Visual Hull Alignment and Refinement Across Time: A 3D Reconstruction Algorithm Combining Shape-From-Silhouette with Stereo

German K.M. Cheung Simon Baker Takeo Kanade
Robotics Institute, Carnegie Mellon University,
Pittsburgh PA 15213
german+@cs.cmu.edu simonb+@cs.cmu.edu tk+@cs.cmu.edu

Abstract

Visual Hull (VH) construction from silhouette images is a popular method of shape estimation. The method, also known as Shape-From-Silhouette (SFS), is used in many applications such as non-invasive 3D model acquisition, obstacle avoidance, and more recently human motion tracking and analysis. One of the limitations of SFS, however, is that the approximated shape can be very coarse when there are only a few cameras. In this paper, we propose an algorithm to improve the shape approximation by combining multiple silhouette images captured across time. The improvement is achieved by first estimating the rigid motion between the visual hulls formed at different time instants (visual hull alignment) and then combining them (visual hull refinement) to get a tighter bound on the object's shape. Our algorithm first constructs a representation of the VHs called the bounding edge representation. Utilizing a fundamental property of visual hulls which states that each bounding edge must touch the object at at least one point, we use multi-view stereo to extract points called Colored Surface Points (CSP) on the surface of the object. These CSPs are then used in a 3D image alignment algorithm to find the 6 DOF rigid motion between two visual hulls. Once the rigid motion across time is known, all of the silhouette images are treated as being captured at the same time instant and the shape of the object is refined. We validate our algorithm on both synthetic and real data and compare it with Space Carving.

1. Introduction

Three dimensional shape estimation from multiple cameras has long been an important and active research topic in computer vision. Among the algorithms that were proposed in the last two decades, the method of Visual Hull (VH) construction or Shape-From-Silhouette (SFS) approximates the shape of an object using silhouette images. Since its first introduction in [1], different variations, representations and applications of SFS have been proposed [14, 15, 10, 11, 2, 4, 9, 21], and it has become a standard and popular method of shape estimation. Estimating 3D shape using SFS has many advantages. Silhouettes are readily and easily obtainable and the implementation of SFS methods is generally straightforward. The visual hulls constructed using SFS provide an upper bound on the shape of the object. This inherently conservative property is particularly useful in applications such as obstacle avoidance and visibility analysis.

On the other hand, if there are only a few cameras, the visual hulls obtained using SFS can be a very coarse approximation to the shape of the actual object. This poses a big disadvantage for SFS in applications such as detailed shape acquisition and realistic re-rendering of objects.

Better shape estimates can be obtained using SFS if the number of distinct silhouette images is increased. This can be done in one of two ways: across space or across time. By across space, we mean increasing the number of cameras used. The across space approach, though simple and straightforward, may not be feasible in many practical situations due to financial (buying more cameras) or physical (system setup) limitations. For rigidly moving objects, an alternative way to increase the number of effective cameras is by combining silhouette information across time. In other words, if we estimate the rigid motion of the object between the time instants, we can combine the silhouette images at these time instants to get a refined shape of the object. We refer to the task of computing the rigid transformation as *visual hull alignment* and the task of combining the larger number of images as *visual hull refinement*.

It can be shown that using silhouette images alone to align two visual hulls is inherently ambiguous. Generally there are an infinite number of consistent rigid motion/object shape pairs which produce the same sets of silhouette images at two different time instants [5]. In order to get an unambiguous alignment between two visual hulls, additional information besides the silhouette images is needed. In this paper, we show how color consistency can be used to break the alignment ambiguity.

The remainder of this paper is organized as follows. The problem scenario is introduced in Section 2. In Section 3 a new visual hull representation called the bounding edge representation is proposed. We then use a fundamental property of the bounding edge representation to find a number of 3D points on the surface of the object. These 3D surface points are used to align two visual hulls using a 3D image alignment algorithm. After the alignment process, the shape of the object is refined by treating all the silhouette images as being captured at a single time instant. In Section 4 the problem of determining visibility is addressed while both synthetic and real results are presented in Section 5. We conclude in Section 6 with a summary and discussion.

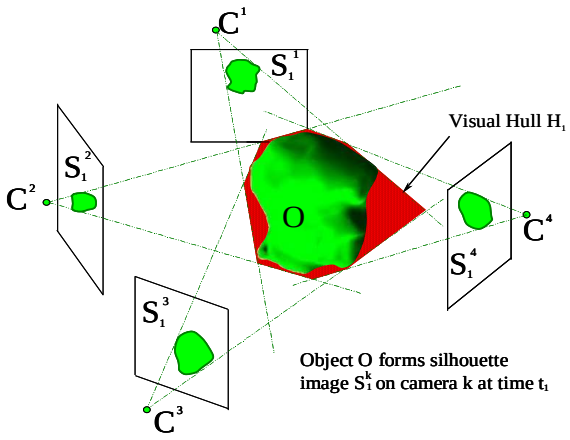

Figure 1. An example of classical Shape-From-Silhouette with a head-shaped object and four cameras at time t_1.

2. Problem Scenario

Suppose there are K fixed cameras positioned around a rigid Lambertian 3D object $\boldsymbol{O}$. Let $\{I_j^k; \quad S_j^k; \quad k = 1, \cdots\cdots, K\}$ be the set of color and corresponding silhouette images of the object $\boldsymbol{O}$ obtained from the K cameras at time t_j. It is assumed that the cameras are color balanced and calibrated with $\Pi^k() : I\!R^3 \rightarrow I\!R^2$ and C^k being the perspective projection function and the center of camera k respectively. Using the silhouette images $\{S_j^k\}$ and the camera calibration functions $\{\Pi^k\}$, an upper bound of the object's shape called the visual hull H_j at time t_j can be reconstructed by either visual cone intersection [10] or voxel-based [15] methods. The situation at time t_1 with four cameras is shown in Figure 1. Now suppose the object is moving arbitrarily but rigidly over time and we are given J sets of color and silhouette images taken at time $t_1, \cdots\cdots, t_J$. Without loss of generality, we assume the orientation and the position of the object at time t_1 is $(\boldsymbol{I}, \underline{0})$ and that at time t_j it is $(\boldsymbol{R}_j, \boldsymbol{t}_j); \quad j = 2, \cdots\cdots, J$. The problem is to find the motion of the object $(\boldsymbol{R}_j, \boldsymbol{t}_j)$ over time and a refined shape of the object by combining the $K \times J$ silhouette images.

3. Visual Hull Alignment and Refinement

Although SFS is a popular method of shape reconstruction at single time instant, little work has been done in extending it across time. The work most related to this paper is by Cipolla and Wong [20, 21]. They study the problem of estimating the structure and motion of an object undergoing *circular motion* from silhouette profiles. They assume a single camera which is weakly calibrated (i.e. with known intrinsic but unknown extrinsic parameters). Either the camera (on a robotic arm) or the object (on a turntable) performs unknown circular motion while the silhouette images are taken. They identify and estimate the *frontier points* on the silhouette boundary and use them to estimate the circular motion (axis of rotation) between images. Once the motion is estimated, the object shape is reconstructed using a voxel-based SFS method. Ponce et al. also study the problem of recovering the motion and shape of a *smooth curved* object from silhouette images in [7, 19]. They define a local parabolic structure on the surface of the object and use epipolar geometry to locate corresponding frontier points on three silhouette images. Motion between images is then estimated by a two-step nonlinear minimization.

It is shown in [5] that the problem of visual hull alignment using silhouette images alone is inherently ambiguous. The motion ambiguity is closely related to the indeterminacy in shape. To break this ambiguity, we now describe how to incorporate color information into the traditional SFS formulation, thereby combining SFS and stereo.

3.1. The Bounding Edge Visual Hull Representation

Consider the set of K silhouette images $\{S_j^k\}$ at a given time instant t_j. We propose the following representation of the visual hull H_j which we call the bounding edge representation.

Definition of Bounding Edge
Let u_j^i be a point on the boundary of the silhouette image S_j^k. By projecting u_j^i into 3D space through the camera center C^k, we get a ray r_j^i. A bounding edge E_j^i is defined to be the part of r_j^i such that the projection of E_j^i onto the l^{th} image plane lies completely inside the silhouette S_j^l for all $l \in \{1, \cdots\cdots, K\}$. Mathematically the condition is expressed as

$$E_j^i \subset r_j^i \text{ and } \Pi^l(E_j^i) \subset S_j^l \ \forall \ l \in \{1, \cdots\cdots, K\}. \quad (1)$$

An example illustrating the definition of a bounding edge at t_1 is shown in Figure 2. A bounding edge can be computed by first projecting the ray r_j^i onto the $K-1$ silhouette images $S_j^l, \quad l = 1, \cdots\cdots, K; \quad l \neq k$, and then re-projecting the segments which overlap with S_j^l back into 3D space. The bounding edge is the intersection of the reprojected segments. Note that the bounding edge E_j^i is not necessarily a continuous line. It may consist of several segments if any of the silhouette images are not convex. Hereafter, a bounding edge E_j^i is denoted by a set of *ordered* 3D vertex pairs as:

$$E_j^i = \left\{ \left(SV_j^i(m), FV_j^i(m)\right); \quad m = 1, \ldots, M_j^i \right\}, \quad (2)$$

where $SV_j^i(m)$ and $FV_j^i(m)$ represent respectively the start vertex and finish vertex of the m^{th} segment of the bounding edge and M_j^i is the number of segments that E_j^i is comprised of. By sampling points on the boundaries of all the silhouette images $\{S_j^k; \quad k = 1, \cdots\cdots, K\}$, we can construct a list of L_j bounding edges that represents the visual hull H_j.

There are two main advantages of using bounding edges to represent the visual hull. Firstly, they lie exactly on the surface of visual hull and therefore are not approximations like discrete voxel representations [8]. Secondly the essence

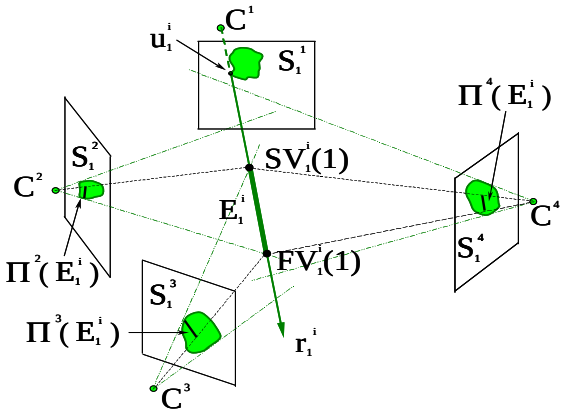

Figure 2. The bounding edge E_1^i is obtained by first projecting the ray r_1^i onto S_1^2, S_1^3, S_1^4 and then re-projecting the segments overlapped with the silhouettes back into the 3D space. E_1^i is the intersection of the reprojected segments.

of Shape-From-Silhouette is naturally embedded in the definition of bounding edges, as indicated by the second fundamental property of visual hulls stated below.

Fundamental Properties of Visual Hulls (FPVHs)
1st FPVH: *The object that created the silhouette lies completely inside the visual hull.*
2nd FPVH: *Each bounding edge of the visual hull touches the object (that formed the silhouette images) at at least one point.*

The 2nd FPVH allows us to use bounding edges to store and represent the key shape information of the object that can be obtained from the set of silhouette images. In the next section, we will combine the 2nd FPVH and color stereo on the bounding edges to extract 3D points on the surface of the objects. One negative aspect of the bounding edge representation is that it is incomplete. Since the boundary of the silhouette is sampled at a finite collection of points, the surface of the visual hull is not represented entirely by the bounding edges. By increasing the number of samples on the boundary, the surface can be represented more completely.

In [4, 13, 3], Matusik et al. proposed algorithms to build and render visual hulls in real-time. Their way of intersecting viewing rays for VH rendering [13] and intersecting visual cones for VH construction [3] are similar to the way our bounding edges are constructed. However, there are two fundamental differences between their algorithms and the definition of bounding edge. Firstly, our bounding edges are originated only from points on the silhouette boundary while in [13], the viewing rays can originate from any point inside the silhouette. Also the VH constructed in [3] is 2D surface-based while our bounding edge is 1D line-based. Secondly, both their viewing rays and the surface-based VH do not embed the important 2^{nd} FPVH as bounding edges do. On the other hand, Lazebnik et al. proposed a new way of representing visual hulls in [12]. The definition of bounding edge in our paper is theoretically equivalent to the edge of the visual hull mesh defined in their work, although they derive the edges from locating frontier and triple points

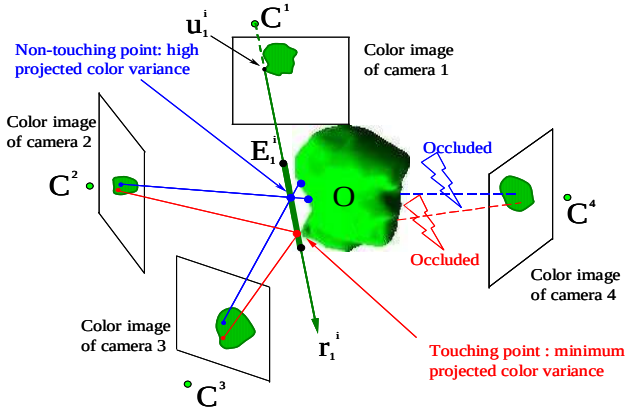

Figure 3. Locating the touching point (colored surface point) by searching along the bounding edge for the point with the minimum projected color variance.

while we construct the edges directly from the silhouette images.

3.2. Locating 3D Colored Surface Points

The 2nd FPVH states that each bounding edge touches the object at at least one point. How are we going to to locate this touching point? Here we use information from the color images $\{I_j^k\}$ to help find the touching point. Since the object is assumed to be Lambertian and all the cameras are color balanced, any point on the surface of the object will have the same projected color in all of the color images. In other words, for any point on the surface of the object, its projected color variance across the *visible* cameras should be zero. Hence on a bounding edge, the point with zero projected color variance is the point where the edge touches the object. Hereafter we call these touching points the *colored surface points* of the object.

To express the idea mathematically, consider a bounding edge E_j^i from the j^{th} visual hull. We parameterize a point $W_j^i(m,w)$ on E_j^i by two parameters m and w, where $m \in \{1, \cdots\cdots, M_j^i\}$ and $0 \leq w \leq 1$ with

$$W_j^i(m,w) = SV_j^i(m) + w * \left(FV_j^i(m) - SV_j^i(m)\right) \ . \quad (3)$$

Let $\boldsymbol{c}_j^k(P)$ be the function which returns the projected color of a 3D point P on the k^{th} color image at time t_j. The projected color mean $\mu_j^i(m,w)$ and variance $\sigma_j^i(m,w)$ of the point $W_j^i(m,w)$ are given as

$$\begin{aligned} \mu_j^i(m,w) &= \frac{1}{n_j^i}\sum_k \boldsymbol{c}_j^k(W_j^i(m,w)); \\ \sigma_j^i(m,w) &= \frac{1}{n_j^i}\sum_k [\boldsymbol{c}_j^k(W_j^i(m,w)) - \mu_j^i(m,w)]^2 \ . \end{aligned} \quad (4)$$

The projected color $\boldsymbol{c}_j^k(W_j^i(m,w))$ from camera k is used in calculating the mean and variance *only if* $W_j^i(m,w)$ is *visible* in that camera and n_j^i denotes the number of the visible cameras for point W_j^i. We will discuss in Section 4 how

to conservatively determine the visibility of a 3D point w.r.t a camera using only the silhouette images. Figure 3 illustrates the idea of locating the touching point by searching along the bounding edge.

In practice, due to noise and inaccuracies in color balancing, instead of searching for the point which has a zero projected color variance, we locate the point with the minimum variance. In other words, we set the colored surface point of the object on E_j^i to be $W_j^i(\tilde{m}, \tilde{w})$ where $\tilde{m}$ and $\tilde{w}$ minimizes $\sigma_j^i(m, w)$ for $0 \leq w \leq 1; \quad m \in \{1, \cdots, \cdots, M_j^i\}$. Note that by choosing the point with the minimum variance, the problem of tweaking parameters or thresholds of any kind is also avoided. The need to adjust parameters or thresholds is always a problem in other shape reconstruction methods such as space carving [9] or stereo. Space carving relies heavily on a color variance threshold to remove non-object voxels and stereo matching results are sensitive to the search window size. In our case, knowing that each bounding edge touches the object at at least one point (2nd FPVH) is the key piece of information that allows us to avoid any thresholds. In fact locating CSPs is a special case of the problem of matching points on pairs of epipolar lines as discussed in [17, 6]. In [17] and [6], points are matched on "general" epipolar lines on which there may or may not be a matching point so threshold and an independent decision is needed for each point. To locate CSPs, points are matched on "special" epipolar lines which guarantee to have at least one one matching point so no threshold is required. Note that there is *no* point-to-point correspondence relationship between two different sets of CSPs obtained at different time instant. The only property common to the CSPs is that they all lie on the surface of the object. For simplicity, we denote $W_j^i(\tilde{m}, \tilde{w}), \sigma_j^i(\tilde{m}, \tilde{w})$ and $\mu_j^i(\tilde{m}, \tilde{w})$ by $\tilde{W}_j^i, \tilde{\sigma}_j^i$ and $\tilde{\mu}_j^i$ respectively.

3.3. Alignment Using Colored Surface Points

Suppose we have located two sets of colored surface points at two different time instants t_1 and t_2. To align the visual hulls H_1 and H_2, we use an idea similar to the 2D image alignment problem as in [16]. In our case, instead of aligning a 2D image with another 2D image, we align 2D images ($\{I_2^k\}$) at time t_2 with a "3D image" (the colored surface points $\{\tilde{W}_1^i\}$) at time t_1 through the projection functions $\{\Pi^k\}$. The error measure used is the sum of color differences between the colored surface points at time t_1 and their projected colors from the color images at time t_2 and vice versa. Mathematically, let $\{I_j^k, S_j^k, \tilde{W}_j^i, \tilde{\mu}_j^i, \tilde{\sigma}_j^i; i = 1, \cdots\cdots, L_j; \quad k = 1, \cdots\cdots, K; \quad j = 1, 2\}$ be the two sets of data. To find the most color consistent alignment $(\boldsymbol{R}, \boldsymbol{t})$, consider the color error function

$$e = \sum_{i=1}^{L_2} e_{1,2}^i + \sum_{i=1}^{L_1} e_{2,1}^i \,, \tag{5}$$

$$e_{1,2}^i = \sum_k e_{1,2}^{i,k} = \sum_k [\boldsymbol{c}_1^k(\boldsymbol{R}^T \tilde{W}_2^i - \boldsymbol{R}^T \boldsymbol{t}) - \tilde{\mu}_2^i]^2 \,, \tag{6}$$

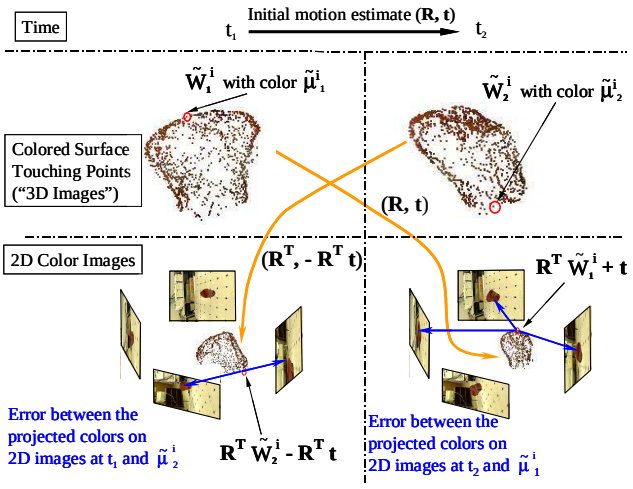

Figure 4. Visual alignment using color consistency. The error between the colors of the 3D surface points and their projected image colors is minimized.

$$e_{2,1}^i = \sum_k e_{2,1}^{i,k} = \sum_k [\boldsymbol{c}_2^k(\boldsymbol{R}\tilde{W}_1^i + \boldsymbol{t}) - \tilde{\mu}_1^i]^2 \,, \tag{7}$$

where $e_{2,1}^{i,k}$ represents the difference between the mean color $\tilde{\mu}_1^i$ of the colored surface point $\tilde{W}_1^i$ at time t_1 and its projected color $\boldsymbol{c}_2^k(\boldsymbol{R}\tilde{W}_1^i + \boldsymbol{t})$ in camera k at time t_2. Note that at time t_2, the new position of $\tilde{W}_1^i$ is $\boldsymbol{R}\tilde{W}_1^i + \boldsymbol{t}$ due to the motion of the object. Likewise, $e_{1,2}^{i,k}$ is the difference between the mean color $\tilde{\mu}_2^i$ of $\tilde{W}_2^i$ and its projected color $\boldsymbol{c}_1^k(\boldsymbol{R}^T\tilde{W}_2^i - \boldsymbol{R}^T\boldsymbol{t})$ on camera k at time t_1. Just as for calculating the color consistency of points on the bounding edge, the summations in equations (6) and (7) include the projected color of camera k *only if* the point of interest is visible in that camera. The minimization of Eq. (5) can be solved by Gradient Descent [18] or Levenberg-Marquardt algorithm as discussed in [16]. The process of visual hull alignment by color consistency is illustrated in Figure 4.

3.4. Visual Hull Refinement

After estimating the alignment across time, the rigid motion $\{(\boldsymbol{R}_j, \boldsymbol{t}_j)\}$ is used to combine the J sets of silhouette images $\{S_j^k; \quad k = 1, \cdots\cdots, K; \quad j = 1, \cdots\cdots, J\}$ to get a tighter upper bound of the shape of the object. By fixing t_1 as the reference time, we combine $\{S_j^k\}; \; j = 2, \cdots\cdots J$ with $\{S_1^k\}$ by considering the former as "new" silhouette images captured by additional cameras placed at positions and orientations transformed by $(\boldsymbol{R}_j, \boldsymbol{t}_j)$. In other words, for the silhouette image S_j^k captured by camera k at time j, we use a new perspective projection function $\Pi_{j\to 1}^k$ derived from Π^k through the rigid transformation $(\boldsymbol{R}_j, \boldsymbol{t}_j)$. As a result, the effective number of cameras is increased from K to $K \times J$.

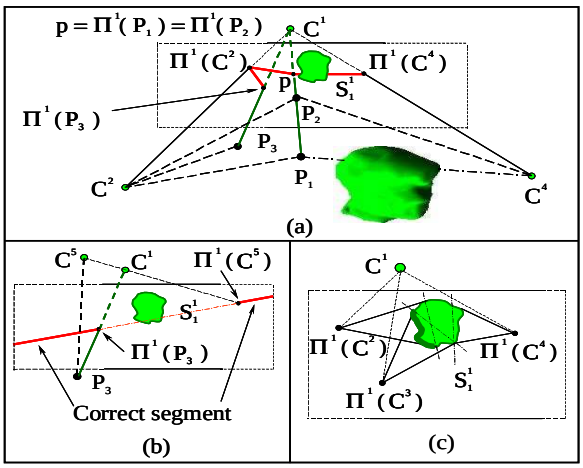

Figure 5. (a) Visibility of points with respect to cameras using Lemma 1. (b) An example where C^5 is behind C^1. The correct line to be used in Lemma 1 is the outer segment which passes through infi nity instead of the direct segment. (c) Boundary points that can be used to construct bounding edges are marked by the thick boundary. These boundary points are the ones which the resulting bounding edges can be seen by at least two other cameras besides camera 1.

4. Determining Visibility

To locate the colored surface points using Eq. (4), the visibility of the 3D point $W_j^i(m,w)$ with respect to all K cameras is required. Here, we present a way to determine the visibilities *conservatively* using only the silhouette images. Suppose we are given a 3D point P and a set of silhouette images $\{S_j^k\}$ with camera centers $\{C^k\}$ and projection functions $\{\Pi^k()\}$. The following lemma then holds:

Lemma 1 : Let $\Pi^l(P)$ and $\Pi^l(C^k)$ be the projections of the point P and the k^{th} camera center C^k on the (infinite) image plane of camera l. If the 2D line segment joining $\Pi^l(P)$ and $\Pi^l(C^k)$ does not intersect the silhouette image S_j^l, *then* P is visible with respect to camera k at time t_j.

Figure 5(a) gives examples where the points P_1, P_2 and P_3 are visible with respect to camera 2. The converse of Lemma 1 is *not* necessarily true : the visibility *cannot* be determined if the segment joining $\Pi^l(P)$ and $\Pi^l(C^k)$ intersects the silhouette S_j^l. One counter example is shown in Figure 5(a). Both points P_1 and P_2 project to the same 2D point p on the image plane of camera 1 and the segment joining p and $\Pi^1(C^4)$ intersects with S_1^1. However, P_1 and P_2 have different visibilities with respect to camera 4 (P_2 is visible while P_1 is not). Note that special attention must be given to situations in which camera center C^k lies behind camera center C^l. In such cases, the correct line segment to be used in Lemma 1 is the outer line segment (passing through infinity) joining $\Pi^l(P)$ and $\Pi^l(C^k)$ rather than the direct segment. An example is given in Figure 5(b).

Though conservative, there are three advantages of using Lemma 1 to determine visibility in our alignment algorithm. First of all, Lemma 1 uses information directly from the silhouette images, avoiding the need to estimate the shape of the object for the visibility test. Secondly, recall that to construct a bounding edge E_j^i, we start with the boundary point u_j^i of the k^{th} silhouette. Hence all the points on E_j^i project to the same 2D point u_j^i on camera k which implies all points on the bounding edge E_j^i have the same set of conservative visible images. This property ensures the color consistencies of points on the same bounding edge are calculated from the same set of images. Accuracy in searching the optimal point $\tilde{W}_j^i$ is increased because comparisons are made fairly among points on the same bounding edge. Finally Lemma 1 also provides a guideline to sample the silhouette boundary points for constructing bounding edges. To have meaningful color consistencies, the number of color images used in (4) has to be at least 2 (or otherwise the projected color variances will always be 0). By Lemma 1, boundary points u_j^i are chosen such that the resulting E_j^i is seen by at least 2 other images (excluding the image S_j^k from which the boundary point is chosen from). An example is shown in Figure 5(c). Only points on part of the boundary of S_1^1 (marked by thicker lines) are used to construct bounding edges because they are the points from which the resulting bounding edges can be seen by at least two other cameras (cameras 2 and 3).

5. Experimental Results

5.1. Synthetic Data Set : Torso Sequence

A synthetic data set is created to demonstrate the validity of our alignment algorithm. A textured wire-frame computer model resembling the human torso was used. The model was moved under a known trajectory for twenty two frames. At each time instant, images of six cameras ($K = 6$) with known camera parameters are rendered using OpenGL. A total of 22 sets of color and silhouette images are generated. Example input images can be found in the movie clip torso.mpg. All the movie sequences mentioned in this paper can be found at http://www.cs.cmu.edu/~german/research/CVPR2003/VisualHull.

5.1.1. Alignment

Three alignment algorithms were implemented to compare the effectiveness of using bounding edges/colored surface points to align visual hulls as compared to using voxel models created by Shape-From-Silhouette (SFS) and Space Carving (SC) [9]. In algorithm I, bounding edges and colored surface points are extracted and used to find the alignment between frames as described in Section 3.3. Two of the six estimated motion parameters for this algorithm are plotted in Figure 6 as red dashed lines with asterisks. They are very close to the ground truth values represented by the black solid lines. In algorithm II, a voxel model is built from the silhouette images using voxel-based Shape-From-Silhouette (SFS). Surface voxels are extracted and colored by back-projecting onto the color images. The centers of the

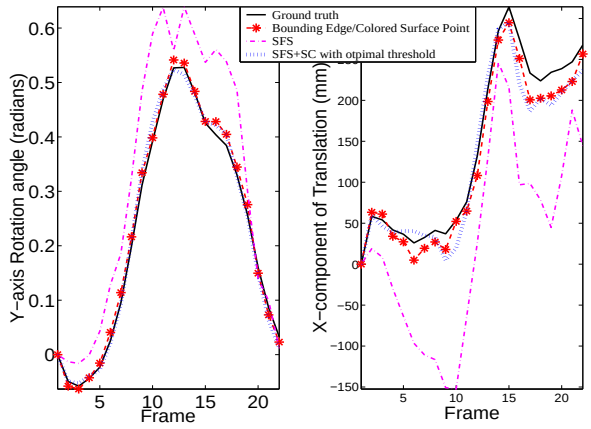

Figure 6. Results of two motion parameters (Y-axis rotation angle and X-component of translation) estimated over time from alignment experiments using different input data : using bounding edges/colored surface points (red dashed lines with asterisks), using SFS voxel models (magenta dotted-dashed lines), using SFS+SC voxel models with optimal threshold (blue thick dotted lines) and ground-truth values (solid black lines). The results of using bounding edges/colored surface points are better than the other two.

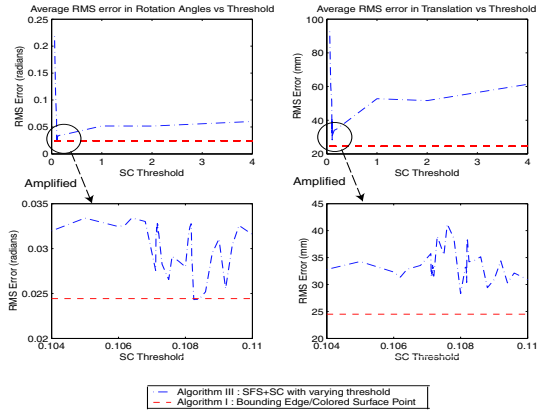

Figure 7. Graphs of average RMS errors in rotation and translation against the threshold used in SC in algorithm III. The bottom half of the fi gure illustrates the amplifi ed part of the graph near the optimal threshold value (0.108). Using bounding edges is always more accurate than using SC in alignment, even optimal threshold is used for SC.

colored surface voxels are treated as input data points to the same alignment algorithm used in algorithm I. The results of algorithm II are plotted as magenta dotted-dashed lines in Figure 6. As can be seen, alignment using the SFS voxel model is much less accurate than using bounding edges.

In algorithm III, a voxel model is first built by using SFS (as in algorithm II) and further refined by Space Carving (SC). The centers of the surface voxels (which are already colored by SC) are used for alignment. To study the effect of the SC threshold on alignment, different values of the threshold are used and the estimated motion parameters are compared with the ground truth values. Graphs of the average RMS errors in the rotation and translation parameters against the threshold used are shown as blue dotted-dashed lines in Figure 7. When the threshold is too small, many correct voxels are carved away, resulting in a voxel model much smaller than the actual object. When the threshold is too big, extra incorrect voxels are not carved away, leaving a voxel model bigger than the actual object. In both cases, the wrong data points extracted from the incorrect voxel models cause errors in the alignment process. The optimal threshold value is found to be around 0.108 and the graph is amplified in the vicinity of this value in the bottom part of Figure 7. As a comparison, the average RMS errors for rotation and translation parameters obtained from algorithm I is drawn as the horizontal red dashed line. With the optimal SC threshold, the performance of using SFS+SC voxel models is comparable but less accurate than that of using bounding edges. The estimates of the motion parameters using SC with the optimal threshold (the thick blue dotted lines) are also included in Figure 6. SC with the optimal threshold performs well but not as good as bounding edges. Table 1 gives a rough comparison of the computational time needed for each step of all the experiments. The timing is obtained on a 500MHz Pentium III CPU.

Table 1. The approximate time for each step in the alignment experiments. BE is about the same as SFS and faster than SFS+SC.

Step	**Time required per frame**
Extracting bounding edge (BE)	0.92 s
Locating surface colored points (100 points searched on each BE)	0.16s
SFS with 128^3 voxels	1.08s
SFS + SC with 128^3 voxels and optimal threshold	4.74s
Alignment	16.2s

5.1.2. Refinement

The estimated parameters in the alignment experiments are used to refine the shape of the torso model, using the voxel-based SFS method [15]. The visual hull at time t_j is constructed by using the silhouettes at t_j and all those from the previous frames $\{t_1, \cdots\cdots, t_{j-1}\}$, transformed by the estimated motion parameters as described in Section 3.4. To quantify the refinement results, the ground-truth wire-frame model used to render the input images is converted into a ground-truth voxel model and compared to the refined voxel models. The results are plotted in Figure 8 with graphs (a) and (b) showing respectively the number of extra and missing voxels between the refined shapes and the ground-truth voxel models against the number of frames used. Figure 8(c) illustrates the ratio of total incorrect (missing plus extra) to total voxels.

In all the experiments, the number of extra voxels decreases as the number of frames used increases because a tighter visual hull is obtained with an increase in the number of distinct silhouette images. However, the number of missing voxels also increases as the number of frames used increases. This is due to alignment errors which remove correct voxels during construction. The number of missing

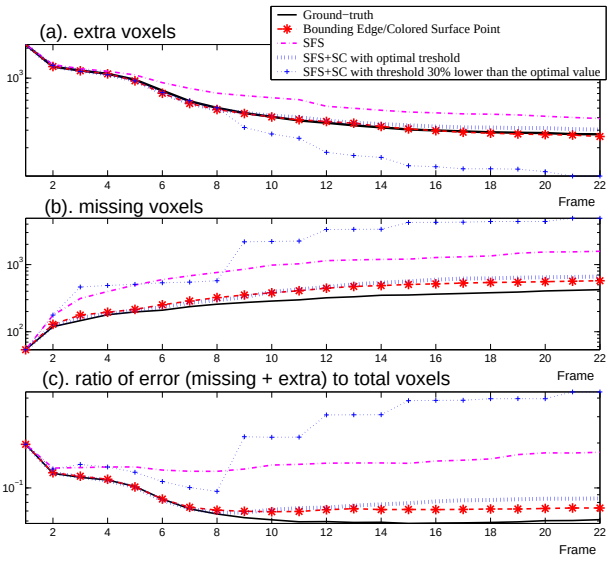

Figure 8. Graphs of refinement errors (missing and extra voxels) across time (frames). Using bounding edges has lower error ratio than using either SFS or SFS+SC.

voxels is very large if the alignments are way off (e.g. the magenta dotted-dashed curve of results from algorithm II or the blue dotted curves (with '+' markers) of results from algorithm III with threshold 30% lower than the optimal value). The refinement result is the best using the motion parameters estimated using bounding edges (the red dashed lines with asterisks in Figure 8). The video clip torso.mpg shows one of the six input image sequences (camera 4), the unaligned and aligned colored surface points and the temporal refinement/alignment results using bounding edges.

5.2. Real Data Sets

Pooh Sequence:
The first real test object is a toy (Pooh) and six calibrated cameras ($K = 6$) are used. The toy is placed on a table and moved to new but unknown positions and orientations manually in each frame. A total of fifteen frames are captured. The input image of camera 1 at time t_1 is shown in Figure 9(a). The extracted bounding edges with the corresponding colored surface points at time t_1 are shown in Figures 9(b) and (c). Figures 9(d) and (e) show respectively the unaligned and aligned colored surface points from all fifteen frames. Refinement is done using the voxel-based SFS method. Figures 9(f),(g) and (h) illustrate the refinement results at three time instants t_1, t_6 and t_{15}. The improvement in shape is very significant from t_1 when 6 silhouette images are used to t_{15} when 90 silhouette images are used. Note that for shape refinement, Space Carving (SC) can also be used. Figures 9(i)(j) show the refinement results using SFS + SC at t_1 (6 images) and t_{15} (90 images). Generally with a *good threshold*, refinement using SFS + SC is better than SFS for the same number of images. The video clip pooh.mpg shows one of the six input image sequences (camera 4), the unaligned/aligned colored surface points and the temporal refinement/alignment results.

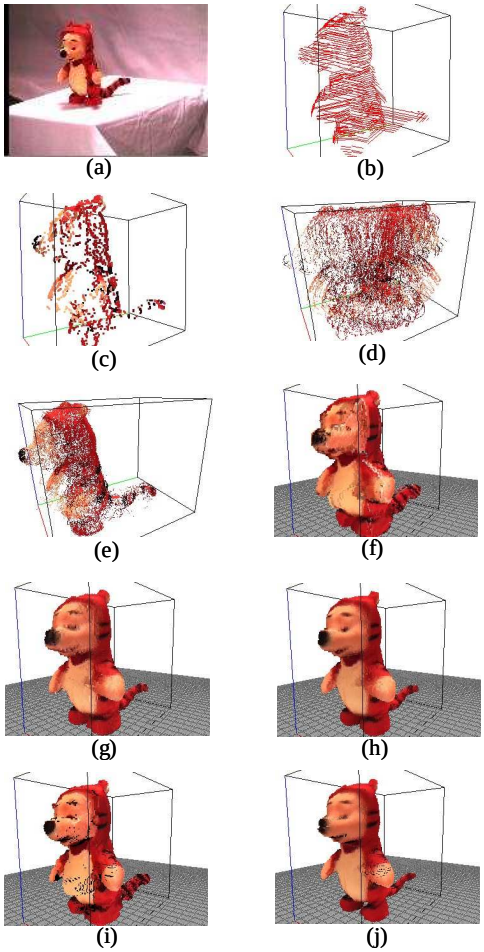

Figure 9. Pooh Data Set. (a) Example input image, (b) Bounding edges at t_1, (c) Colored surface points at t_1, (d) Unaligned colored surface points from all frames, (e) Aligned colored surface points of all frames, (f) SFS model at t_1 (6 images used), (g) SFS refined shape at t_6 (36 images used), (h) SFS refined shape at t_{15} (90 images used), (i) SFS + SC model at t_1, (j) SFS + SC refined model at t_{15}.

Dinosaur-Banana Sequence:
A second real data set of a toy dinosaur on top of a bunch of bananas is also captured with six cameras. The dinosaur and the bananas are placed on a turntable with unknown rotation axis and rotation speed. Fifteen frames are captured and the alignment and refinement results are shown in Figure 10 and in the movie clip dino-bana.mpg.

6. Summary and Discussion

In this paper we have proposed an algorithm to perform Shape-From-Silhouette (SFS) across time for a rigid object undergoing arbitrary rigid motion. At each time instant bounding edges are constructed from the silhouette images and colored surface points are located on the bounding edges by comparing color consistencies. The colored surface points are used to estimate the rigid motion of the

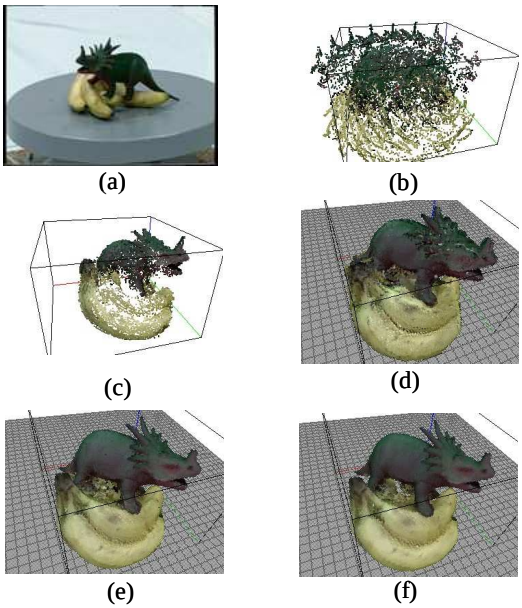

Figure 10. Dinosaur-Banana Sequence. (a) Example input image, (b) Unaligned colored surface points from all frames, (c) Aligned colored surface points, (d) SFS model at t_1 (6 images used), (e) SFS refined shape at t_6 (36 images used), (f) SFS refined shape at t_{15} (90 images used).

object across time, using a 2D images/3D points alignment algorithm. Once the alignment is known, all of the images are considered as being captured at the same instant. The refined shape of the object can then be obtained by any reconstruction method such as SFS or Space Carving.

Our algorithm combines the best advantages of both SFS and Stereo. A key principle behind SFS, expressed in the 2nd Fundamental Property of Visual Hulls, is naturally embedded in the definition of the bounding edges. The bounding edges give us, as a representation for the visual hull, all the accurate information that can be obtained from the set of silhouette images. To locate the touching surface points, multi-image stereo (color consistency among images) is used. Two major difficulties of doing stereo : visibility and search size are both handled naturally by the properties of the bounding edges. The ability to combine the advantages of both SFS and Stereo is the main reason why using bounding edges/colored surface points gives better results in motion alignment than using voxel models obtained from SFS or SC, as is evident from the results in Section 5.1. Another disadvantage of using voxel models and Space Carving is that each decision (voxel is carved away or not) is made *individually* for each voxel according to a criterion involving thresholds. On the contrary, in locating colored surface points on bounding edges, the decision (which point on the bounding edge touches the object) is made *cooperatively* (by finding the point with the highest color consistency) along all the points on the bounding edge, without the need of adjusting thresholds. In other words, the information contained in bounding edges/colored surface points is more accurate than that contained in voxel models from SC/SFS. In parameter estimation, few but more accurate data is always preferred over abundant but less inaccurate data, especially in applications such as alignment.

References

[1] B. Baumgart. *Geometric Modeling for Computer Vision*. PhD thesis, Stanford University, 1974.

[2] J. S. D. Bonet and P. Viola. Roxels: Responsibility weighted 3D volume reconstruction. In *Proc. of ICCV'99*, Sept. 1999.

[3] C. Buehler, W. Matusik, and L. McMillan. Polyhedral visual hulls for real-time rendering. In *Proceedings of Eurographics Workshop on Rendering*, 2001.

[4] C. Buehler, W. Matusik, L. McMillan, and S. Gortler. Creating and rendering image-based visual hulls. Technical Report MIT-LCS-TR-780, MIT, 1999.

[5] G. K. M. Cheung. Visual hull construction, alignment and refinement across time. Technical Report CMU-RI-TR-02-05, Carnegie Mellon University, January 2002.

[6] M. Irani, T. Hassner, and P. Anandan. What does the scene look like from a scene point? In *Proc. of ECCV'02*, pages 883–897, May 2002.

[7] T. Joshi, N. Ahuja, and J. Ponce. Structure and motion estimation from dynamic silhouettes under perspective projection. Technical Report UIUC-BI-AI-RCV-95-02, UIUC, 1995.

[8] Y. C. Kim and J. K. Aggarwal. Rectangular parallelepiped coding: A volumetric representation of three dimensional objects. *IEEE Journal of Robotics and Automation*, RA-2:127–134, 1986.

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

[10] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. PAMI*, 16(2):150–162, February 1994.

[11] A. Laurentini. The visual hull of curved objects. In *Proc. of ICCV'99*, Sept. 1999.

[12] S. Lazebnik, E. Boyer, and J. Ponce. On computing exact visual hulls of solids bounded by smooth surfaces. In *Proc. of CVPR'01*, Dec. 2001.

[13] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *Proc. of SIGGRAPH 2000*, July 2000.

[14] M. Potmesil. Generating octree models of 3D objects from their silhouettes in a sequence of images. *Computer Vision, Graphics and Image Processing*, 40:1–20, 1987.

[15] R. Szeliski. Rapid octree construction from image sequences. *CVGIP : Image Understanding*, 58(1):23–32, July 1993.

[16] R. Szeliski. Image mosaicing for tele-reality applications. Technical Report CRL 94/2, Compaq Cambridge Research Laboratory, 1994.

[17] R. Szeliski and P. Golland. Stereo matching with transparency and matting. In *Proc. of ICCV'98*, pages 517–524, Jan. 1998.

[18] W. T. Vetterling and et. al. *Numerical Recipes in C*. Cambridge University Press, 1993.

[19] B. Vijayakumar, D. Kriegman, and J. Ponce. Structure and motion of curved 3D objects from monocular silhouettes. In *Proc. of CVPR'96*, pages 327–334, June 1996.

[20] K. Y. K. Wong and R. Cipolla. Head model acquisition and silhouettes. In *Proc. of International Workshop on Visual Form IWVF-4*, May 2001.

[21] K. Y. K. Wong and R. Cipolla. Structure and motion from silhouettes. In *Proc. of ICCV'01*, June 2001.