Occlusion Handling in Video-Based Augmented Reality Using the Kinect Sensor for Indoor Registration

Jesus Adrián Leal-Meléndrez, Leopoldo Altamirano-Robles, and Jesus A. Gonzalez

National Institute for Astrophysics, Optics and Electronics, Computer Science Department,
Luis Enrique Erro No. 1, Tonantzintla, Puebla, Mexico
{jalm,robles,jagonzalez}@ccc.inaoep.mx
http://ccc.inaoep.mx

Abstract. Video-based Augmented Reality (VAR) aims to add 3D virtual objects (3D VOs) to a real world video sequence, in order to provide additional and useful information to facilitate some tasks, like computer aided surgery, simulation in a real environment, satellite positioning, interior design, among others. To achieve a consistent and convincing augmented scene, it is necessary that the VOs are properly occluded by real objects (Occlusion Problem in VAR); in this paper, we present a strategy based on the use of the *Kinect* sensor to solve this problem. In the occlusion stage we evaluate distances between real and VOs. Then, the parts of the VO occluded by a real object are calculated and removed. We found that the *Kinect* sensor is appropriate to be used for handling occlusions in indoor environments, dynamic scenarios and real-time applications. Experiments showed comparable results with the state of the art in both issues: occlusion handling and processing time.

Keywords: occlusion handling, video based augmented reality, hidden surface removal, kinect.

1 Introduction

Augmented Reality (AR) could be the answer for the growing demand of new user interfaces, in which space is not restricted to a screen and controls become unnecessary. AR adds 3D virtual objects (3D VOs) to a real scene, allowing the superposition of computer-generated graphics on real world scenes, in such a way that both look as a part of the same 3D scene [6]. In this way, a user can receive useful information in real time and in the most adequate place (real environment) and be guided in a determined task. Nowadays several applications in areas such as medicine, entertainment, education, architecture, among others, use AR; soon, even more areas will benefith from it.

An important task in order to create a synthetic *realistic* scene, is to align virtual and real objects in two ways: geometrical (spatial precision) and semantical (graphic credibility) [4]. Spatial precision requires the *3D VOs* to be appropriately registered in the real world, which means that they always must be in the right position and orientation with respect to the world. On the other hand, graphic credibility refers to the scene realism, i.e., the illusion of both elements, virtual and real, coexisting at the same spatiotemporal place. Graphic credibility has two main branches: the photo-realism,

J. Ruiz-Shulcloper and G. Sanniti di Baja (Eds.): CIARP 2013, Part II, LNCS 8259, pp. 447–454, 2013.

wich deals with illumination effects such as shadows and reflections, and occlusion handling, which requires that the *3D VOs* are correctly occluded by real-world elements.

The occlusion problem consists of determining which objects, real or virtual, are visible from a given vision angle and, based on that, hiding certain elements from the user view, all of this considering a 3D environment. Occlusion occurs when an object close to the user hides a further object on the same vision line.

According to some of the most recent works in the literature [3], [1], the use of depth information about the real world has lead to better results in the occlusion handling in AR. In this approach several stereo vision systems and 3D cameras have been used to calculate the distances in the real world. Despite leading to better results, the use of these technologies brings some problems: the intensity image and the depth map are not aligned and have low resolution, some stereo vision systems require excesive processing time and are inadequate to be used in real time, equipment is expensive and unaccesible to most users and, finally, some systems rely on big hardware and are not adequate to mobile configurations.

In this work we propose a strategy based on depth information and visual markers tracking. Our method combines the well known framework *ARToolKit* with the *Kinect* sensor to deal with, respectively, the positioning and AR occlusion issues. Moreover, we add a processing stage to correct the depth map and present our related conclusions.

This work improves the existent related works in the following aspects: 1) the use of the *Kinect* sensor allows us to work in real time environments and mobile configurations with resolutions above 640×480 pxs; 2) the tracking of visual markers allows correct registration of virtual objects (position and orientation); and 3) the parallel implementation (tracking and depth improvement) makes possible to work in real-time applications.

The rest of this work is organized as follows: section 2 shows a summary of related work, section 3 introduces our method and each of its parts, section 4 decribes the methodology we used to perform experiments and the obtained results; finally, section 5 presents our conclusions and future work.

2 Related Work

The first efforts to solve the occlusion problem in *AR* are focused on the segmentation of images. The main idea of this approach is to segment the real object that must occlude the *VO*; then, the *VO* is drawn on the real scene and, finally, the previously segmented region is put on top, in such a way that the real object occludes the *VO*. Some works that use this approach are [6] and [7].

In recent years, with the emergence of stereo vision systems and TOF cameras, the use of depth information has become the dominant approach to occlusion handling. A method to solve the occlusion problem in *VAR* is proposed in [2]. The authors use stereo vision and contour matching to calculate the depth of the objects in the foreground (user hands). Due to the high processing cost, this work focuses in the particular case in which the user hands must occlude the *VO* and viceversa. In addition, as a result of the approach used to segment the user hands, the method is not appropiate to work with occlusive objects with different color and texture.

Zhu et al. [3] propose a probabilistic approach to handle occlusion in *AR* using depth information obtained from a stereo vision system. Instead of using only the estimated depth, their method combines depth, color and neighborhood information, therefore reducing the noise inherent to the stereo pair. In order to accelerate the matching process between images, the authors incorporate a color quantization method; they also introduce Mixed Gaussian Kernels to describe objects of interest and to background subtraction. Finally, the estimated depth is used, together with a color addition method and neighborhood information, to establish occlusion relations between the objects of interest (only in the forefront). Due to the high computational cost of this approach, the authors focus on handling occlusions by certain pre-defined objects of interest, thus disabling the proposed method to work on dynamic scenarios.

Dong et al. [1] propose an algorithm for occlusion handling using depth obtained by a high-resolution TOF camera (*PMD CamCube 3.0*) and technology based on hardware to supress the background illumination. The authors add a second camera in order to obtain a RGB image; the first task performed is the alignment of both images. Then, to handle the occlusion, they use the principle of hidden surfaces removal to draw on the scene only the parts of the *VO* that are not occluded by a real object. The main drawback of this work is the alignment stage between the RGB image and the depth map, which produces over-occluding VOs leaving blank gaps around the occlusor real objects. Furthermore, the use of a specialized high-cost camera makes this work inaccessible to most users.

3 Occlusion Handling

Considering the drawbacks of the works described in the previous section, we focus on a method that covers differents cases: the ability of handling occlusion relationships between several objects despite their shape, size or color; dynamic cases in which the scene changes over time; and the use of technologies accessible to the majority of users. Furthermore, we consider both, geometrical and semantic, aspects. In the former, we use visual marker tracking to align the *VO*; in the later, we handle occlusion in real time.

In this section we describe the three main stages of our proposed method. It is important to point out that the first two stages take place in a parallel way, speeding up the processing time and making our method suitable for real-time applications.

3.1 Markers Recognition

The stage of markers tracking makes use of the framework *ARToolKit*[1], with modifications that include the integration of *Kinect* as the video input device, and the implementation of the function *Automatic thresholding* based on *ARToolKitPlus* [2]. The *tracker* is initialized through a calibration file for *Kinect*, obtained in previous offline calibration stage and feeded with the RGB image delivered by *Kinect*. In the final recognition stage, the view model matrix is obtained and applied to the VO when this is rendered.

[1] `http://www.hitl.washington.edu/artoolkit/`
[2] `http://handheldar.icg.tugraz.at/artoolkitplus.php`

3.2 Depth Improvement Stage

The method we propose to handle occlusion is based on the evaluation of distances between real and virtual objects; for this reason, a precise depth map is required. Fig. 1(a) y (b) shows the original images delivered by the *Kinect* sensor. As we can see in 1(b), the depth map is contamined with noise. These black pixels represent blind spots to the sensor, in which it was not able to estimate the distance to the real objects; later, in the experiments section, we show the negative impact of this issue on the occlusion handling.

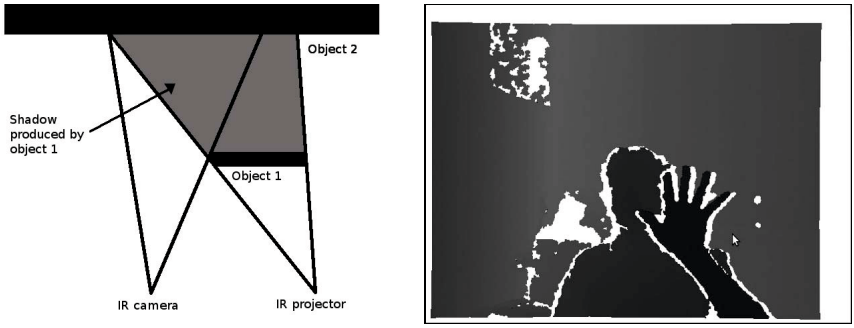
During the correction stage we evaluated different methods based on an inpaiting technique to correct the depth map. Fig. 1(c) shows the results obtained with the *inpainting telea* algorithm [5], which was the one with the best results when the corrected depth map was used for occlusion handling. In general, the methods based on inpainting techniques estimate the missing data using the neighborhood's information by expanding regions and they do not take into account the RGB image. Therefore, as shown in Fig. 1(c), the expanded regions in the depth map do not correspond to the original RGB image and there is a gap if both images are superimposed (we show this in the first experiment, in the section 4).

(a) RGB image (b) Original depth map (c) Corrected depth map

Fig. 1. Correction of the depth map: this figure shows the original images delivered by the sensor (Fig. 1(a) and 1(b)) and the depth map corrected by using the *inpainting telea* algorithm (Fig. 1(c)). Note that the black holes in (b), are calculated in (c) by using neighborhood information (with radius equal to 5).

After the black holes in the depth map are calculated, we can still appreciate a lack of alignment between the *RGB* image (Fig. 1(a)) and the depth map (Fig. 1(c)). In Fig. 3(b) the result in occlusion handling is shown when using the corrected map. We can see that the lack of alignment between the images is more evident when the scene is augmented. Analyzing the previous images we conclude that the main problem in the depth map is the effect we call *shadow effect*, which can be seen in Fig. 2(a). In this image there is a separation between the projector and the *IR* camera, and this results in a blind point between what is projected and what is seen. As a consequence, a black hole appears in object 2 that looks like a shadow of the object 1.

Taking into account this situation, we want to supress the shadow effect, since it belongs to object 2 (the furthermost object) and, therefore, it should not affect in an occlusion by object 1 (object in the foreground). Fig. 2(b) shows the depth map once



(a) Shadow effect explanation (b) Shadow effect suppression

Fig. 2. Shadow effect generated by the *Kinect* sensor

the black holes have been removed (they go from being in the foreground to being in the background). Fig. 3(c) shows the occlusion handling method using this idea.

3.3 Depth Buffering

Occlusion handling is performed applying the technique known as hidden-surface removal, which consists on removing object parts that are obscured by a closer object.

The distances obtained through the *Kinect* are in the range of $[0 - N]$ mm. Considering that the optimal range for the correct *Kinect* operation is between 50 and 4000mm, all the values that exceed 4000mm are scalated. Before they are written in the depth buffer, the vertices are transformed to the clip coordinates through the equation

$$cc = \frac{rd * (f + n)}{f - n} - \frac{2 * f * n}{f - n}, \tag{1}$$

then, they are normalized $ndc = \frac{cc}{rd}$ and forced to be in the range $[-1, 1]$. Finally, the values are transformed to the range $[0 - 1]$ by $fd = \frac{ndc+1}{2}$. Where rd is the distance obtained using the *Kinect* sensor (*raw data*), n and f are, respectively, the near and far plane projections, cc is the clip coordinates after the projection matrix, ndc represents the normalized device coordinates and fd is the final depth written on the depth buffer.

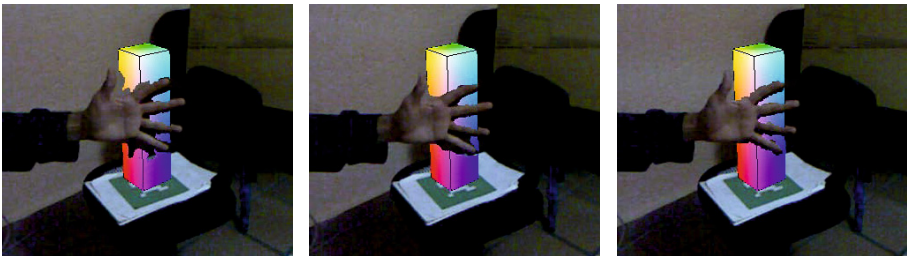
These distances are stored in the depth buffer and represent the distances of the real scene. When a new object is drawn on the screen, its depth is compared against the depth previously stored in the depth buffer, and only if the new object's depth is less, the VO is actually drawn.

4 Experiments

To perform our experiments, we built a system that integrates the acquisition, processing and display of an image. The experiments were performed inside a room (indoor configuration), where a user moved freely across the room and interacted with the virtual

content. All the experiments have the purpose of evaluating the robustness of the proposed strategy to solve occlusion relationships in *VAR*, and the impact of the quality of the depth map on the occlusion handling.

Experiment 1: Impact of the Depth Map. Fig. 3 shows the results of the stage of the depth map correction. Despite correcting the depth map, when VO occlusion is handled there is a lack of alignment with the RGB image; this translates into a poor occlusion (Fig. 3(b)). Fig. 3(c) shows the results of removing the shadow effect from the depth map; this technique gave better results.



(a) Original depth map (b) Depth map improved by *inpainting* (c) ***Shadow effect*** **removed**

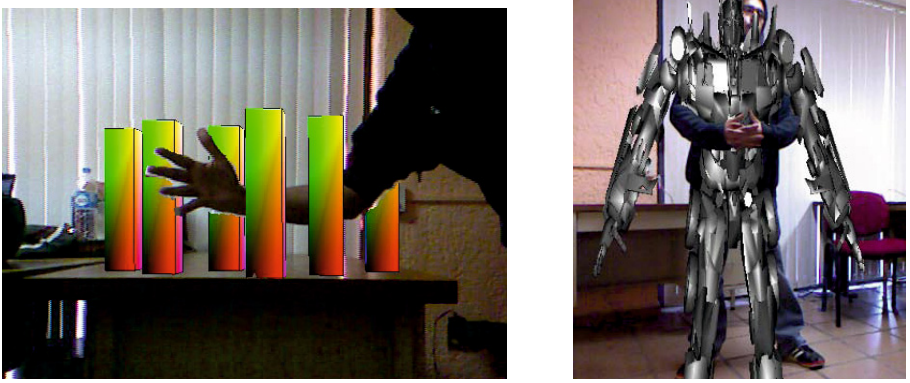
Fig. 3. Impact of the depth map on occlusion handling: (a) occlusion handling using the depth map delivered by the sensor, without processing; (b) occlusion handling using the depth map corrected by the *inpainting telea* algorithm; (c) occlusion handling with supression of the shadow effect

Experiment 2: Variant Lighting Conditions. In Fig. 4 we observe the handling occlusion under three scenaries with different lighting. In the three scenes we can see that the marker was recognized and the *VO* was correctly drawn over the marker and occluded by the user hand.

(a) Ideal lighting (b) Moderate lighting (c) Poor lighting

Fig. 4. Occlusion handling under variant lighting conditions

Experiment 3: Multiple and Deformable Objects. In this experiment, we worked with multiple and deformable objects. Fig. 5(a) shows the results of the proposed method when working with multiple VOs. In the image we can see the user interacting with virtual content and how the user's arm is occluded by VOs that are closer and occluding *VOs* elements located further away. Also, in this experiment we used more realistic *VOs*, with a bigger size and undefined shapes. Fig. 5(b) shows how the virtual elements are correctly (partially) occluded by the user.



(a) Multiple VOs (b) Deformable VOs

Fig. 5. Occlusion of multiple and deformable virtual objects. Image on the left shows that multiple virtual objects can be added to the scene and the proposed method is able to solve the occlusion relationships between them and the real objects. Image on the right shows partial occlusion of deformable virtual objects.

4.1 Discussion of the Results

The experiments showed that our proposed method can handle occlusion of deformable objects, multiple objects (occlusive and occluded) and work under different lighting conditions. It was also shown that the method is appropiate to work in environments demanding real-time response; the performed experiments reached a processing rate over 30 f/s.

Some of the drawbacks found in the use of the *Kinect* sensor are (1) the inability to handle occlusion with transparent or refracting objects, due to the fact that the sensor is not able to solve the object distance; and (2) a great sensibility to sun light, therefore the method is only appropiate for indoor configurations.

5 Conclusions

We have explored the use of a motion sensing input device, the *Kinect* sensor, in an Augmented Reality task. The experiments showed the feasibility of this method to

build augmented scenes properly occluded under indoor configurations and real-time; this could lead to new tasks for mobile robots, for example, by including AR in their navigation tasks. Furthermore, the obtained results are comparable to those of other works in the state of the art that use stereo vision systems and depth cameras with high economical and computational cost.

In our future work we are interested in exploring the use of algorithms that take into account the RGB image to correct the depth map, in such a way that the map can attain higher precision while maintaining the real-time requirement, and the removal of visual markers by calculating, instead, flat surfaces like tables, floors, walls, etc. Moreover, we would like to investigate the construction of a 3D model of the environment, so that we can keep virtual objects registered even when the camera angle changes.

Considering that we obtained good quality results, in different scenarios, with a low computational cost, we can say that the *Kinect* sensor is suitable for handling occlusions in AR applications.

Acknowledgments. This work was done under partial support of CONACyT-Mexico (scholarship 301754).

References

1. Dong, S., Feng, C., Kamat, V.R.: Occlusion Handling Method for Ubiquitous Augmented Reality Using Reality Capture Technology and GLSL. In: Proceedings of the 2011 ASCE International Workshop on Computing in Civil Engineering, Reston, VA, pp. 494–503 (2011)
2. Li, L., Guan, T., Ren, B.: Resolving occlusion between virtual and real scenes for augmented reality applications. In: Proceedings of the 12th International Conference on Human-Computer Interaction: Interaction Platforms and Techniques, Beijing, China, pp. 634–642 (2007)
3. Zhu, J., Pan, Z., Sun, C., Chen, W.: Handling Occlusions in Video-Based Augmented Reality Using Depth Information. Journal of Animation and Virtual Worlds: Wiley Online Library Computer 21, 509–521 (2010)
4. Zhu, J., Pan, Z.: Occlusion registration in video-based augmented reality. In: Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry, vol. 10, pp. 1–6 (2008)
5. Telea, A.: An Image Inpainting Technique Based on the Fast Marching Method. Journal of Graphics, Gpu, and Game Tools 9, 23–34 (2004)
6. Lepetit, V., Berger, M.-O.: Handling occlusion in augmented reality systems: a semi-automatic method. In: Proceedings EEE and ACM International Symposium on Augmented Reality (ISAR 2000), pp. 137–146 (2000)
7. Fischer, J., Bartz, D., StraBer, W.: Occlusion Handling for Medical Augmented Reality using a Volumetric Phantom Model. In: Journal of VRST 2004 Proceedings of the ACM Symposium on Virtual Reality Software and Technology, pp. 174–177 (2004)