City Research Online

City, University of London Institutional Repository

Citation: Slabaugh, G. G., Schafer, R. W. and Hans, M. C. (2002). Image-Based Photo Hulls. Paper presented at the First International Symposium on 3D Data Processing Visualization and Transmission, 19-06-2002 - 21-06-2002, Padova, Italy.

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: http://openaccess.city.ac.uk/5785/

Link to published version:

Copyright and reuse: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Image-Based Photo Hulls

Greg Slabaugh, Ron Schafer
Georgia Institute of Technology
Center for Signal and Image Processing
Atlanta, GA 30332
{slabaugh, rws}@ece.gatech.edu

Mat Hans
Hewlett-Packard Laboratories
Client & Media Systems Lab
Palo Alto, CA 94304
mat_hans@hp.com

Abstract

We present an efficient image-based rendering algorithm that computes photo hulls of a scene photographed from multiple viewpoints. Our algorithm, Image-Based Photo Hulls *(IBPH), like the* Image-Based Visual Hulls *(IBVH) algorithm from Matusik et. al. on which it is based, takes advantage of epipolar geometry to efficiently reconstruct the geometry and visibility of a scene. Our IBPH algorithm differs from IBVH in that it utilizes the color information of the images to identify scene geometry. These additional color constraints often result in more accurately reconstructed geometry, which projects to better synthesized virtual views of the scene. We demonstrate our algorithm running in a realtime 3D telepresence application using video data acquired from four viewpoints.*

1 Introduction

The task of generating a photo-realistic 3D representation of a visual scene is an important and challenging problem. Debevec et. al. [3] demonstrated in their Campanile movie that it is possible, using a user-assisted 3D modelling program and a handful of photos of a college campus, to produce a digital model of the scene that when rendered, yields images of stunning photorealism from novel viewpoints. Since this work, there has been much interest in producing results of similar quality using algorithms that are automatic and work on scenes composed of surfaces of arbitrary geometry.

Recently, researchers have become interested in reconstructing time-varying scenes [11, 10, 9, 15, 1]. Unfortunately, most standard approaches to the 3D scene reconstruction problem are too slow for realtime application on current computer hardware. When working with multi-view video data, such techniques perform the 3D reconstruction offline after the images have been acquired. Once the reconstruction is complete, it is rendered in realtime.

A notable exception is Matusik et. al's Image-Based Visual Hulls (IBVH) algorithm [8]. This algorithm is efficient enough to reconstruct and render views of the scene in realtime. By taking advantage of epipolar geometry, all of the steps of the algorithm function in the image space of the photographs (also called *reference views*) taken of the scene.

While the IBVH algorithm is exceptionally efficient, the visual hull geometry it reconstructs is not very accurate. Computed using silhouettes at each reference view, the visual hull [7] is a volume that represents the region of 3D space that projects to pixels in all silhouettes. The visual hull contains the true scene geometry. When photographed by only a few cameras, the scene's visual hull is often much larger than the true scene. When rendering new views, one can partially compensate for such geometric inaccuracies by view-dependent texture-mapping (VDTM), as done in the IBVH approach.

To obtain better geometric accuracy, more constraints are necessary. We propose the use of color. Our Image-Based Photo Hulls (IBPH) algorithm finds a set of 3D points that are photo-consistent [12, 6]. A point in space is said to be *photo-consistent* if (1) it does not project to background and (2) when visible, the light exiting the point (i.e. radiance) in the direction of each reference view is equal to the observed color in the photograph. The *photo hull* is then the spatially largest set of points in 3D space that project to photo-consistent colors in the reference images.

The photo hull is also a volume that contains the scene surfaces being reconstructed. However, it is a tighter fit to the true scene geometry than the visual hull. That is,

$$\text{True Scene} \subseteq \text{Photo Hull} \subseteq \text{Visual Hull}$$

New views synthesized using the more accurate geometry of the photo hull have improved photorealism. Like IBVH, all the steps of our algorithm function in image space (hence we call our algorithm *Image-Based* Photo Hulls). Our approach combines the efficiency of the IBVH algorithm with the improved geometric accuracy of the photo hull.

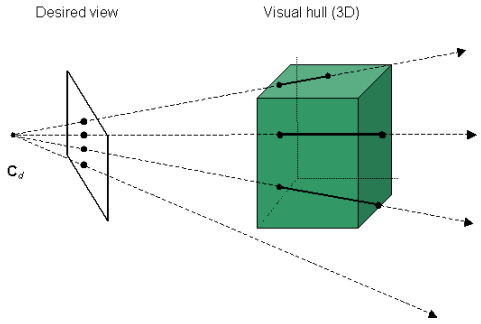

Figure 1. View-dependent geometry.

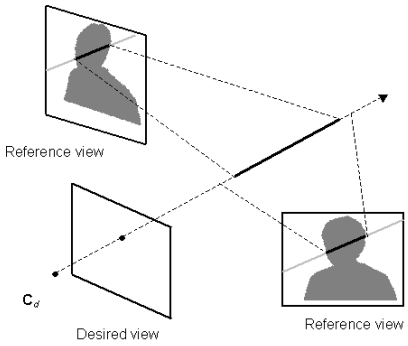

Figure 2. Computing a ray's visual hull intervals.

Like existing photo hull reconstruction algorithms, [12, 6, 2, 5] our IBPH approach starts with a surface larger than the scene, and then iteratively removes (carves) space using a photo-consistency measure until the visible points are photo-consistent. Our approach differs in that our algorithm functions in image space and produces a view-dependent reconstruction. Rather than reconstruct the full photo hull geometry, our method only reconstructs the portion of the photo hull that is visible to a virtual viewpoint.

For an overview of methods that reconstruct visual and photo hulls, please refer to [4, 13].

2 Image-Based Visual Hulls

In this section, we briefly review how the visual hull geometry is reconstructed by the IBVH algorithm. In the following section, we will show how we extend this algorithm to reconstruct photo hulls.

One of the unique properties of the IBVH algorithm is that the geometry it reconstructs is view-dependent. A user moves a virtual camera about the scene. For each virtual camera placement (also called *desired view*), the IBVH algorithm computes the extent that back-projected rays from the center of projection $\mathbf{C}_d$ intersect the visual hull in 3D space. This is shown in Figure 1. Thus, the reconstructed geometry changes as the user moves the virtual camera.

Consider an individual ray, as shown in Figure 2. The ray is back-projected from the desired view's center of projection, through a pixel in the image plane, and into 3D space. The ray projects to an epipolar line in each reference view. For each reference view, the IBVH algorithm determines the 2D intervals where the epipolar line crosses the silhouette. These 2D intervals are then "lifted" back onto the 3D ray and intersected for all reference views. The resultant set of intervals, called *visual hull intervals*, describe where the ray pierces the visual hull. In Figure 2, one visual hull interval is found along the 3D ray. This process is then repeated for all rays back-projected from the desired view.

3 Image-Based Photo Hulls

In this section we describe our new method of reconstructing photo hulls. More details are presented in the expanded version of this paper [14].

3.1 Approach

Our IBPH approach first computes the visual hull intervals using the IBVH algorithm, which quickly eliminates a large portion of 3D space that does not contain scene surfaces. Our algorithm then evaluates the photo-consistency of the closest point on the visual hull along each ray back-projected from the desired view. If the point is inconsistent, we take a small step along the ray, moving away from the desired view, as depicted in Figure 3. We continue stepping along an inconsistent ray until it either becomes consistent or we have stepped beyond all visual hull intervals along the ray. This latter case indicates that there is no photo-consistent geometry along the ray.

Note that in this approach, only the points on the hull that are visible in the desired view are processed. These points are the first points in the first visual hull interval along each back-projected ray.

3.2 Photo-Consistency

To determine the photo-consistency of a 3D point $\mathbf{P}$ on a ray, we project $\mathbf{P}$ into the ith reference view, yielding an image-space point $\mathbf{p}_i$. We only perform this projection for the reference views that have visibility of $\mathbf{P}$. Visibility is computed using the technique described in [8]. Around each $\mathbf{p}_i$ we collect a small neighborhood of pixels, N_i to use in our photo-consistency function.

There are many methods one can employ for matching color distributions to determine photo-consistency. In our

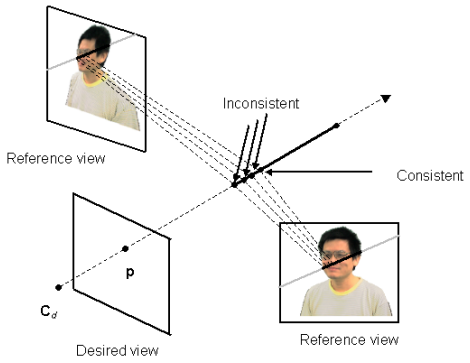

Figure 3. Computing the image-based photo hull.

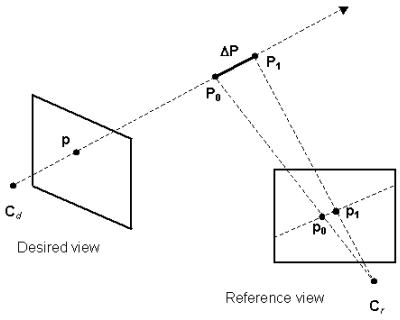

Figure 4. Stepping along an epipolar line.

approach, we assume the scene is Lambertian. Therefore, to be photo-consistent, a point must project to similar colors in each reference view. Our photo-consistency measure is

$$\text{Consistency} = \begin{cases} \text{True, if } \sigma \leq T_1 + \overline{\sigma} T_2 \\ \text{False, otherwise} \end{cases} \quad (1)$$

where σ is the standard deviation of all pixels, $\sum_i N_i$, and T_1 and T_2 are user-defined thresholds. The parameter $\overline{\sigma}$ is the mean standard deviation computed by averaging the standard deviation from each neighborhood of pixels N_i.

This consistency measure is spatially adaptive. The value of $\overline{\sigma}$ will be small when the 3D point projects to homogenous colors in each image, so the $\overline{\sigma} T_2$ will have little influence. If these colors are similar, the point will be declared consistent, for $\sigma < T_1$. If these colors are dissimilar, the point will be declared inconsistent, for $\sigma > T_1$. When the point projects to highly varying pixels in each image, the $\overline{\sigma}$ term will increase the maximum value of σ allowable for the point to be declared consistent. This allows for textured surfaces, as well as edges, to be correctly reconstructed. It also eases the Lambertian assumption.

3.3 Stepping Along Epipolar Lines

As we step in 3D along an inconsistent ray, looking for the point at which it becomes consistent, we must simultaneously step along an epipolar line in each reference view. The brute-force way of stepping along the epipolar line in the ith reference view is to simply project each 3D point $\mathbf{P}_j$ on the ray to the reference view point $\mathbf{p}_j$ by multiplying the reference view's projection matrix H_i with $\mathbf{P}_j$, i.e. $\mathbf{p}_j = H_i \mathbf{P}_j$. Such an approach will work, but will require a large number of matrix multiplications.

While the step size $|\Delta \mathbf{P}|$ in 3D is constant, the step size between adjacent points along the epipolar line in a 2D reference view varies due to the projection. However, since the projection is a homography (linear projective transformation), the step size *is* constant in homogeneous coordinates. We use this fact to produce a more efficient procedure for stepping along the epipolar line.

Consider the 3D point $\mathbf{P}_0$ on the ray, as shown in Figure 4. It projects to a point $\mathbf{p}_0 = H_i \mathbf{P}_0$ in a reference image. If we take a step along the ray, we arrive at a 3D point $\mathbf{P}_1 = \mathbf{P}_0 + \Delta \mathbf{P}$. The point $\mathbf{p}_1$, the projection of $\mathbf{P}_1$ into the reference view can be written as

$$\begin{aligned} \mathbf{p}_1 &= H_i \mathbf{P}_1 = H_i(\mathbf{P}_0 + \Delta \mathbf{P}) \\ &= \mathbf{p}_0 + H_i \Delta \mathbf{P} \end{aligned}$$

Thus, we can incrementally update the homogeneous position of the point along the epipolar line in the reference view. That is,

$$\mathbf{p}_j = \begin{cases} H_i \mathbf{P}_0, & j = 0 \\ \mathbf{p}_{j-1} + H_i \Delta \mathbf{P}, & j > 0 \end{cases} \quad (2)$$

We pre-compute $H_i \Delta \mathbf{P}$ for each ray and store it in a look-up table. As we step along the epipolar line, we use Equation 2 to compute the homogeneous position of the point $\mathbf{p}_j$. With this approach, stepping along an epipolar line is very efficient.

4 Results

We have implemented the IBPH algorithm on a multi-camera system. We have four calibrated and synchronized Sony DFW-V500 digital cameras. Each camera is connected to an 800 MHz HP Kayak XM600 machine, which performs background subtraction on the incoming frames, segmenting them into foreground and background regions. The resolution of the reference images is 320 x 240 pixels.

The segmented reference images are sent over a 100 Mb/s switch to our server machine, which computes the 3D reconstruction. Our server machine is a dual processor 2

GHz HP x4000 workstation. Our algorithm has been multi-threaded so that the kth thread reconstructs the scene using a set of images corresponding to time t_k.

Figure 5 compares the IBVH reconstruction to the IBPH reconstruction for a virtual viewpoint halfway between two reference views. The left two images show the visual hull reconstruction. At this viewpoint, the right side of the face is texture-mapped with one reference image, while the left side of the face is texture-mapped with another. Due to the geometric inaccuracy of the visual hull, there is a salient seam along the face where there is a transition between the two images used to texture-map the surface. The improved geometry of the photo hull corrects this problem, as shown in the right two images of the figure.

Figure 6 shows a view from a realtime 3D telepresence application we are currently developing with HP labs. The 3D model of the person's head and upper body is reconstructed online using the IBPH algorithm. The model of the person is then depth-composited with a 3D model of a conference room. New synthesized views of this composited scene are generated at 7.5 frames per second.

5 Conclusion

In this paper we have presented our Image-Based Photo Hulls algorithm that efficiently produces views of the photo hull.

There are some limitations to our approach. While the photo-consistency measure described in Section 3.2 has some tolerance for non-Lambertian scenes, it fails for scenes that have significant specularities. For fairness, we should note that visual hull reconstruction algorithms like IBVH do not have a problem with non-Lambertian scenes, as they do not perform color matching. For the data sets shown in this paper, the IBPH algorithm requires approximately five times the computation of the IBVH algorithm.

6 Future work

There are several future directions we are interested in pursuing. We currently do not take into consideration temporal coherence. Motion constraints could be imposed to improve efficiency and reconstruction quality. Also, an adaptive approach to determining the step size taken along each ray may improve the efficiency of our algorithm.

7 Acknowledgements

We thank Wojceich Matusik, Chris Buehler, and Leonard McMillan of MIT for sharing the IBVH source code and system configuration with us. We also thank Bruce Culbertson, Tom Malzbender, and Irwin Sobel of HP Labs for several useful discussions regarding the IBPH algorithm. Finally, we thank Dan Gelb for producing the virtual conference room model used in Figure 6. This work has been funded by HP labs.

References

[1] R. Carceroni and K. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid motion, shape, and reflectance. In *ICCV*, volume 2, pages 60–67, 2001.

[2] W. B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. In *ICCV Workshop, Vision Algorithms Theory and Practice*, pages 100–115. Springer-Verlag LNCS 1883, 1999.

[3] P. Debevec, C. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH*, pages 11–20, 1996.

[4] C. R. Dyer. Volumetric scene reconstruction from multiple views. In L. S. Davis, editor, *Foundations of Image Understanding*, pages 469–489. Kluwer, 2001.

[5] P. Eisert, E. Steinbach, and B. Girod. Multi-hypothesis, volumetric reconstruction of 3-d objects from multiple calibrated camera views. In *ICASSP*, volume 6, pages 3509–3512, 1999.

[6] K. Kutulakos and S. Seitz. A theory of shape by space carving. *Int. J. Computer Vision*, 38(3):199–218, 2000.

[7] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994.

[8] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *SIGGRAPH*, pages 369–374, 2000.

[9] S. Moezzi, L. C. Tai, and P. Gerard. Virtual view generation for 3d digital video. *IEEE Multimedia*, 4(1):18–26, 1997.

[10] P. Narayanan, P. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *ICCV*, pages 3–10, 1998.

[11] A. C. Prock and C. R. Dyer. Towards real-time voxel coloring. In *Image Understanding Workshop*, pages 315–321, 1998.

[12] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. *Int. J. Computer Vision*, 35(2):151–173, 1999.

[13] G. Slabaugh, W. B. Culbertson, T. Malzbender, and R. Schafer. A survey of volumetric scene reconstruction methods from photographs. In K. Mueller and A. Kaufman, editors, *Volume Graphics 2001, Proc. of Joint IEEE TCVG and Eurographics Workshop*, pages 81–100. Springer Computer Science, 2001.

[14] G. Slabaugh, R. Schafer, and M. Hans. Image-based photo hulls. Technical Report HPL-2002-28, Hewlett-Packard Labs, 2002.

[15] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6d. In *CVPR*, volume 2, pages 592–598, 2000.

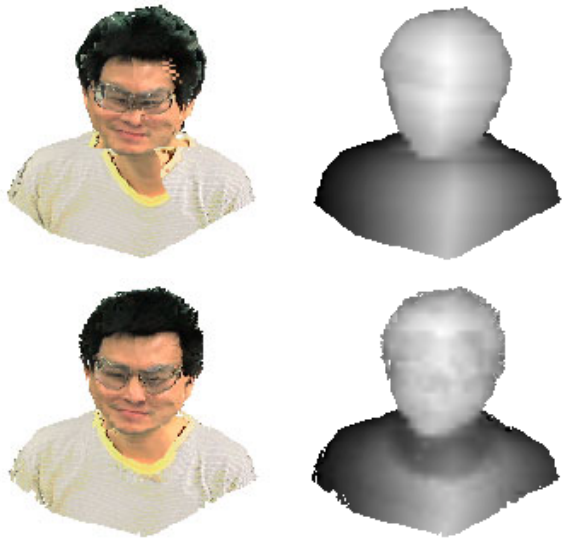

Figure 5. Visual hull reconstruction (upper row) vs. photo hull reconstruction (lower row). Left to right: synthesized view and depth map.

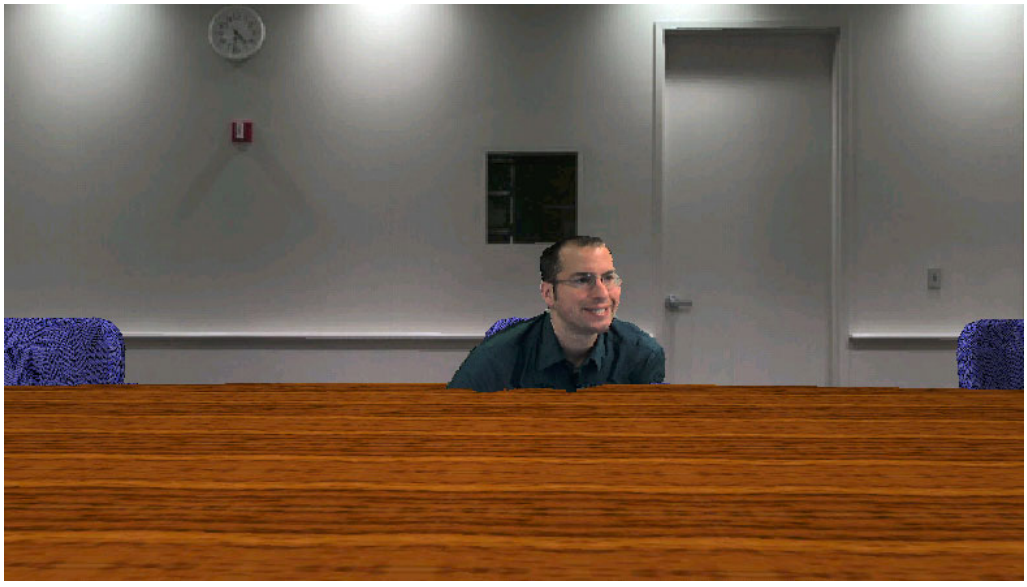

Figure 6. Using IBPH in a realtime 3D telepresence application.