Automatic generation of consistent shadows for Augmented Reality

Katrien Jacobs
Department of Computer Science
University College London

Cameron Angus
Department of Computer Science
University College London

Celine Loscos
Department of Computer Science
University College London

Jean-Daniel Nahmias
Department of Computer Science
University College London

Alex Reche
INRIA/REVES
Sophia-Antipolis

Anthony Steed
Department of Computer Science
University College London

Abstract

In the context of mixed reality, it is difficult to simulate shadow interaction between real and virtual objects when only an approximate geometry of the real scene and the light source is known. In this paper, we present a real-time rendering solution to simulate colour-consistent virtual shadows in a real scene. The rendering consists of a three-step mechanism: shadow detection, shadow protection and shadow generation. In the shadow detection step, the shadows due to real objects are automatically identified using the texture information and an initial estimate of the shadow region. In the next step, a protection mask is created to prevent further rendering in those shadow regions. Finally, the virtual shadows are generated using shadow volumes and a pre-defined scaling factor that adapts the intensity of the virtual shadows to the real shadow. The procedure detects and generates shadows in real time, consistent with those already present in the scene and offers an automatic and real-time solution for common illumination, suitable for augmented reality.

Key words: Augmented reality, image based rendering, common illumination.

1 Introduction

A wide range of applications use computer-generated animations in combination with pictures or footage of real scenes. Examples of such mixed reality applications are manufacturing, medical training, medical surgery, entertainment, education or cultural heritage. Some (e.g. surgery [26]) require that the inclusion of the virtual elements in the real scene is instantaneous, without pre-processing time and without having a model of the real scene. This implies numerous calibration and registration problems when inserting the virtual elements [2, 3]. Others (e.g. cultural heritage [19]) allow pre-processing and use a reconstruction model of the real scene, allowing a more accurate calibration and registration of the virtual elements within this model.

Important progress has been made in registration and calibration [3, 18]. However, the rendering of the virtual objects within the real scene often remains inconsistent: their illumination is different from that of the real objects. Few attempts are made to understand the current lighting in order to compute the shading. It is clear that a consistent illumination improves the overall impression of the mixed reality. First, a consistent shadow of the virtual objects gives a correct interpretation of the relative distance between the virtual objects and the real objects. Second, a correct lighting enhances the feeling that the virtual objects are part of the real scene.

Since the early nineties [16] a few solutions for the illumination inconsistency have been proposed. Most of them assume that a model of the real scene is available. This reconstruction can be obtained with a scanner, but most commonly real scenes are reconstructed using photographs from different viewpoints [22, 10, 21, 23, 15, 8]. The latter is a photogrammetric, error prone process as the reconstruction usually contains a mismatch between the simplified geometry and the texture. Due to the various errors, shadows generated using the approximated geometry and light source position may not align with the shadows visible in the texture.

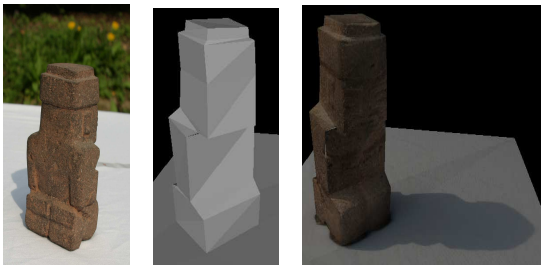


Figure 1: Virtual model obtained using stereopsis. Left: outdoor scene set-up. Middle: reconstructed geometry. Right: geometry and texture combined, the shadow is visible in the texture. A clear mismatch between geometry and texture is visible, the curved shape of the shadow indicates that the geometric model is inaccurate.

In Figure 1 (Middle) an example is shown of the 3D reconstructed model of a real object photographed in a real outdoor environment (Left). The reconstructed geometry is a fair but simple approximation of the real shape that consists of many curved surfaces. A clear mismatch between texture and geometry is visible (Right). Due to the misalignments resulting from the reconstruction, it may not be straightforward to extract the illumination properties accurately [20]. One can spend more time on the geometry reconstruction or use a more accurate geometry reconstruction procedure (e.g. laser) to guarantee that all geometry parameters are exact, but this adds an overhead cost to the application. In this paper a new procedure is presented that offers a solution for rendering consistent shadows in real time for mixed reality, regardless of the quality of the geometric reconstruction.

1.1 Related work

Jacobs et al. [16] provid a classification of the illumination methods for mixed reality. The methods for consistent rendering can be classified in two categories: common illumination and relighting. Common illumination simulation provides a consistent lighting when virtual objects are inserted in a real context. Relighting goes one step further and enables the modification of the original illumination. Using inverse illumination [20] is one way to achieve common illumination or relighting. This refers to the retrieval of the illumination parameters such as the BRDF of the objects in the real scene. In the classification of Jacobs et al. [16], some methods provide a real-time rendering for common illumination [5, 1, 13, 11]. However to our knowledge no method has yet been proposed that provides a real-time illumination estimate. Most of the time, a lot of effort is spent in the pre-computation step to find an accurate model of the real scene and the lighting properties via inverse illumination methods. After the geometry is captured, the pre-computed set up usually takes a minimum of a few minutes to several hours.

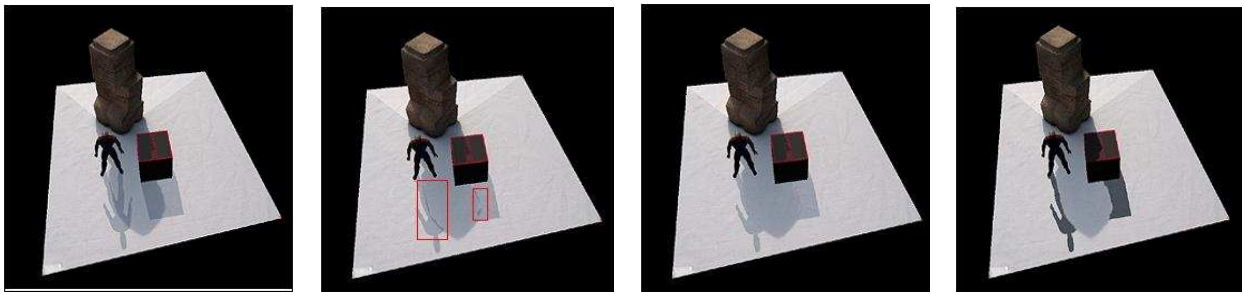
Illumination methods in computer graphics can consider local and global illumination. Local illumination effects simulate only the direct effects due to a light source, global illumination deals with the inter-reflections between objects as well. While it is important to consider the global effect of an added virtual object, local illumination is essential, especially for shadows. Recent developments provide the possibility of simulating shadows at low cost, using for example shadow maps [6] or shadow volumes [9]. A recent survey [14] presents the classification of soft shadow methods that could be used in the context of mixed reality. State et al. [27] demonstrate the use of shadow maps in an augmented reality system. To achieve a similar effect Haller et al. [13] use shadow volumes. While these two methods showed impressive improvement in the capability of registration, overlap between virtual and real shadows were avoided. Gibson et al. [11, 12] developed two methods for simulating soft shadows of virtual objects in complex mixed reality environments. While the rendering of the shadow is interactive, the method requires accurate input geometry and a good estimate of the BRDF and original illumination. From the examples shown in the papers, it is also not clear if the overlap between real and virtual shadows are handled. The method presented in this paper provides a real-time system both for processing the original shadow regions due to real objects and rendering new shadows due to the insertion of virtual objects. Therefore overlap between real and virtual shadows are treated and the shadows of virtual objects due to a real light source are consistent with the shadows of real objects.

1.2 Method overview

In this paper, a solution is presented to provide consistent shadows between virtual and real objects for a scene with one main real light source. The geometry of the considered real objects and the position of the real light source only need to be known approximately. Algorithms such as State et al. [27] or Haller et al. [13] to render the shadows of the virtual objects, provide a solution similar to the one shown in Figure 2(a), where the shadows of the virtual objects (virtual avatar and box) lie correctly on the ground as expected, but overlap incorrectly with the shadow of the real statue due to a real light source. The correct result should be as in Figure 2(c) which is computed using the algorithm presented in this paper.

When the virtual shadow is rendered using shadow maps or shadow volumes, those pixels that lie inside the virtual shadow have their intensity scaled using an appropriate scaling factor. When virtual and real shadow partially overlap one need to decide how to select the scaling factor. If the scaling factor is 1 (no scaling), those pixels in the overlapping part have a colour consistent with the real shadow pixels, the non-overlapping pixels, however, will be brighter than required for a consistent shadow. If the scaling factor is smaller than 1, the pixels in the non-overlapping areas will have a colour similar to that of the real shadow pixels, but the pixels in the overlapping area will be too dark, making the shadow, again, inconsistent. Based on this information, a three-step mechanism is designed and each step is briefly discussed here; a schematic overview is given in Figure 3.

The **shadow detection step**, deals with the detection of the position and shape of the *real shadows* of the real objects in the scene, conform with the texture of the real scene. It takes as input the scene geometry, its texture and an approximation of the light source position present in the scene. First a shadow contour estimate is calculated



(a) (b) (c) (d)

Figure 2: (a) Generation of the virtual shadows using one scaling factor. (b) Pixels inside the geometrical estimate of the real shadow are protected against the scaling. Due to misalignments not all pixels of the real shadow are protected. (c) When using an appropriate scaling factor and protecting the shadow pixels identified by the shadow detection step, the virtual shadows are consistent with the real shadow. (d) The virtual shadow is inconsistent with the real shadow when an inappropriate scaling factor is used.

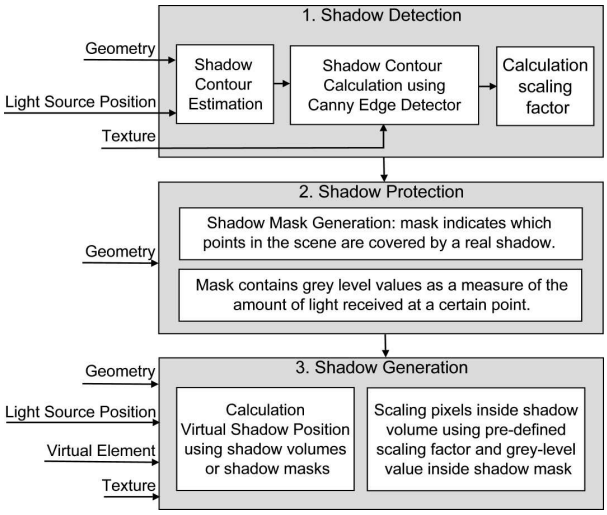


Figure 3: Three-step mechanism: a schematic overview of the three steps and their interaction.

using the geometry and the light source position. Due to the reconstruction errors, mentioned in the introduction, and the error in the registration of the light source in the scene, this estimate deviates from the shadows visible in the texture. Next, the exact shadow contour is extracted, for instance by using a Canny edge detector [7] in a region defined by the shadow contour estimate.

Once the true shadow contour is known, it is possible to calculate a scaling factor per material in shadow that reflects the colour intensity in the shadow region. This factor relates the colour of the material in shadow with the one not in shadow. A more detailed explanation of the shadow detection step is given in section 2.

In the **shadow protection step**, a binary shadow mask is created that is used to protect those points inside a real shadow from any scaling. The scaling factor is chosen to match the colour of the non-overlapping areas with the points inside the real shadow. If the geometrical estimate is used to protect the real shadow points, this results in a misalignment of the shadow mask with the real shadow as can be seen in Figure 2(b). When the output of the shadow detection step is used to complete the shadow mask, all points inside the real shadow are correctly blocked from further rendering: the resulting virtual shadow is consistent, see Figure 2(c). More details about the shadow protection step are given in Section 3.

In the **shadow generation step**, a real-time shadow method such as shadow maps or shadow volumes is used to generate the virtual shadows. The intensity of the shadow relates to the appropriate scaling factor computed previously; overlap between real and virtual shadows is prevented by using the mask generated in the previous step. Those parts of the virtual shadow that lie inside a real shadow are ignored. The intensities of the pixels in the virtual shadow in the non-overlapping regions are calculated by scaling the texture colour with the scaling factor. The influence of the choice of the scaling factor is illustrated in Figure 2(c) and 2(d). During the rendering, a different scaling factor needs to be used for different materials and for polygons with a different orientation towards the light source. Section 4 provides more details about this third and final step.

After presenting the three-step method, the performance of the system is illustrated using various scenes in a different context in Section 5, followed by a conclusion in Section 6.

2 Automatic shadow detection

2.1 Shadow contour detection

In order to protect the existing shadows in the scene from any post-processing, the shadow pixels in the texture need to be identified. Two types of shadows exist: soft shadows and hard shadows. Ideally both types of shadows are detected, but in practice it is much easier to detect hard shadows. This paper does not aim at developing a new state of the art shadow detection method but instead claims that the three-step mechanism is independent of the choice of shadow detection in this first step. In this paper a hard shadow detector is implemented. Hard shadows appear in scenes with bright light sources, e.g. sunny outdoor scenes, indoor scenes with one main light source. The presented shadow detector has been tested in both types of scenes with good results. However, it should be stressed that if a soft shadow detector is used the method can be applied on less restricted scenes.

A Canny edge detector [7] is implemented to detect shadow contours in a similar approach as in [4, 25, 24]. The main advantage of the Canny edge detector is that it returns one-pixel thick edges and that it effectively neglects edges due to noise because of the non-maximal suppression and hysteresis. The shadow detector uses a geometric estimate of the shadow contour to generate the contour of the real shadow. The geometric estimate of the shadow contour comes from the generated shadow computed from the approximate geometry and the light position. In general the shadow estimate gives a good indication of the position of the real shadow, but it can be inaccurate due to the error of the estimated light source position within the scene and the geometric approximation of the real scene geometry. An example of input and output of the Canny edge detector is given in Figure 4. This type of shadow edge detector guarantees a correct shadow contour detection when:

1. The position of the geometrical estimate is close to that of the real shadow, regardless of the difference in shadow shape or detail.

2. The shadow is a hard shadow or a soft shadow that shows a relatively high contrast with the background.

3. The contrast of the shadow and the background is larger than the contrast in the texture pattern of the background. Using an appropriate choice of the size of the smoothing filter and the upper and lower thresholds, the edges of the background texture will be suppressed against the edges of the shadow.

Usually an edge detector can only detect hard shadows and give a rough estimate of a soft shadow contour as long as the gradient of the soft edge is larger than the gradient of any other underlying structures in the texture. When the parameters of the edge detector are tuned to detect the soft edges, the amount of false edges due to noise for instance, increases as well. This restriction to hard shadows might be a limitation of the presented technique, since true hard shadows are rare in nature. Nevertheless, a pseudo-shadow detection algorithm is implemented with satisfying results, as is explained in Section 3. The pseudo-soft shadow detector achieves good results on outdoor sunny scenes and indoor scenes where the main light source is a spotlight directed onto the objects under consideration, this is discussed in Section 5.

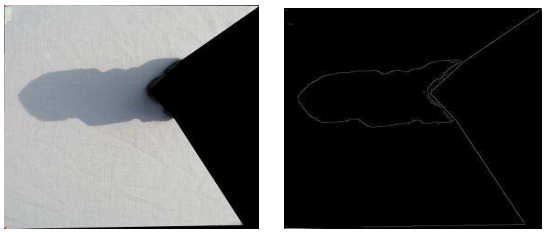


Figure 4: Left: the input texture image of the shadow detector. Right: based on a geometrical estimate, the shadow contour visible in the texture is extracted.

The computation speed of the shadow edge detector depends on the resolution of the input image. Since the most time consuming operations are the smoothing and gradient convolutions these can be implemented in the fourier domain, hereby decreasing the computation time. However, this is not implemented since the search window in which the edge detector operates is already drastically reduced by limiting the search to a region around the geometrical estimate of the real shadows. In other words, the computation speed rather depends on the size of the real shadows. When the real shadows cover a large region in the image, one can argue that the shadow edge detector might not operate in real time. But even in this case the size of the search window can be reduced if we consider only the region where real and virtual shadows overlap. In all experiments carried out, the shadow detection operated in real time (30 frames per second).

2.2 Automatic estimate of the shadow intensity

Once the shadow edge is defined, it is possible to calculate a scaling factor for each material in shadow. The scaling factor is in fact a triplet $[\rho_R, \rho_G, \rho_B]$, defined by dividing the three colour channels of the pixels inside the shadow with those outside the shadow. In the current implementation, a small region of pixels inside and a small region of pixels outside the shadow were selected and used to derive a mean scaling factor. The follow-

ing equation explains how the triplet is calculated. $C = \{R, G, B\}$ represents the colour channel: red, green or blue; SR stands for shadow region, while NSR stands for non shadow region; P_{SR} and P_{NSR} are the number of pixels in respectively the shadow and non shadow region:

$$\rho_C = \frac{\frac{\sum_{\forall p \in SR} C_p}{P_{SR}}}{\frac{\sum_{\forall p' \in NSR} C_{p'}}{P_{NSR}}}$$

The colour of the virtual shadow is now defined by the multiplication of the underlying texture value and the scaling factor. In general this scaling factor varies across the points on a surface, and depends on the orientation of that surface to the light source. Though it would be better to assign a scaling factor per pixel, only one scaling factor was used per material, but it has been identified that a better scaling factor selection procedure needs to be implemented for future applications.

3 Shadow protection

Based on the shadow contour, a binary mask is constructed. This mask is used to indicate which points in the real model are in shadow or not. In a way, it can be considered as a texture map that overlays the textures of the scene. The output of the shadow detector is an edge image, containing information about the shadow contour, at pixel level (see Figure 4, Right). Since the Canny edge detector detects edges of one-pixel thickness and removes most noise inside the image, the shadow region can be derived relatively easy: the shadow mask construction takes the edge map as input and uses a region growing algorithm to fill the shadow region inside the shadow contour. The starting point of the region growing can be any point inside the shadow region, which can be derived from the shadow estimate.

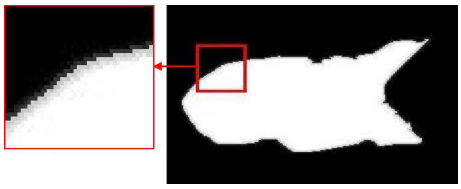


Figure 5: Left: close-up of a pseudo-soft shadow grey level mask, the gradient of the border protection is quadratic. Right: a binary shadow mask.

An example of a shadow mask is given in Figure 5 (Left). The edge detector can only detect hard shadows. Nevertheless the creation of a pseudo-soft shadow mask enables the use of the presented method on soft shadows. As is illustrated in Figure 5 (Right), a grey-level shadow mask is constructed with values ranging from 0 to 1 instead of a binary one. Those pixels inside the shadow are entirely protected (bright), the pixels outside the shadow are not protected (dark), the pixels at the shadow edge are partially protected (gradient grey level). The gradient in the soft region is quadratic to reflect the visibility variation in respect to an area light source. The support of the quadratic gradient increases with the *degree of softness*. This can be estimated if the distance of the light source and its area are approximately known. For instance for an outdoor scene, on a sunny day it is known that the sun has a solid angle of 0.5 degrees. Together with the geometry estimate and the direction of the sunlight an approximation of the degree of softness can be calculated . In the other cases, one can use the behaviour of the pixel intensities around the detected shadow edge. Up till now, the degree of softness has been defined manually.

The design of a shadow mask allows any upgrade of the system to soft shadow detection once an implementation is provided to detect soft shadows at pixel level. For each pixel a grey level can be written in the shadow mask, that indicates how much light it still receives, without modifying the shadow generation step.

4 Shadow generation

Once the real shadow regions are identified, it is possible to simulate the shadows due to the insertion of the virtual objects. These shadows can be generated with different algorithms. In the current implementation, shadow maps and shadow volumes are used, but any other real-time shadow generation procedure, including soft shadows, is suitable. The shadows are computed using the approximate position of the real light source and consider the interactions between virtual and real objects. Shadows from virtual objects are therefore cast on real and virtual objects, and real objects can cast shadows on virtual objects. Shadows due to real objects onto other real objects are already included. The intensity of the shadows are estimated using the scaling factor as computed in Section 2.2. This scaling factor differs for each object on which the shadow is cast. It should reflect both the material properties and the ambient light received in each region.

In order to prevent any overlap between the generated shadows and the real shadows the shadow mask computed in Section 3 is used to indicate the regions that can be drawn in the colour buffer. Regions with zero values (dark) in the shadow mask, can be overdrawn and regions with one values (bright) are fully protected. Regions with grey level values are partially protected. We use the shadow mask in two different ways. When a

textured model of the real scene is available, the multi-texturing capability of OpenGL is used to multiply the original texture and the shadow mask texture. When ARToolKit is used, we draw the shadow mask directly in the stencil buffer. In this implementation each (visible) scene point needs to have one entry in the shadow mask.

An alternative implementation could pass a geometrical shadow mask instead of the texture mask in the stencil buffer. The geometric mask can be created by representing the edges of the shadow contour as found in the shadow edge detection step analytically. However the texture mask ensures that complex shapes can easily be dealt with. For a more complex scene for which texture memory is an issue, a geometric mask can be extracted from the shadow detection and used in the stencil buffer.

5 Results

The presented method has been tested in two different contexts. One is computer augmented reality where a textured model representing the reality is used and the shadow detection is carried out offline. The other is augmented reality using a pre-estimated geometry of the real scene and texture information from a video camera. In both cases the interactions of the virtual objects are computed in real time.

5.1 Computer augmented reality

Figure 2 (c) already showed the result of the presented method on an outdoor scene using a pseudo-soft shadow map. Another example of a more complex scene is given here. A geometric model, shown in Figure 6 (a), was reconstructed from photographs of a real scene, see Figure 6 (b), using [15]. It is a fairly simple geometric model but it contains many different objects that cast shadows on different, sometimes textured materials. The scene was illuminated by a spotlight to create distinct shadows with a low degree of softness created by the indirect sunlight coming through the windows, see Figure 6 (b). Once the edge detector is used to find shadow edges, a shadow mask with soft borders is constructed as explained in Section 3.

Figure 6 (c)(d) illustrate the shadow protection and generation, using the calculated shadow masks shown in Figure 6 (f), when two virtual objects (a floating box and an avatar) are added to the scene. Figure 6 (e) gives a closeup of the shadow protection. It is clear that the smoothness introduced in the grey-level shadow mask is not sufficient to compensate for the soft shadow edges, but the approximation is of such quality that the user can hardly detect the shadow border when an appropriate scaling factor is used.

In this type of application, the light source position cannot change, since the texture information is only known for one distinct light source position. There is however no restriction on the viewpoint position.

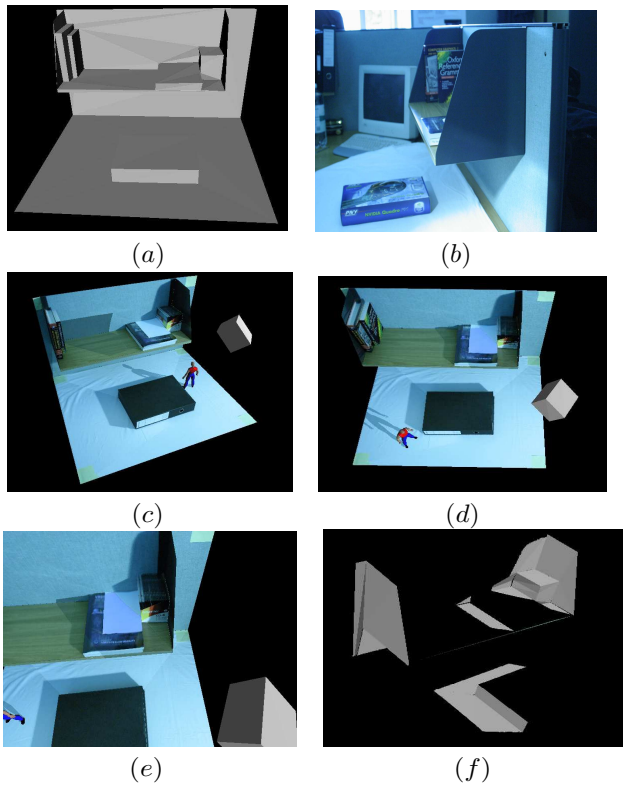


Figure 6: (a) The geometry extracted. (b) Scene set-up. (c), (d) Addition of two virtual objects (floating box and an avatar) in the scene projecting shadows onto the captured real objects. (e) A closeup view of the shadow of the box on the wooden shelf, interacting with real shadows. (f) The shadow masks projected on top of the geometry, white indicates that the pixels are in shadow.

5.2 Real-time application using ARToolkit

The second experiment uses ARToolkit [17] to perform real-time tracking of the camera. While the camera captures the scene, a virtual avatar walks across the real shadow borders. The virtual shadows of the avatar are rendered consistently within the real scene, using the scaling factor and the shadow mask calculated after the real-time shadow detection. In the same manner, the avatar receives shadows from the real object. The camera calibration is updated every frame and therefore the shadow mask also needs to be updated every frame, rather than calculating it once and incorporating it into the scene geometry.

The illumination sources in the scene are the light coming from the windows in the room and a spotlight di-

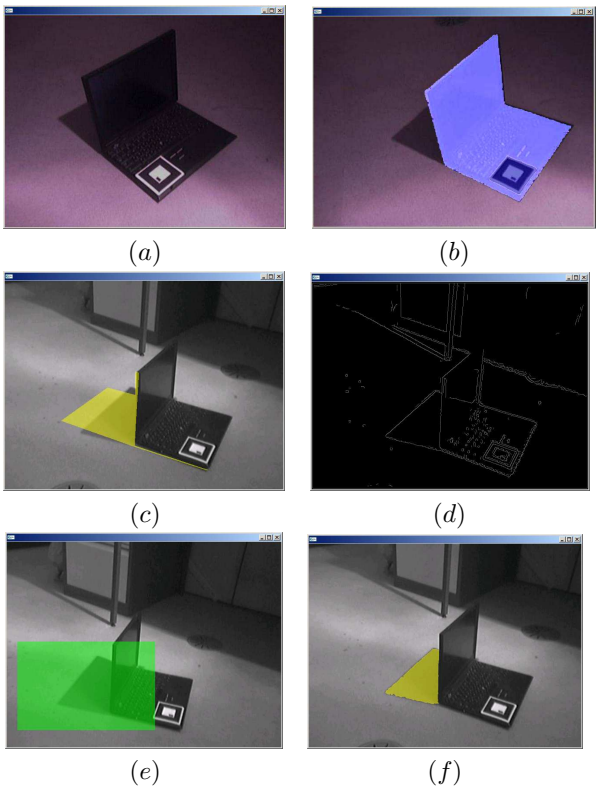


Figure 7: (a) A picture of the real scene set up. (b) The projection of the reconstructed laptop using ARToolKit. (c) Shadow as computed using the geometry input is shown in yellow. There is a clear mismatch with the real shadow. (d) Result of the edge detector. (e) Search region extracted from the shadow in (c), in which the shadow region detection is performed. (f) Result of the shadow mask generation.

rected to the real objects under consideration, to enhance the sharpness of their shadows. No further action has been undertaken to create more beneficial environment influences and therefore the shadows are not hard shadows. An example of the original scene and the process of the shadow detection and masking is shown in Figure 7 (a)(b), the projection of the known geometry is overlaid on the image. The computation of the shadow region using only the geometric position of the light source and the model is shown in (c). The position of the shadow at this stage is incorrect compared to the real shadow. In (d), the edge detection is shown. A search region, constructed from (c) is shown in (e). The shadow extracted from the region in (e) using our algorithm is shown in (f).

In Figure 8 examples of the shadow generation are shown with a virtual avatar walking around the computer.

The program runs at 15 to 30 frames per second on an input image of 320x240, and 10 to 15 frames per second on an input image of 640x480. These timings were recorded on a HP 3000+ Athlon with a GeForce FX 5950 Ultra. These timings include all computations and the camera capture at each frame; except for the geometry extraction and the light source registration no pre-computations are made.

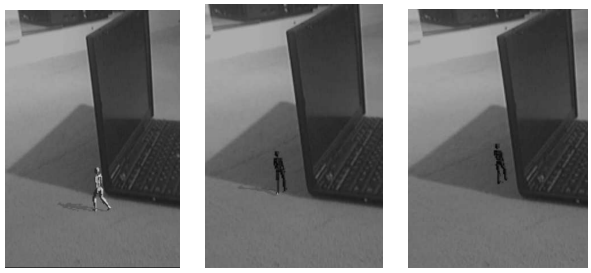


Figure 8: Results of the algorithm for an animated virtual avatar walking around a real laptop. Shadows are automatically detected and generated using our real-time algorithm.

It is important to notice that only a rough estimate of the shadow region is needed. The light source can be physically moved around its original position without making a position update in the system. The shadow detector has no problem with tracking the modified shadow position. The new position of the shadow can be used to track the position of the light source but this is currently not implemented.

6 Conclusions and future work

A real-time algorithm was presented that offers a consistent shadow simulation for mixed reality applications. There are three requirements for a successful shadow detection and generation: the geometry and the light source position need to be known approximately and only hard or semi-soft shadows are allowed.

The method consists of a three-step mechanism: shadow detection, shadow protection and shadow generation. This three-step algorithm generates consistent shadows between real and virtual objects in real-time. This allows the use of the presented method in augmented reality applications. Even though the shadow detection is not implicitly made for soft shadows, the experiments carried out show that the method manages to detect the shadow contours. To our knowledge no such real-time and automatic algorithm has been presented before that treats overlapping shadow regions between virtual and real objects.

We have shown results in two different contexts: com-

puter augmented reality and augmented reality. Further efforts need to be undertaken to scale the method to more complex environments. We will also search for ways in making our algorithm more robust for any types of soft shadows.

In this paper, we have considered only one light source and we have ignored the overall lighting conditions. It is therefore possible that the shadow of a virtual object falls on a highlight making the blend unrealistic. In future work we would like to provide a more general algorithm that will adapt to several light sources and compensate for illumination effects.

Acknowledgements

This project was partly funded by CREATE (IST-2001-34231), a 3-year RTD project funded by the 5th Framework Information Society Technologies (IST) Programme of the European Union, and by the UK's EQUATOR Interdisciplinary Research Collaboration (EPSRC Grant GR/N15986/01).

References

[1] K. Agusanto, L. Li, Z. Chuangui, and N. W. Sing. Photorealistic rendering for augmented reality using environment illumination. In *Proceedings of ISMAR 03*, pages 208–216, 2003.

[2] R. Azuma. A survey of augmented reality. In *Proceedings of Siggraph 1995, Course Notes ♯9: Developing Advanced Virtual Reality Applications*, pages 1–38, 1995.

[3] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics Applications*, 21(6):34–47, 2001.

[4] A. Bevilacqua. Effective shadow detection in traffic monitoring applications. *WSCG*, 10(2):57–64, 2003.

[5] O. Bimber, A. Grundheimer, G. Wetzstein, and S. Knodel. Consistent illumination within optical see-through augmented environments. In *Proceedings of ISMAR 03*, pages 198–207, 2003.

[6] J. Blinn. Me and my (fake) shadow. *IEEE Computer Graphics Application*, 8(1):82–86, 1988.

[7] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[8] Canoma. www.metacreations.com/products/canoma.

[9] F. C. Crow. Shadow algorithms for computer graphics. In *Computer Graphics (Proceedings of SIGGRAPH 77)*, volume 11, pages 242–248, 1977.

[10] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Computer Graphics (Proceedings of SIGGRAPH 96)*, pages 11–20, 1996.

[11] S. Gibson, J. Cook, T. Howard, and R. Hubbold. Rapid shadow generation in real-world lighting environments. In *Proceedings of Eurographics Symposium on Rendering 03*, pages 219–229, 2003.

[12] S. Gibson and A. Murta. Interactive rendering with real-world illumination. In *Proceedings of Eurographics Symposium on Rendering 00*, pages 365–376, 2000.

[13] M. Haller, S. Drab, and W. Hartmann. A real-time shadow approach for an augmented reality application using shadow volumes. In *Proceedings of VRST 03*, pages 56–65, 2003.

[14] J. Hasenfratz, M. Lapierre, N. Holzschuch, and F. Sillion. A survey of real-time soft shadows algorithms. In *Eurographics*, 2003. State-of-the-Art Report.

[15] Image Modeller. www.realviz.com.

[16] K. Jacobs and C. Loscos. Classification of illumination methods for mixed reality. In *Eurographics*, 2004. State-of-the-Art Report.

[17] H. Kato, M. Bilinghurst, B. Blanding, and R. May. Artoolkit, 1999. Technical Report Hiroshima City University.

[18] V. Lepetit, L. Vacchetti, D. Thalmann, and P. Fua. Fully automated and stable registration for augmented reality applications. In *Proceedings of ISMAR 03*, pages 93–102, 2003.

[19] C. Loscos, H. Ritter Widenfeld, M. Roussou, A. Meyer, F. Tecchia, G. Drettakis, and et al. The create project: Mixed reality for design, education, and cultural heritage with a constructivist approach. In *Proceedings of ISMAR 03*, 2003.

[20] G. Patow and X. Pueyo. A survey on inverse rendering problems. *Computer Graphics forum*, 22(4), 2003.

[21] Photomodeller. www.photomodeler.com.

[22] P. Poulin, M. Ouimet, and M.-C. Frasson. Interactively modeling with photogrammetry. In *Proceedings of Eurographics Symposium on Rendering 98*, pages 93–104, 1998.

[23] Renoir. www.integra.co.jp/eng/products/renoir.

[24] E. Salvador, A. Cavallaro, and T. Ebrahimi. Cast shadow segmentation using invariant color features. *Computer vision and image understanding*, 95(2):238–259, 2004.

[25] E. Salvador, P. Green, and T. Ebrahimi. Shadow identification and classification using invariant color models. In *Proceedings of ICASSP 01*, volume 3, pages 1545–1548. IEEE, 2001.

[26] A. State, D. T. Chen, C. Tector, A. Brandt, H. Chen, T. Ohbuchi, M. Bajura, and H. Fuchs. Case study: Observing a volume-rendered fetus within a pregnant patient. In *Proceedings of IEEE Visualization 94*, pages 364–368, 1994.

[27] A. State, G. Hirota, D. T. Chen, W. F. Garrett, and M. A. Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In *Computer Graphics (Proceedings of SIGGRAPH 96)*, pages 429–438, 1996.