SHAPE FROM INCONSISTENT SILHOUETTE FOR FREE VIEWPOINT VIDEO

Jose-Luis Landabaso[*], *Leonardo Lizcano,*

Telefónica R&D, Video Technologies Dept.
Vía Augusta 177, Barcelona, Spain

Montse Pardàs

Tech. Univ. of Catalunya, Im. Processing G.
Jordi Girona 1-3, Barcelona, Spain

ABSTRACT

In this paper we present an efficient image-based rendering algorithm that obtains novel images from a set of views of the scene of interest. The approach described uses silhouette image data to compute the Visual Hull, the largest volume that is compatible with the silhouettes that delimit the objects of interest. Since the Visual Hull is not explicitly computed, this approach does not suffer from the quantization artifacts of volumetric approaches. In contrast to previous works, we explore how detection errors in the silhouettes affect the novel rendered view and propose a method to detect errors in the original silhouettes based on the consistency principle that states that the projection of the Visual Hull should exactly correspond with original silhouettes.

Index Terms— Free viewpoint video, shape from silhouette, inconsistency, image-based rendering.

1. INTRODUCTION

This paper is focused on free view-point video for immersive videoconferencing applications. A key part of immersive videoconferencing systems is being able to compose scenes bringing together all the conferees in a single view. In these applications it is important to have fast methods that generate a single view of the scene where the conferees can be rendered as if they had been seen from different viewpoints.

Many different methods have been proposed in the past. A great amount of these works [1, 2, 3] are based on the Image Based Visual Hull (IBVH) approach that was described by Matusik et al. [4]. The IBVH algorithm computes and shades 3D shapes (the Visual Hull) from silhouette image data, i.e., binary masks that delimit the regions to be reconstructed. Their system generates image-based models in real-time, performing geometric computations in the image space, therefore eliminating the resampling and quantization artifacts of other approaches. By taking advantage of epipolar relationships, all of the steps of the algorithm can be implemented in the image space of the reference images.

The ideas behind IBVH have served others in creating algorithms that render more photo-realistic images. These works usually take the initial IBVH and refine it by carving the initial volume using color information [1, 2, 3].

While previous works have successfully solved the problem of shading and achieve real-time operation, very little work has been done on studying how errors in the silhouettes affect the quality of the novel rendered view. This paper addresses the issue of silhouette error detection providing the basis for improving the mentioned algorithms above.

2. IMAGE BASED VISUAL HULL CREATION

IBVH is basically a ray-tracing algorithm. Given a desired virtual view, all the pixels are rendered in the following steps. First, optical distortion is corrected in all the reference (captured) images. Then, for each pixel to be rendered in the desired (virtual) view, a ray is traced from the optical center of the virtual camera through the point in the optical plane corresponding to that pixel (see Fig. 1). If the projections of a certain segment of the viewing ray match silhouettes in all the images, then the pixel is assumed to be observing part of the Visual Hull (VH), and is therefore rendered.

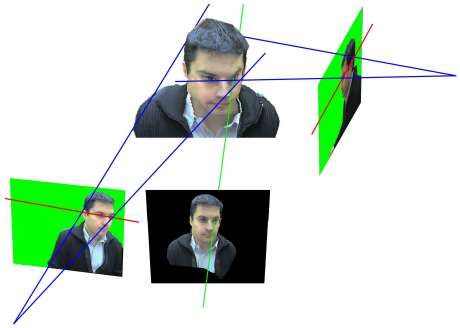


Fig. 1. IBVH is basically a ray-tracing algorithm.

In our approach, the 3D ray is traced progressively in optical center to optical plane order. Differently to other approaches, in our implementation the pixels for each 3D point in the ray are progressively projected until all the projections match a silhouette. If there are C matches, then an intersection has occurred. The details of the algorithm are as follows. First, the ray is scanned by taking steps of $\Delta = 1cm$ until an intersection is found. Then, the ray is retraced again in the

[*]This work has been partially funded by Spanish Project CENIT VISION.

intersecting area with a precision of $\Delta = 1mm$. In order to speed up the projection process, we do not compute all the projections for each step but simply update the pixel coordinates employing the following relations.

Let $\mathbf{X}' = \mathbf{X_0} + t\mathbf{\Delta}$, where $\mathbf{X_0}$ corresponds to the optical center of the desired view and t is the t-th point of the ray to be projected. Then, the location of the projection of a pixel $\mathbf{x}' = (x', y', w')$, in homogeneous coordinates, corresponds to $\mathbf{x}' = P_c\mathbf{X_0} + tP_c\mathbf{\Delta}$, where P_c is the projection matrix in a certain camera c. Finally, the pixel locations are

$$x' = \frac{(P_c\mathbf{X_0} + tP_c\mathbf{\Delta})_{[1]}}{(P_c\mathbf{X_0} + tP_c\mathbf{\Delta})_{[3]}}. \quad y' = \frac{(P_c\mathbf{X_0} + tP_c\mathbf{\Delta})_{[2]}}{(P_c\mathbf{X_0} + tP_c\mathbf{\Delta})_{[3]}},$$

where subindexs [1], [2] and [3] denote first, second and third components of a vector, respectively.

The implementation is fairly fast since most of the computations ($P_c\mathbf{X_0}$, $P_c\mathbf{\Delta}$) can be precalculated. In addition, we also use some heuristics such as start tracing from points that correspond to slightly advanced positions to the ones obtained in neighboring desired pixels that have been already calculated. As final remark, note that this approach does not require lifting back 2D segments and intersecting them later in 3D as most of the other approaches do.

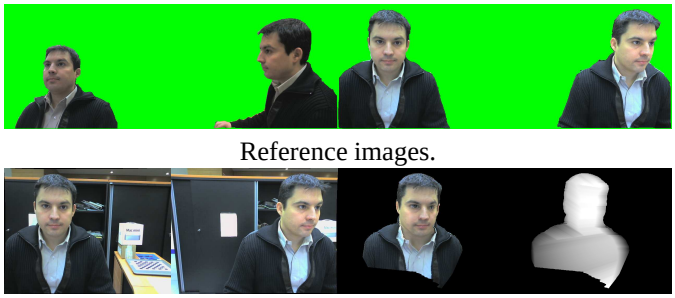
Obtaining the projection of the Visual Hull is only half of the problem. It is also important to determine the color in which the desired pixel will be rendered. There are two aspects to consider. The first aspect is deciding how to determine the visibility of the views and, second, how to weight the colors from the different reference views. Regarding the first aspect, in our implementation we only map colors from the pair of cameras that are on the left and right sides of the virtual camera. In our particular scenario, this simplifies the process of occlusion testing since we can assume, without introducing significant errors, that the silhouettes in these two reference views are unoccluded in the desired one. Color weighing is solved as follows:

$$\mathbf{C} = \frac{w_1 \cdot \mathbf{C_1} + w_2 \cdot \mathbf{C_2}}{w_1 + w_2}, \tag{1}$$

where $\mathbf{C_1}$ and $\mathbf{C_1}$ are the colors of the pixels in the reference images where the 3D point in the viewing ray has been projected. Weights are computed as:

$$w_i = \left(\frac{\left(\frac{v_d}{||v_d||}\right)^T \cdot \frac{v_i}{||v_i||}}{1 - \left(\frac{v_d}{||v_d||}\right)^T \cdot \frac{v_i}{||v_i||}} \right)^k, \tag{2}$$

where v_q is a column vector expressing 3D direction from the optical center of camera q to the 3D point that corresponds to the projection of the matched point in the viewing ray over the optical plane of view q. q may be the desired view ($v_q = v_d$) or any of the two reference views ($v_q = v_i$) used for color mapping in our implementation. Finally, we typically use values of $k = 6$.



Reference images.

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

Fig. 2. Desired view (c) has been created from 4 reference views. The optical center of the desired view is in the middle of the segment formed by the optical centers of cameras (a) and (b). Image (d) shows the depth of the desired image.

To sum up, it is important to note that only silhouette data is used for obtaining the structure of the desired view. Color information is used for color mapping. Not having to correlate color information in the reference views to obtain the structure makes the algorithm very fast. Finally, also note that chroma key or at least a very controlled background is often used to extract the silhouettes because IBVH is very sensitive to errors in the silhouettes. The problems derived from inaccurate silhouette detection are explained in the following section.

3. 2D TO 3D ERROR PROPAGATION

So far, we have described our particular implementation for obtaining the IBVH, assuming perfect silhouettes, similarly to what previous works do. However, it is important to observe that silhouettes are usually plagued with detection errors.

In the process of obtaining the IBVH, a false alarm in a reference view does not contribute to a false alarm in the desired view unless the visual cone that is erroneously created intersects simultaneously with other $C-1$ visual cones, where C is the total number of cameras. If the intersection is produced, then an invalid false alarm may appear in the desired view. Since the reconstructed shape is consistent because its projection in all the views matches with the silhouettes, then the 2D false alarm is undetectable (see Fig. 3 (a.1), (a.2)). However, the shape is not reconstructed when at least one of the erroneous visual cones does not intersect simultaneously with other $C - 1$ visual cones. This is the most usual case since the major part of the volume is unoccupied in most of the scenarios. In such cases, the cones produced by 2D false alarms do not intersect with visual cones from the rest of cameras (see Fig. 3 (b.1), (b.2)). Thus, 2D false alarms typically do not affect rendered views when obtaining the IBVH.

Contrarily, a miss in 2D ineluctably conduces to a miss in the 3D space and therefore a miss in the desired view (see

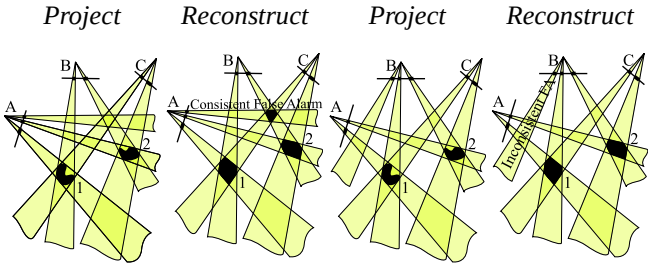


Fig. 3. In (a.1) there has been a false detection in camera A. The false visual cone intersects with other $C-1$ visual cones forming a false shape reconstruction (a.2). Another false alarm in camera B is depicted in (b.1). In this case, the false alarm forms an inconsistent cone for not intersecting with other $C-1$ visual cones (b.2). This type of false alarm, which is the most common case, does not affect IBVH rendering.

Fig. 4). In this case, the shape is not reconstructed in classic IBVH algorithms since an intersection of C cones is not produced.

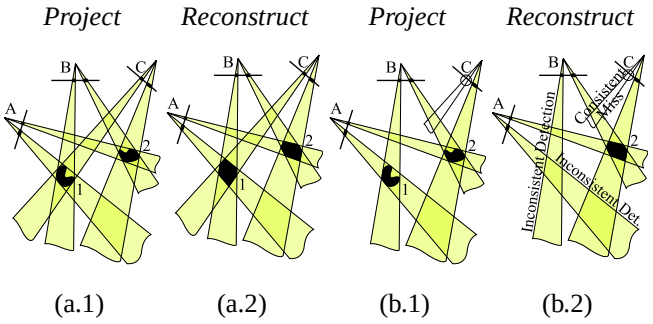


Fig. 4. In (a.1), objects 1 and 2 are correctly detected in all the cameras. In (b.1), object 1 has been missed in camera C. In (b.2), the Visual Hull is depicted. Note that the visual cones which do not intersect with any reconstructed shape are considered to be inconsistent with respect to the Visual Hull.

It seems clear that the very sensitive response of classical IBVH to 2D misses contradicts the general notion that "as the number of cones increases, the object is reconstructed with higher precision". In fact, an infinite number of silhouettes with a low but non null rate of randomly distributed misses will not reconstruct any shape.

In conclusion, VH reconstruction methods, in general, and IBVH algorithms, in particular, tend to penalize 2D misses in front of 2D false alarms. We propose taking another approach where the pixels in the desired view are assigned based on initial silhouettes and also on the probabilities of 2D false alarm and miss of the silhouette segmentation method.

4. SHAPE FROM INCONSISTENT SILHOUETTE

In Shape from Inconsistent Silhouette (SfIS), the VH is reconstructed assuming consistent silhouettes and corrected later with those parts of the volume which were not correctly classified. 3D misclassifications can be detected by examining the inconsistent regions of the silhouettes. To detect inconsistent regions, one can project back the VH and test whether the projections match with the reference silhouettes. Preliminary work on voxel-based SfIS was presented in [5].

We propose to extrapolate voxel-based SfIS to IBVH rendering. In a first stage, a preliminary desired view is built as described in section 2. At this stage, color mapping is still not required. In this first pass, a support image mask is created for each reference view as follows. Every time that a point in the viewing ray intersects the VH, the pixels in the support masks corresponding to the projections of the 3D point in the viewing ray are marked. Note that in a scenario with perfect silhouettes, the support masks and the reference masks should be equivalent at the end of this stage. The pixels rendered in the desired view are kept as valid. In the second stage of the algorithm, some erroneously missed pixels are recovered by testing the inconsistencies between support and reference masks. In this second pass, for each pixel in the desired view, a ray is traced again. However, in this second pass, both reference silhouettes and support masks are checked. For each 3D point in the viewing ray, a counter of inconsistencies $\mathcal{I}$ is set to 0. Each 3D point is projected to the reference images as usual. However, if a pixel in a reference silhouette is matched but its counterpart on the support mask is not, then $\mathcal{I}$ is increased. If $\mathcal{I}$ is higher than $T^\star$, then the desired pixel is rendered and the ray tracing process is continued for the following pixel.

Similarly to what we showed in [5], the main problem to solve is how to choose the minimum number of inconsistent intersections ($T^\star$) that have to be produced so that it can be determined that a pixel in the desired view was missed during the initial reconstruction process. The optimal threshold $T^\star$ has to be such that if $\mathcal{I} \geq T^\star$, then it is probabilistically better in terms of error reduction to render the derived pixel:

$$\begin{array}{rcl} \mathcal{I} \geq T^\star & \Rightarrow & \text{decide } \textit{to render} \\ \mathcal{I} < T^\star & \Rightarrow & \text{decide } \textit{not to render} \end{array} \quad (3)$$

In order to find $T^\star$, we have to express the probability of misclassification for any point in the viewing ray $P(Err_{3D}[T])$ so that $T^\star$ is that one which minimizes it:

$$T^\star = \underset{T}{\operatorname{argmin}}\, P(Err_{3D}[T]). \quad (4)$$

Since classification errors may be due to either false alarms or misses, the probability that a point in the viewing ray is misclassified is:

$$P(Err_{3D}) = P_B P(FA_{3D}) + P_S P(M_{3D}), \quad (5)$$

where P_S and P_B are prior probabilities[1] of a 3D point in the desired viewing ray forming part of the Visual Hull or not, respectively and where $P(FA_{3D})$ and $P(M_{3D})$ correspond to the probabilities of false alarm and miss in a 3D point. Both probabilities can be computed, as shown in [5], as a function of the threshold T and $\mathcal{O}$, corresponding to the number of reference views where the projection of a point in the viewing ray matches both the support mask and the reference mask. The first part of the function is:

$$P(FA_{3D}) = \sum_{i=max(T,1)}^{C-\mathcal{O}-1} \binom{C}{i} P(FA_{2D})^i (1 - P(FA_{2D}))^{C-i},$$

corresponding to the summation of all possible combinations that trigger a false alarm in a point of the viewing ray. And,

$$P(M_{3D}) = \sum_{i=max(C-\mathcal{O}-T+1,1)}^{C-\mathcal{O}-1} \binom{C}{i} P(M_{2D})^i (1 - P(M_{2D}))^{C-i}.$$

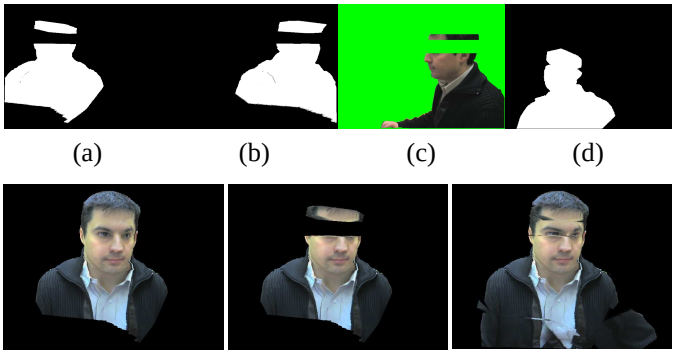
$P(M_{2D})$ and $P(FA_{2D})$ correspond to the probabilities that a reference pixel has been and has not been detected by error, respectively. These values depend on each foreground segmentation method. Our silhouette segmentation method is based on [6]. Finally, $T^\star$ in (4) can be obtained with basic full-search of the T that minimizes (5) or by employing a convex optimization strategy as we introduced in [7].

5. RESULTS AND FUTURE WORK

The algorithm has been evaluated with real-world calibrated images. In order to visually show how SfIS for IBVH operates, we present results where some errors have been intentionally introduced in one view (Fig. 5 (c)). Image (e) is presented to allow an easy visual comparison of a desired image computed with classical IBVH and perfect silhouettes.

Images (a), (b) and (d) show the support masks obtained at the end of the first stage of the algorithm employing the defective image (c). Note how the missed detections in (c) have propagated to the support masks. In (f), the results of classical IBVH is shown. The view shows how the misses in (c) have also been propagated to the desired view. In (g), we present the result of SfIS for IBVH. Note that most of the errors in the detection are recovered, even those misses corresponding to those parts of 3D object that was not visible in all the reference views (lower part of the body). However, there are still some color mapping errors due to the discontinuities that appear in the recovered parts of the desired view. Some of the future work we envision is improving the quality of the reconstruction by smoothing the depth map of the desired view.

[1] P_S and P_B can be set by computing the percentages of intersected/non-intersected 3D points in the viewing ray, for instance.



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

(e) Perfect references (f) Classical IBVH (g) SfIS for IBVH

Fig. 5. Results with perfect silhouettes and classical IBVH in (e). Results employing erroneous silhouettes and classical IBVH in (f) and with SfIS for IBVH in (g).

6. REFERENCES

[1] G. Slabaugh, R. Schafer, and M. Hans, "Image-based photo hulls," in *International Symposium on 3D Data Processing Visualization and Transmission*, Los Alamitos, CA, USA, 2002, vol. 0, p. 704.

[2] H. H. Baker, N. Bhatti, D. Tanguay, I. Sobel, D. Gelb, M. E. Goss, W. B. Culbertson, and T. Malzbender, "Understanding performance in coliseum, an immersive videoconferencing system," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 1, no. 2, pp. 190–210, 2005.

[3] G. Miller, A. Hilton, and J. Starck, "Interactive free-viewpoint video," in *European Conference on Visual Media Production*, November 2005.

[4] Wojciech Matusik, Chris Buehler, Ramesh Raskar, Steven J. Gortler, and Leonard McMillan, "Image-based visual hulls," in *Proceedings of International Conference and Exhibition on Computer Graphics and Interactive Techniques*, New York, NY, USA, 2000, pp. 369–374, ACM Press.

[5] J. L. Landabaso, M. Pardàs, and J.R. Casas, "Reconstruction of 3D shapes considering inconsistent 2D silhouettes," in *Proceedings of International Conference on Image Processing*, 2006.

[6] Chris Stauffer and W. Eric L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[7] J. L. Landabaso and M. Pardàs, "Shape from Inconsistent Silhouette," *submitted to Journal of Computer Vision and Image Understanding*, 2007.