Addressing the Occlusion Problem in Augmented Reality Environments with Phantom Hollow Objects

Jesús Gimeno
Department of Computer Science, University of Valencia

Sergio Casas
Department of Computer Science, University of Valencia

Cristina Portalés*
Department of Computer Science, University of Valencia

Marcos Fernández
Department of Computer Science, University of Valencia

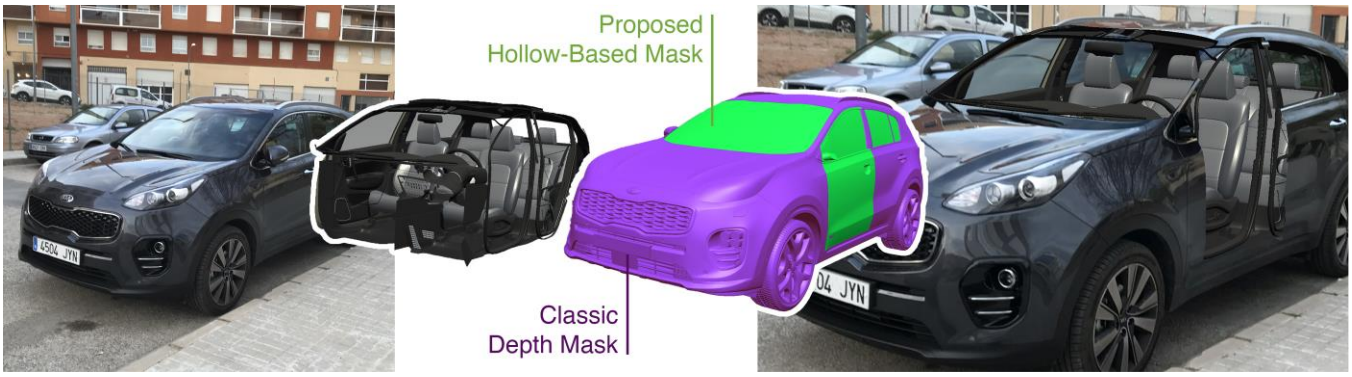

Figure 1: The proposed occlusion system. A real object (left) is augmented (right) with virtual content through the use of an inverse hole-based depth mask, shown with green color, instead of the traditional one, depicted in red color (middle).

ABSTRACT

Occlusion handling is essential to provide a seamless integration of virtual and real objects in AR applications. Different approaches have been presented with a variety of technologies, environment conditions and methods. Among these methods, 3D model-based occlusion approaches have been extensively used. However, these solutions could be too time-consuming in certain situations, since they must render all the occlusion objects even though they are invisible. For this reason, we propose an inverse 3D model-based solution for handling occlusions, designed for those AR applications in which virtual objects are placed inside a real object with holes or windows. With this restriction, the occlusion problem could be solved by rendering the geometry of transparent/hollow objects instead of rendering the opaque geometry. The method has been tested in a real case study with an augmented car in which the virtual content is shown in the interior of the vehicle. Results show that our method outperforms the traditional method, proving that this approach is an efficient option for solving the occlusion problem in certain AR applications.

Keywords: Augmented Reality, occlusion, hollow, inverse phantom objects, depth rendering, offline 3D reconstruction.

Index Terms: K.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities.

1 INTRODUCTION AND RELATED WORK

Occlusions between virtual and real objects are essential in Augmented Reality (AR) applications. An occlusion can be defined as the situation in which an object blocks the view of another object.

* Cristina.Portales@uv.es

In order to properly handle occlusions in AR environments, it is necessary to compare the depth of the pixels of the virtual objects with the depth of the pixels corresponding with real objects. Depth information (calculated relative to the user's point of view) about virtual objects can be easily obtained. However, it is not trivial to estimate the depth of the pixels corresponding with real objects.

Depth retrieving algorithms for AR applications can be broadly categorized in two groups: dynamic and static. The former group accounts for the most general approach in which real objects cannot be previously known and depth information is calculated in real time, whereas the latter deals with objects whose geometries are known and can be modelled *a priori*.

Although the dynamic approach can be implemented with different techniques, the most common one is the use of depth sensors (RGB-D or time-of-flight cameras) which are able to provide depth information (although depth resolution is typically lower than pixel resolution and they usually have a small working range) [1]. Other common approaches are stereoscopic vision [2] and the use of Simultaneous Localization And Mapping (SLAM) algorithms. However, the use of SLAM techniques in real-time non-static environments presents problems since these algorithms usually work with point clouds, which are converted into real-time polygon-based meshes [3]. These meshes are typically low-resolution shapes and the occlusion is usually not very accurate.

In the static approach, the occlusion problem is addressed by means of *phantom objects* [4, 5] which represent the geometry of the real objects of the scene. These phantom objects are rendered to the depth buffer but not to the frame buffer. Therefore, if proper virtual models of the real objects are provided, the method, yet simple, provides a way to handle any kind of static occlusion. However, an accurate model of the objects of the real scene involved in occlusions is needed, which could be very time-consuming. This is the most significant disadvantage of these methods, which could be also improved to account for objects that move but are not deformable, provided that they can be tracked.

Finally, ad-hoc occlusion effects can be achieved using computer vision techniques to identify real static or dynamic objects in the

scene [6] so that they can be annotated and sorted. An example of this approach are those AR applications that render the user's hands (segmented by contour or color) on top of virtual objects [7].

Despite the methods based in offline 3D models or in depth sensors represent the most common approaches, no perfect method exists for the retrieval of depth information. For instance, semi-transparent surfaces make some algorithms, like those based in some RGB-D cameras or stereoscopy, fail, since depth information should take into account visibility and not just the real depth of the object. This kind of transparent or hollow objects can be addressed with offline 3D reconstruction occlusion methods, but this may also require creating very complex geometries that are not necessary to see through a simple window or hole. This is one of the main reasons for proposing the method described in this paper, where an inverse approach, by focusing on the hollow surfaces instead on rendering the opaque objects, is designed. The main contribution of the paper is this new approach that provides several advantages, namely a time reduction (and thus cost reduction) in the offline phases and a lower computational requirement during the rendering of the augmented scene.

2 Proposed Inverse Method For Occlusion Handling

The method proposed in this paper is based on an inverse approach, focusing on hollow geometry instead of on the opaque objects. It works by using an inverse 3D model of the real objects and it is designed to work when virtual objects are placed inside real objects that have windows or holes. The rationale of using this method is that traditional static phantom objects are often too expensive and inefficient when it comes to render AR scenes in which there are windows and/or holes, through which virtual objects are seen. The proposed method is designed to improve the performance of the AR occlusion handling in those situations in which the hollow geometry is simpler than the rest of the real opaque objects that cover the virtual content. Despite simplified models are sometimes used in the traditional 3D reconstruction occlusion method, this increases the complexity of the offline stage and the result could be less accurate than when using the original model. In any case, this simplification could be also applied to our inverse approach, although it may not be necessary in most cases, since hollow geometry is typically simpler than opaque geometry. We talk about an *inverse* approach because instead of modelling the physical objects of the 3D environment, we model the non-physical objects, i.e. we model the holes where light (and therefore user's sight) can introduce. This *inverse phantom objects* imply also a shift, from an implementation point of view, of the process of blending virtual and real information.

In order to exemplify the conceptual difference of our approach in comparison to the traditional 3D model-based offline method, a schema has been built, which is shown in Figure 2. In the traditional static occlusion method based on 3D models, the process – assuming a video-based AR system (although it could be applied, as our approach, to other types of AR systems) – involves drawing the image from the real camera into the frame buffer (typically on the far plane). Later, the phantom object is rendered into the depth buffer (opaque geometry). Finally, the virtual objects are rendered into the frame buffer. This method does not support to handle real semi-transparent objects, although ad-hoc solutions can be applied, such as X-ray visualizations [8].

Our proposal solves the occlusion problem in a different way, which is summarized in the following steps:

(i) Draw the image from the real camera into the frame buffer (on the far plane). This step is only necessary if one or more of the virtual objects of the scene are transparent or semi-transparent.

(ii) Render the virtual objects to the frame buffer, with their corresponding depth information to the depth buffer, so that virtual objects are properly sorted (between them) in the render process.

(iii) Clear the depth buffer and then render the transparent/hollow geometry (holes, windows, etc.) into the depth buffer.

(iv) Finally, draw the image from the real camera on the far plane. Since the occlusion hollow objects were previously rendered into the depth buffer, video information will not be visible for those pixels corresponding with hollow geometry. Instead, the virtual objects will be visible, creating a correct occlusion effect.

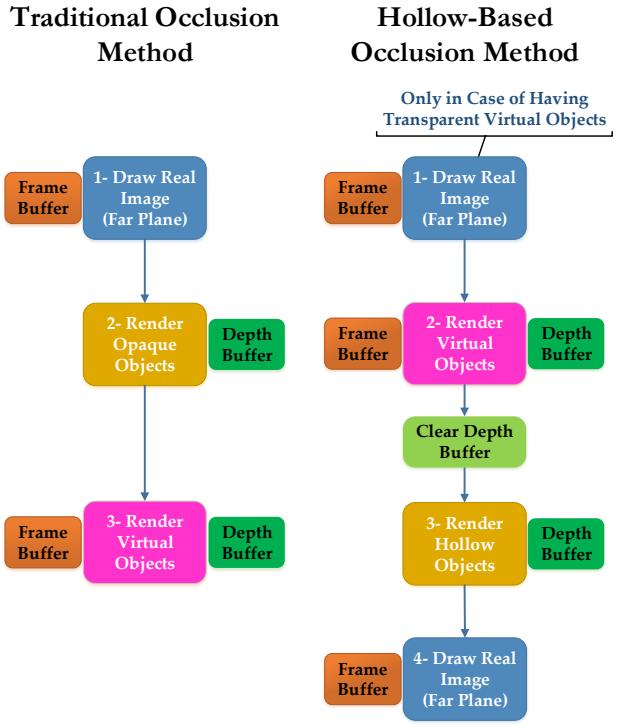

Figure 2: Scheme of the traditional 3D model-based offline method (left), and of the new proposed method (right).

An example is shown in Figure 3. In Figure 3(a) a physical object (a car model) that might cause occlusions has been modelled (traditional approach), where in Figure 3(b) only the holes have been modelled (new approach). As it can be seen, the former has a much more complex shape than the latter, even considering the car door as a transparent object, which is not, in order to allow the observation of additional virtual objects like the steering wheel.

As it has been already stated, our method is designed for those cases in which virtual objects are placed inside a real opaque solid object with holes. Besides this particular limitation, the method has similar features and advantages than the traditional offline 3D-based occlusion, but it is expected to be much faster than the traditional approach, since the geometry of holes and windows is typically less complex than the geometry of the complete object.

3 Experiments and Results

This section is dedicated to analyze the performance and resulting occlusion handling of the proposed method. Since the approach is a model-based method, we propose to compare its performance with the traditional 3D reconstruction method. A case study with an augmented car is analyzed next. Although the relative performance of the solution could be different for other AR environments, we believe this example could be a representative case in which the proposed method would be applicable. In fact, this particular AR application was developed as part of a driving safety exhibition alongside other IT-based solutions [9, 10]. The augmented car was used to show different driving safety tips.

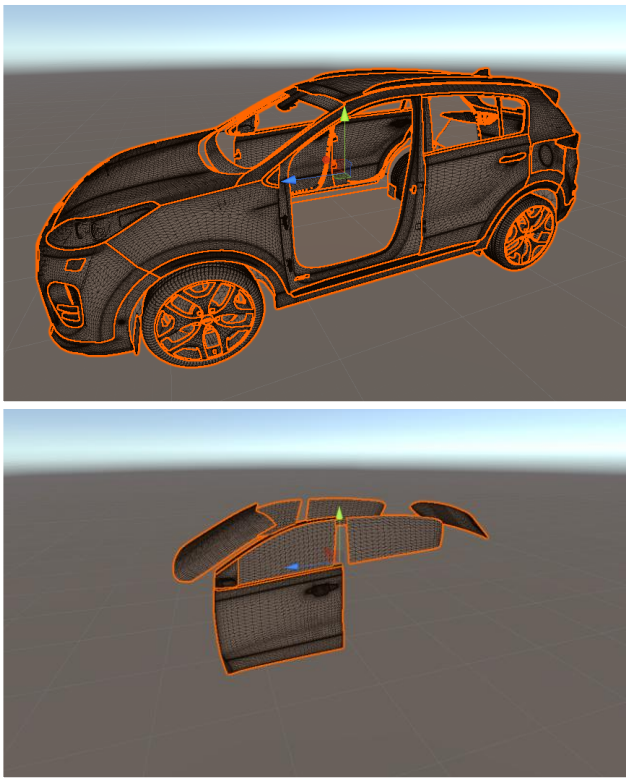

Figure 3: Differences in the occlusion geometries in a car model: (a) The traditional method (up) typically needs a much more complex phantom object than (b) the new proposed method (down).

Figure 4 shows the result of applying both occlusion methods, where a virtual recreation of the interior of a real Kia Sportage 2016 was used to augment this vehicle. Windows and even the driver's door are considered hollow geometries in this example, in order to enhance (virtually) the interior of the car. As it can be seen, occlusion is properly handled with either method. Windows are dual surfaces of the car body and the results are alike, although there are differences in the way the driver's door is treated. This illustrates that there might be small differences between the two methods in particular cases like this where there are complex extruded holes. Both solutions could be considered correct and the election of one or the other depends on the desired effect.

This AR application, which was implemented with Unity 3D and the libraries Vuforia and ARKit to track the user's point of view, was evaluated in terms of performance. An iPad Pro 10.5 was used and render times (time needed to render each frame) were measured every second during 60 seconds for each of the occlusion approaches, with a 10% moving average to smooth the results and avoid fluctuations (see Table 1). The results are also compared with the no-occlusion case. In order to show also the performance of the method when the amount of virtual content is very small, another experiment was conducted with a low-detail (LD) virtual model by placing only a virtual cube inside the car, instead of the high-detail (HD) model of the car's interior. In all cases, the phantom objects use the original geometry of the car (both in the traditional and in the inverse approach).

As it can be seen, the new method significantly reduces the time necessary to render the AR scene in both cases (HD and LD). In the first case (HD), the traditional method needs 28.936 ms (milliseconds) to render the scene, whereas 20.913 ms are needed with the new proposed method (close to the 19.732 ms needed when no occlusion is performed). In the second case (LD), the improvement is from 10.209 ms to 6.718 ms, with 6.629 ms for the no occlusion case. Standard deviation is small. Therefore, no statistical tests are needed to conclude that the proposed method outperforms the traditional one.

Figure 4: Result of occlusion handling with an augmented car: (up) the traditional method, and (down) the new inverse approach.

	Time per Frame - Mean (ms)	*Time per Frame - Std. Dev. (ms)*	*Total Number of Vertices Rendered*
No Occlusion – HD	19.732	0.521	1,020,541
Occlusion – HD	28.936	1.292	2,101,586
Inv. Occlusion – HD	20.913	0.512	1,048,884
No Occlusion – LD	6.629	0.158	28,000
Occlusion – LD	10.209	0.244	1,081,077
Inv. Occlusion – LD	6.718	0.215	28,375

Table 1: Compared performance of the occlusion methods.

It is true that the amount of improvement depends on the device and also on the scene being rendered. In this case, this new inverse occlusion approach reduces the number of vertices sent to the graphics pipeline – from 2,101,586 to 1,048,884 (roughly a 50% reduction), and from 1,081,077 to 28,375 (97% reduction!) in the two experiments performed. Whether this vertex reduction has a significant or relative impact on the application performance depends on several factors. Nevertheless, a reduction in rendering geometry can be expected to improve the performance, unless this reduction is too small to compensate the time needed to double-render the real image, in case semi-transparent or transparent virtual objects are used. Although we did not use this kind of objects in the car case study, the calculated performance evaluation includes this double render of the real image. Thus, the

performance results could even be improved if this stage is eliminated, provided that no transparent virtual objects are used.

4 Conclusion

In this paper, we present a novel method for handling the occlusion problem in AR applications. The method is conceived for its use in those cases where virtual objects must be placed in the interior of real objects. In this new approach, instead of rendering the opaque geometry to the depth-buffer, hollow geometry is rendered to the depth buffer. In order to provide proper occlusions, the traditional process is inverted: the real image is rendered after the hollow phantom objects have been rendered to the depth buffer.

The applicability of the proposed method has been demonstrated by testing this new inverse approach in a case study in which a car is augmented by means of virtual content shown inside the vehicle. The method shows a superior performance with respect to the traditional occlusion technique based on opaque models. The performance improvement with respect to the traditional method in our test scenarios is of 27.73% and 34.20 % respectively in terms of mean frame time, which represents a substantial gain. In other situations, and depending on the rendering device, the complexity of the scene and/or the case study, the performance could be even better. The more complex the geometry of the 3D phantom object is with respect to the geometry of holes or windows, the more suitable our method would be. However, in situations where the opaque geometry is not complex, it could not be the best choice.

Our method includes a double render of the real image. However, rendering hollow geometry to the depth buffer should be less expensive than rendering the opaque geometry. The final question is whether rendering the hollow geometry plus adding a second render of the real image would be more efficient than rendering opaque geometry to the depth buffer. In most cases, the answer would be yes. In addition, the first render of the real image is only necessary when there are transparent or semi-transparent objects in the virtual scene. Therefore, we can conclude that the proposed method should represent an improvement over the traditional one in many scenarios and could be a suitable choice for handling occlusions in AR scenes in which virtual objects are placed inside real static objects.

Future work includes a broader comparison with other occlusion methods, especially depth-based methods, and other case studies and performance experiments. In addition, the method could be improved by exploring ways to support virtual objects that are placed inside but also behind opaque objects with one or multiple overlapping windows. Finally, rendering optimizations could be implemented computing the boundary of the hollow objects and rendering only the virtual content visible behind that boundary with a specific culling method.

References

[1] J. Fischer, et al., "Using Time-of-Flight Range Data for Occlusion Handling in Augmented Reality," *Proc. IPT/EGVE*, 2007, pp. 109-116.

[2] Y. Lu and S. Smith, "Gpu-based real-time occlusion in an immersive augmented reality environment," *Journal of Computing and Information Science in Engineering*, vol. 9, no. 2, 2009, pp. 024501.

[3] R.F. Salas-Moreno, et al., "Dense planar SLAM," *Proc. Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, IEEE, 2014, pp. 157-164.

[4] J. Fischer, et al., "Occlusion handling for medical augmented reality using a volumetric phantom model," *Proc. Proceedings of the ACM symposium on Virtual reality software and technology*, ACM, 2004, pp. 174-177.

[5] J. Gimeno, et al., "An Occlusion-aware AR Authoring Tool for Assembly and Repair Tasks," *Proc. GRAPP/IVAPP*, 2012, pp. 377-386.

[6]Y. Tian, et al., "Real-time occlusion handling in augmented reality based on an object tracking approach," *Sensors*, vol. 10, no. 4, 2010, pp. 2885-2900.

[7] M. VanWaardhuizen, et al., "Table top augmented reality system for conceptual design and prototyping," *Proc. ASME 2011 World Conference on Innovative Virtual Reality*, American Society of Mechanical Engineers, 2011, pp. 395-405.

[8] C. Sandor, et al., "An augmented reality x-ray system based on visual saliency," *Proc. Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, IEEE, 2010, pp. 27-36.

[9] L. Vera, et al., "A Hybrid Virtual-Augmented Serious Game to Improve Driving Safety Awareness," *Book A Hybrid Virtual-Augmented Serious Game to Improve Driving Safety Awareness*, Series A Hybrid Virtual-Augmented Serious Game to Improve Driving Safety Awareness, 2017, pp. 293-310.

[10] S. Casas, et al., "On a First Evaluation of ROMOT—A RObotic 3D MOvie Theatre—For Driving Safety Awareness," *Multimodal Technologies and Interaction*, vol. 1, no. 2,6, 2017, pp. 1-13.