering for Augmented Reality Using Environment Illumination”. In IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR03), page 208, October 2003.

3. M. Haller, S. Drab, W. Hartmann: “A Real-Time Approach for an Augmented Reality Application Using Shadow Volumes”. In ACM Symposium on Virtual Reality Software and Technology (VRST03), pages 56-65, 2003.

4. J. F. Blinn, M. E. Newell: “Texture and Reflection in Computer Generated Images”. In Communications of the ACM, volume 19, no. 10, pages 542-547, October 1976.

5. N. Greene: “Environment Mapping and Other Applications of World Projections”. In IEEE Computer Graphics and Applications, volume 6, no. 11, pages 21-29, November 1986.

6. T. Naemura, T. Nitta, A. Mimura, H. Harashima: “Virtual Shadows in Mixed Reality Environment Using Flashlight-like Devices”. In Transactions of the Virtual

Reality Society of Japan, volume 7, no.2, pages 227–237, June 2002.

7. C. Loscos, M.-C. Frasson, G. Drettakis, B. Walter, X. Granier, P. Poulin: “Interactive Virtual Relighting and Remodeling of Real Scenes”. In D. Lischinski and G. Larson (editors), Rendering Techniques ’99 (Proceedings of the 10th Eurographics Workshop on Rendering), volume 10, pages 235–246, June 1999.

8. P. E. Debevec: “Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography”. In Proceedings of SIGGRAPH 98, Computer Graphics Proceedings, Annual Conference Series, pages 189-198, July 1998.

9. J. Kautz, K. Daubert, H.-P. Seidel: “Advanced Environment Mapping in VR Applications”. In Computers & Graphics, volume 28, no. 1, pages 99-104, February 2004.

10. R. Grasset, J. D. Gascuel: “MARE: Multiuser Augmented Reality Environment on Real Table Setup”. In ACM SIGGRAPH Conference Abstracts and Applications, 2002.