Stochastic Refinement of the Visual Hull to Satisfy Photometric and Silhouette Consistency Constraints

John Isidoro and Stan Sclaroff
Computer Science Department,
Boston University, Boston, MA, 02215
{jisidoro,sclaroff}@cs.bu.edu

Abstract

An iterative method for reconstructing a 3D polygonal mesh and color texture map from multiple views of an object is presented. In each iteration, the method first estimates a texture map given the current shape estimate. The texture map and its associated residual error image are obtained via maximum a posteriori estimation and reprojection of the multiple views into texture space. Next, the surface shape is adjusted to minimize residual error in texture space. The surface is deformed towards a photometrically-consistent solution via a series of 1D epipolar searches at randomly selected surface points. The texture space formulation has improved computational complexity over standard image-based error aproaches, and allows computation of the reprojection error and uncertainty for any point on the surface. Moreover, shape adjustments can be constrained such that the recovered model's silhouette matches those of the input images. Experiments with real world imagery demonstrate the validity of the approach.

1. Introduction

There are many computer vision techniques that reconstruct 3D models from multiple views. These techniques enable numerous applications: virtual reality, movie special effects, computer aided design, image compression, robot navigation and path planning, as well as object recognition, tracking, and manipulation. Arguably, each of these applications could benefit from more efficient reconstruction algorithms. Furthermore, many applications could benefit from knowing not only an object's 3D shape and surface coloring, but also the certainty of this data estimated at regular intervals over the reconstructed model's surface.

In this paper, we present an efficient, alternating *maximum a posteriori* (MAP) framework for 3D mesh-based reconstruction given multiple views of an object. The framework takes into account how uncertainty caused by image sensor noise is projected onto the surface's *texture space*. In addition, we present a way to use the texture space error image to find *frontier points* – object surface points that touch the visual hull. The frontier points can be anchored in place easily, thus enforcing silhouette constraints. This addresses a common problem in multiview reconstruction, where the recovered model may not match the silhouettes in the input images due to erroneous shape evolution in noisy situations.

2. Related Work

Many related approaches employ voxel-based space carving. Some produce visual hulls [13] quickly in a single sweep [22]. For a photo-consistent solution, single sweep methods [20, 18] are efficient but require the object to lie completely outside the convex hull of the camera centers. To handle arbitrary camera configurations, more expensive multi-sweep methods can be used [3, 7, 12, 21]. Voxel-based approaches are powerful and straightforward to implement, but they have two well-known shortcomings:

1. Voxel techniques generally produce blocky estimates with many ragged, irregular and disconnected surfaces [21]. In addition to this, a single miscarved voxel can irreversibly produce an erroneous hole in the model.

2. Depending on the camera configuration and model complexity, voxel techniques may incur significant computational cost when a high resolution surface is desired.

A number of remedies have been proposed for the first shortcoming, e.g., a probabilistic framework [1, 4], a level set approach [21], and a method that searches over a pixel neighborhood around a re-projected voxel's position for a color-consistent one [11]. However, these remedies require significant additional computation.

Image based rendering techniques are available for interactively rendering new views of visual hulls [14], photo hulls [21], and general scenes [9]. While they produce new views at interactive framerates and excel in multiview video applications, the image based representation does not have the compact size nor he rendering performance advantages of a texture mapped mesh for generating new views.

Our reconstruction framework is in a class of techniques that use mesh deformation methods. A related approach [24] starts with an ellipsoid and iteratively moves vertices, remeshing the object in order to build a more photometrically-consistent solution. In [5] a multi-stereo
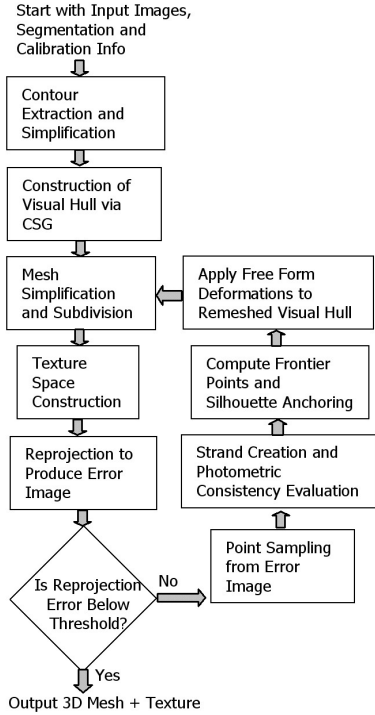
Start with Input Images, Segmentation and Calibration Info

Contour Extraction and Simplification

Construction of Visual Hull via CSG

Mesh Simplification and Subdivision ← Apply Free Form Deformations to Remeshed Visual Hull

Texture Space Construction ← Compute Frontier Points and Silhouette Anchoring

Reprojection to Produce Error Image ← Strand Creation and Photometric Consistency Evaluation

Is Reprojection Error Below Threshold? — No → Point Sampling from Error Image

Yes

Output 3D Mesh + Texture

Figure 1: System Overview Block Diagram

technique is used to estimate depth for each vertex of a visual hull mesh; in practice this technique will build a detailed mesh very quickly, but only when views are placed close enough together to make the window-based correlation stereo algorithm efficient.

Another novel feature of our framework is the silhouette anchoring technique. This ensures that the reconstructed mesh matches the silhouettes in each of the observed views. An alternative approach is to incorporate a soft silhouette constraint [16], where the polygonal visual hull is an active surface which is acted upon by three competing forces: a photo-consistency force, a smoothness force, and a silhouette maintenance force. These forces tend to pull all points towards their closest point on the visual hull. In contrast to this, our formulation finds frontier points [17] using the surface error image, and anchors them in place within our free-form deformation framework.

3. Approach

Fig. 1 gives an overview of the approach. Given a set of input images, full calibration information, and a background segmentation of the object in each view, constructive solid geometry is used to build a visual hull from the silhouette prisms. The resulting polygonal mesh is then simplified and adaptively subdivided to produce a mesh with a more uniform distribution of vertices over the surface. This mesh is used as an initial estimate for a two stage iterative refinement technique.

The first step estimates an RGB surface coloring (texture) for every point on the mesh surface. A surface param-

eterization, hereafter referred to as *texture space*, is defined by unfolding regions of the polygonal mesh onto a plane, and assembling the pieces together to form the texture map for the object. After this, the input images are projected into texture space, and a weighted average is taken to compute a texture map estimate as well as a reprojection error image.

The second step refines the 3D mesh using the current texture estimate. The reprojection error image is used to compute the probability density function (pdf) of error given surface location. Several points are sampled from this distribution using CDF inversion. Along the epipolar lines corresponding to these sampled points, the probability of the surface intersecting each position on the line is evaluated using a photometric consistency score. Also, photometrically consistent points which lie on the visual hull are found, and taken into account. The mesh surface is then pulled towards regions of lower error using a free-form deformation technique [19].

These two steps are repeated in alternation until convergence. Details of the algorithm will now be given.

3.1. Probabilistic Formulation

Let I be a sequence of input images taken from different vantage points, with their corresponding calibration information. Each individual input image will be denoted as I_i where i is the image index. The polygonal mesh representing the reconstructed object shape is written as O_c. The corresponding texture map, T_c, is an RGB image which represents albedo and diffuse illumination for every point on the surface. Given these definitions, the basic formulation can be written as a MAP estimation problem:

$$(\hat{T}_c, \hat{O}_c) = \arg_{T_c, O_c} \max p(I|O_c, T_c)p(O_c, T_c). \quad (1)$$

Each input image I_i can be modeled as the projection $\mathcal{P}_i$ into view i of O_c using T_c, with additive zero-mean Gaussian sensor noise with variance σ_i^2

$$p(I_i|O_c, T_c) \sim N(I_i; \mathcal{P}_i(O_c, T_c), \sigma_i^2). \quad (2)$$

Using this model, the images are conditionally independent from one another given T_c and O_c. Mathematically, the joint probability of the image sequence can be re-written as a product of single image probabilities

$$p(I|O_c, T_c) = \prod_i p(I_i|O_c, T_c). \quad (3)$$

One key issue is how to estimate both the object's texture map *and* its 3D shape. Our approach is iterative. It alternates between estimating the texture map using the current object shape, and then refining the object shape using the current texture map. The idea of alternating between the estimation of two conditionally dependent unknowns is similar in spirit to [2].

Given a polygonal visual hull as the initial shape estimate O_c, the maximum likelihood estimate of the texture can be

found as follows:

$$\hat{T}_c = \arg_{T_c} \max p(O_c)p(T_c) \prod_i p(I_i|O_c, T_c) \quad (4)$$

and by using Bayes' Rule this can be re-written:

$$\hat{T}_c = \arg_{T_c} \max p(O_c)p(T_c) \prod_i \frac{p(T_c|O_c, I_i)p(I_i|O_c)}{p(T_c|O_c)}. \quad (5)$$

For this paper, the object shape O_c is used to establish a relationship between the input images I_i and the texture T_c. Because we do not enforce any priors on input images, textures or object shapes by themselves, the $p(T_c|O_c)$, $p(T_c)$ and $p(O_c)$ terms are assumed to be constant. Using these assumptions, the $p(I_i|O_c)$ also reduces to a constant via marginalization.

We approximate the probability of the texture T_c given the object shape and a single input image as:

$$p(T_c|O_c, I_i) \sim N(T_c; \mathcal{P}_i^{-1}(O_c, I_i), \mathcal{P}_i^{-1}(O_c, \sigma_i^2)). \quad (6)$$

In this equation, $\mathcal{P}_i^{-1}$ represents a *back-projection* from view i back into the texture. In computer graphics, a projection of an image back into texture space is known as texture back-projection. Eq. 6 contains back-projections of both the input image and the sensor noise. Using this technique, the texture from the visible portions of the surface can be estimated from the input images. Also, by back-projecting the noise model, a per-texel variance can be estimated for each of the back-projected images.

Given the formulation above, the MAP estimate of the texture can be computed using a weighted average of the back-projected input images:

$$\hat{T}_c = \arg_{T_c} \max \prod_i p(T_c|O_c, I_i) \quad (7)$$

for each texel k in the texture

$$\hat{T}_{ck} = \frac{1}{\sum_i w_{ik}} \sum_i w_{ik} \mathcal{P}_{ik}^{-1}(O_c, I_i) \quad (8)$$

where the weights w_{ik} are the inverse of the per-texel variances. For visible portions of the texture from view i, the per-texel variance is computed using a back projection of the noise. For regions that are not visible from view i, the per-texel variance for this texel in this view is considered to be infinite, and the corresponding value in w_{ik} is zero.

Now that we have a current estimate of the texture T_c, we can estimate O_c using this value of T_c:

$$\hat{O}_c = \arg_{O_c} \max \prod_i p(T_c|O_c, I_i). \quad (9)$$

Using Eq. 6, it can be shown that the best estimate of O_c is the shape that minimizes the squared reprojection error:

$$\hat{O}_c = \arg_{O_c} \min \sum_{k \in T} \sum_i \frac{(\hat{T}_c - \mathcal{P}_i^{-1}(O_c, I_i))^2}{\mathcal{P}_i^{-1}(O_c, \sigma_i^2)} \quad (10)$$

where the sum $\sum_{k \in T}$ is over all texels in the texture.

Unfortunately, there is no simple closed-form solution for estimating the O_c that minimizes the error. This is due to the fact that the back-projected image $\mathcal{P}_i^{-1}(O_c, I_i)$ is a highly non-linear function of O_c with many local minima. Therefore, to obtain an estimate of O_c we propose a novel sampling-based method. Samples are drawn from the re-projection error image. Using these samples, the mesh is locally drawn towards a photometrically consistent solution using kernel-based freeform deformations [19]. Details of the complete algorithm are given in the following sections.

3.2. Mesh Initialization

An initial visual hull is produced via standard background subtraction, followed by contour extraction, silhouette cone generation, and polygonal constructive solid geometry (CSG). Given an initial visual hull, a second mesh that has a more uniform distribution of vertices over the surface is created via mesh simplification and subdivision techniques. The remeshed visual hull is the initial estimate for the surface mesh $\hat{O}_c$. We next compute a texture space for this mesh in order to enable texture back-projection.

3.3. Texture Space Construction

To evaluate photometric consistency, we employ a single error image in the model's *texture space* that represents the error for the entire surface of the model. This way, any errors in the model are only represented once. Error computation using this representation requires only $\mathbf{O}(n)$ reprojections where n is the number of input images. (In contrast, many other approaches compute error in image space for each of the input images thus requiring $\mathbf{O}(n^2)$ reprojections.) Moreover, by producing a mapping from surface space to image space that is area and angle preserving, we are able to estimate properties for the surface such as surface coloration and reprojection error with no unmeaningful bias towards any particular region.

To compute this texture space we use a rigid unfolding technique that maps our mesh onto the plane. This is described in more detail in Section 5.1.

3.4. Texture and Error Image Estimation

The density of pixel samples reprojected onto texture space can vary. A region on the model which is viewed at an oblique angle, or far from the camera in all input images may have only a few pixels project to it. Conversely, a region on the model which is parallel to the image plane, visible and close to the camera will have far more pixels the project to it. Intuitively, the more pixels that project to a given region on the model, the better that region can be reconstructed. We assume that each input image pixel p_k is the per-channel sum of an RGB color value c_k plus additive per-channel Gaussian (sensor) noise $u_k \sim \mathbf{N}(0, \sigma^2)$,

$$p_k = c_k + u_k. \quad (11)$$

In [10] we show that the per channel variance of a weighted sum of projected pixels onto a region of the object R is proportional to the pixels' projection areas, assuming equal projection areas A_μ. We also show that a good approximation to the pixel projection area A_{ik} of pixel k in view i is proportional to the squared distance between the camera and the model d_{ik}^2, and inversely proportional to the cosine of the angle of incidence $(N_{ik} \cdot V_{ik})$ of the view vector with the model

$$ A_{ik} \simeq \frac{d_{ik}^2}{(N_{ik} \cdot V_{ik})}. \quad (12) $$

Using the inverse variance as the weights, a weighted average of the input images back-projected into texture space can be computed:

$$ \hat{T}_c = \frac{1}{\sum_i w_i} \sum_i w_i \mathcal{P}_i^{-1}(O_c, I_i) \quad (13) $$

where w_i is a weighting image

$$ w_i = v_i \frac{1}{A_i}. \quad (14) $$

The binary visibility mask v_i only has ones in the regions of the texture that are visible from view i. These visibility masks can be computed entirely in graphics hardware via a depth-based shadow technique [6].

The texture space image $\hat{T}_c$ is the weighted average of projected pixel values from each view point. Using this weighted average approach, a color reprojection error image can also be computed:

$$ CError(\hat{T}_c) = \sum_i w_i (\hat{T}_c - \mathcal{P}_i^{-1}(I_i))^2. \quad (15) $$

The weighted average texture and error image can be computed quite efficiently on a modern graphics card with pixel shader capabilities and a floating point pixel pipeline.

3.5. Sampling from the Texture Space Error Image

At this point, the goal is to determine how to move the 3D mesh vertices of the current shape estimate $\hat{O}_c$ in order to improve the photometric consistency. The main issues are how to determine which vertices need to be moved, where they need to be moved to, and how to accomplish this in an efficient manner.

Our system adjusts the surface by performing a series of 1D epipolar searches for displacements with a higher photometric consistency at randomly selected surface points. Samples are selected as follows. The reprojection error image is used to construct a pdf of error given surface location $p(Error(T_c)|k)$.

$$ p(Error(T_c)|k) = \frac{max_{r,g,b} CError(T_c)}{\sum_{m \in T_c} max_{r,g,b} CError_m(T_c)}. \quad (16) $$

The per-texel $max_{r,g,b}$ is used since photometric consistency requires the reprojection error be below the noise threshold for each color channel. Thus, we consider the channel with maximum error on a per-texel basis. The sum over all texels m in the denominator of this equation is a normalization term.

Given the pdf computed in this way, several points on the surface are sampled using CDF inversion [23].

3.6. Stochastic Epipolar Search via "Strands"

For each sample point chosen in the reprojection error image, an epipolar ray is generated for each viewpoint the point on the surface is visible from. The basic idea is to find a photometrically consistent solution along the epipolar rays produced from that sample. We call these epipolar rays *strands*. The strands will help determine the error minimizing freeform deformations.

Starting from the first point of intersection of each strand with the visual hull, photometric consistency is evaluated at evenly-spaced sampling intervals along the strand until the next intersection point with the visual hull is reached. Note that the visual hull is used instead of the current mesh estimate O_c to bound the sample space. Thus, backtracking is possible if some sections of the estimate are deformed too far inward. For the results in this paper, the sampling interval is determined using the length of the visual hull's bounding box longest diagonal.

For each position along a strand, the 3D point location is reprojected into each of the other images I_i that are visible from the strand's ray start point on the surface. Strand visibility can be approximated quickly by using the value of visibility image v_i at the strand's start point. For other positions along the strand, this visibility information is approximate. To account for this, we use an error metric which allows for rejecting viewpoints for which positions along the strand may not be visible.

At each position along the strand, when photometric consistency is evaluated, any views which have a score above a threshold are assigned the worst possible photometric consistency score. This threshold is determined by the sensor noise. Thus, solution points having different visibility than the strand's intersection with the current estimate model can be found. The thresholding also gives the system some robustness to specular highlights, which are view-dependent and generally present in a particular position on the model in a single view. As a result of this thresholding, the best value on the strand is the one which is photometrically consistent in the largest number of views. If there are many points which are consistent in the same number of views, the point closest to the current surface estimate is selected.

If an acceptable photo-consistent position along the strand is found, then it contributes to the deformation field applied to the mesh. For each view consistent with that point on a strand, we compute a vector in the direction of the corresponding epipolar ray, with magnitude equal to the distance between the strand start point and the winning

photo-consistent point. The vector of shortest distance is used as a free-form deformation vector in the shape deformation stage.

4. Enforcing the Silhouette Constraint

Silhouette constraints also fit neatly into our framework. Assuming the computed image silhouettes are correct, and the visual hull accurate enough, there should be a photometrically consistent point on the surface of the visual hull corresponding to each point of each input image silhouette.

As a once-per-reconstruction preprocessing step, an indexed distance transform is computed for each of the input silhouette images. This transform results in an image where each pixel contains an index and distance to its closest silhouette point. Thus, it can be quickly resolved whether or not a back-projected point lies on the silhouette, and if so, which silhouette point it is closest to.

Per iteration, a bin is created for each point on the silhouettes in each of the input images. These bins keep a list of texels which project to this point on the silhouette. Every texel in the texture error image is back-projected into each of the input images, and the closest silhouette point is found. If the silhouette point is less than two pixels away from the back-projected texel, then the point is added to the corresponding bin. Each bin contains a list of all the surface points that project to it.

For each bin, the texel with lowest error is found. The corresponding points in 3D space are considered to be *frontier points*: points on the shape that touch the visual hull. For each of the frontier points, a free-form deformation vector of zero length is created. This will tend to anchor the frontier points in place during mesh deformations, while still providing FFD interpolation properties.

In our tests, for a 512×512 texture and 12 views, frontier point computation took less than 8 seconds, due to the quick distance transform lookup.

5. Implementation Details

At the beginning of each iteration of the algorithm the polygonal model is remeshed in order to maintain a roughly uniform distribution of vertices over the surface of the mesh. The mesh vertices can be thought of as the sample points through which shape is interpolated; thus, an even distribution of vertices over the surface allows us to better represent the shape produced via surface deformation. First, a variant of Garland's quadratic error metric [8] is used to simplify the mesh. Edges shorter than a minimum edge length threshold d_{eMin} are collapsed as long as the genus of the mesh is preserved. After this the mesh is adaptively subdivided. Any edges in the mesh longer than a maximum edge length d_{eMax} are subdivided into a minimal number of equally sized segments shorter than d_{eMax}. Each new polygon is recursively triangulated and any new edges are adaptively subdivided using the same method.

For free-form deformations, we use a variation of the kernel-based free-form deformation technique [19] that we used [10]. However, we incorporate a small per-strand

weight s_j based on the certainty of the surface position estimate of the strand. In this case s_j is equal to the number of views that are photometrically consistent at the best position on the strand. Thus strands with higher certainty have greater influence on the solution.

5.1. Texture Space Generation via Rigid Unfolding

The goal of our texture space generation technique is to efficiently generate a non-distorted texture space for the mesh with an adequate texel to pixel ratio. Our technique breaks the mesh into small connected groups of triangles and unfolds them into the texture plane. The unfolded triangle groups are packed into as small a bounding box as possible given time constraints. Fig. 2 shows the texture space constructed for a model of an armchair.

Each unfolding group starts with a triangle that has not been unfolded yet. In a breadth-first traversal of the face adjacency graph, we attempt to unfold any of the three adjacent triangles that have not been unfolded yet. If an unfolded triangle overlaps any other unfolded triangles in this group, then the triangle is rejected from the group. Otherwise the triangle is added to the group and its adjacent faces are added to the queue of faces to unfold. Triangles are unfolded until either the group has a preset maximum number of triangles, or there are no more adjacent triangles in this group that can be unfolded.

The unfolded groups must now be packed compactly into a single texture. We generally have over 200 unfolded groups. The translational polygon packing problem is known to be NP-hard [15]; therefore, heuristic solutions are employed that trade optimality for speed.

Our heuristic incrementally packs each of the unfolded groups into texture space. Packing the first group is trivial. To add another group to texture space, we first pick a uniformly sampled random point on the current packing's bounding box. Next, we perform a bisection search on the line segment formed by the random point, and the bounding box center to find the closest position for the new group without intersecting the current packing. This step is repeated a number of times, and the translation producing the smallest bounding box is used to add the group to the packing. An alternative algorithm was proposed in [15], and its evaluation in our application remains a topic for future investigation.

To determine the appropriate texture map resolution, we evaluate the texel-to-pixel Jacobian at each vertex of the unfolding to find the minimum texel projection rate. We then choose a texture resolution such that there are at least four texels per pixel for all projections.

6. Results

Our reconstruction framework has been evaluated in experiments with real world imagery. Some results of the experiments are briefly described here. All experiments were conducted on a 1.733GHz Athlon-based PC with 2GB of RAM and a Radeon 9700 graphics card with 128MB RAM.
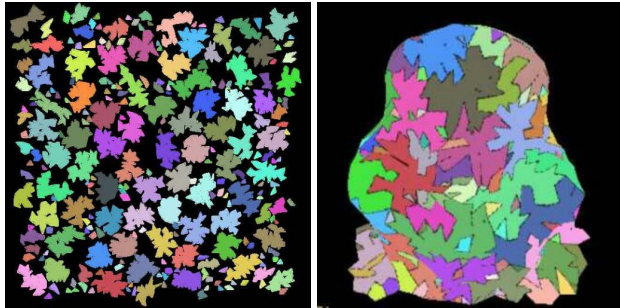
Figure 2: Texture space unfolding for a reconstructed mesh of a toy armchair. The left image shows the unfolding, where each unfolded group is shown in a different color. The right image shows the armchair model rendered using the same texture group coloring scheme. Texture space generation took 37 seconds for a mesh containing 3844 polygons, and produced 246 unfolded groups.
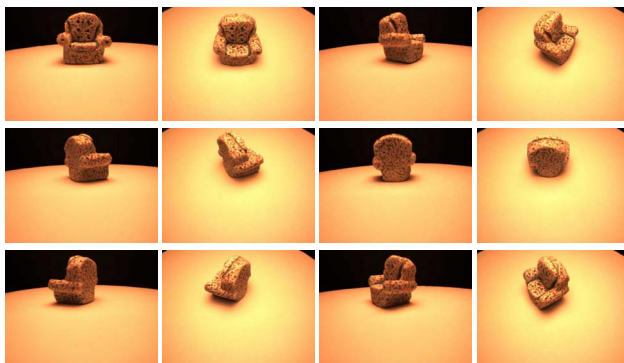

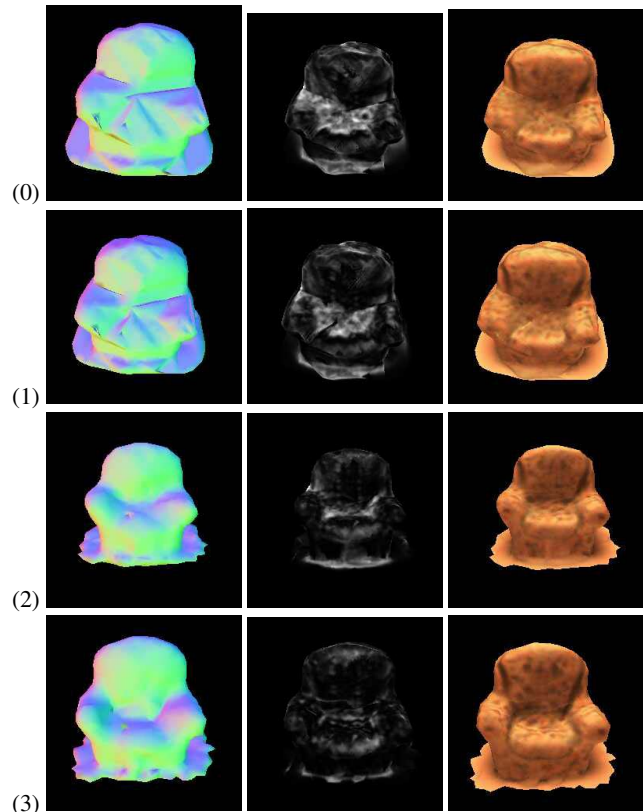
Figure 3: The twelve input views of the toy armchair.

Figure 4: Evolution of the armchair shape during reconstruction: (0) initial visual hull mesh, (1) after first iteration, (2) second, etc. Each row shows the mesh rendered via Gouraud shading, the error texture applied to the model (brighter regions signify higher error), and the reconstructed texture applied to the model. The corresponding average pixel error for each iteration is 0.097, 0.079, 0.042, and 0.040 respectively.

In the first experiment, we reconstruct the model of a toy armchair. Fig. 3 shows the 12 calibrated input images. Each image is 512×384 pixels. The algorithm was run for three iterations. In each iteration, mesh simplification and subdivision took less than 5 seconds, resulting in meshes of 3800-4300 triangles. Each texture space generation and texture estimation step took less than 40 seconds and generated on average 250 texture groups packed together into a 512×512 texture map. Error image sampling and strand generation took under 50 seconds to generate 200 sample points, and 486 (on average) strands per iteration. Generating the silhouette constraint vectors took less than 6 seconds per iteration and generated 7758 vectors. Freeform deformation took less than 2 seconds per iteration. In total, the final result was obtained in under 4 minutes.

Fig. 4 shows the results of the reconstruction after each iteration. In each iteration, the average pixel error was reduced as the mesh deformed towards the correct solution. Note that the visual hull used as the initial 3D mesh did not properly represent the armchair's concavity near the seat cushion. Our algorithm correctly recovers the concavity, improving the estimate and its texture map in each iteration.

In the next experiment, we reconstruct the model of a small bowl-shaped object using the 12 calibrated input images shown in Fig. 5. As before, the algorithm was run for three iterations. The model reconstruction obtained in each iteration is shown in Fig. 6. As can be seen in the figure, the proper concavity was not even present in the initial mesh obtained from the visual hull. However, with each iteration the representation of the concavity improves, as the average pixel error decreases. In total, this reconstruction was obtained in just under 5 minutes, and resulted in a mesh containing 3740 triangles.

Figs. 7 and 8 show two additional reconstructions produced by our method: a pretzel and a plastic toy "alien creature." In both examples, the framework ran for three iterations. The total time for reconstruction was less than 5 minutes in each case.

7. Discussion and Conclusions

We have presented a new framework for estimation of a photometrically consistent 3D mesh and its texture map from multiple, calibrated views of an object. The MAP solution is obtained via an alternating minimization algorithm. As seen in the experiments, the framework is relatively efficient and effective in improving the photometric consistency of the recovered 3D mesh. Our technique has
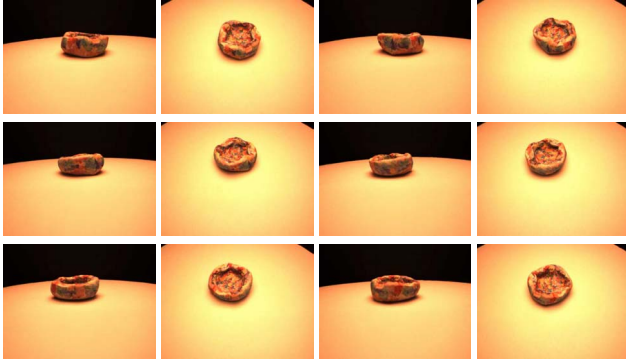
IEEE
COMPUTER
SOCIETY

Figure 5: The twelve input views for the bowl-shaped object.

the added advantage that it can constrain the 3D mesh to match the object's silhouettes in the original input views.

In practice, the use of the silhouette constraint improved the results for all four example objects. Without the silhouette constraint, the algorithm still reduced the average texel reprojection error. However, the resulting mesh would decrease in size each iteration, and result in an estimate smaller than the true shape of the object. By using the silhouette constraint, the estimates were much closer to the true shape of the object.

The convergence properties of our algorithm are expected to be similar to Generalized EM algortihms, where one step is only guaranteed to improve the likelihood rather than maximize it. This is achieved by only accepting changes to the shape that improve the likelihood. Our random sampling technique allows for multiple trials from the same estimate, so this is reasonable.

As with other multiview reconstruction techniques, sufficient surface detail as well as sufficient visibility is required to accurately reconstruct the surface shape. The more varied the surface coloring, and the more cameras that see the surface of the object, the better the algorithm will perform. As seen from the pretzel example, the system handles a small amount of specular highlights due to the view rejection technique used to approximate visibility. To reconstruct more reflective objects, a different photometric consistency metric would be needed, perhaps using knowledge of the incident light cast upon the object.

Another limitation of the technique is that the genus of the final object will have the same genus as the visual hull. In its present state, the polygonal deformation technique does not change the genus of the mesh. An interesting future direction for this work is to investigate mesh techniques which produce genus changes that improve the likelihood of the shape.

In the future, we hope to investigate new ways to generate texture spaces within our framework. Ideally, we seek a texture space generation algorithm that accounts for the effects of texture distortion and can provide a bounded amount of distortion in texture reconstruction. This distortion is due to differences in sampling rates of the reprojected textures over the surface of the model.
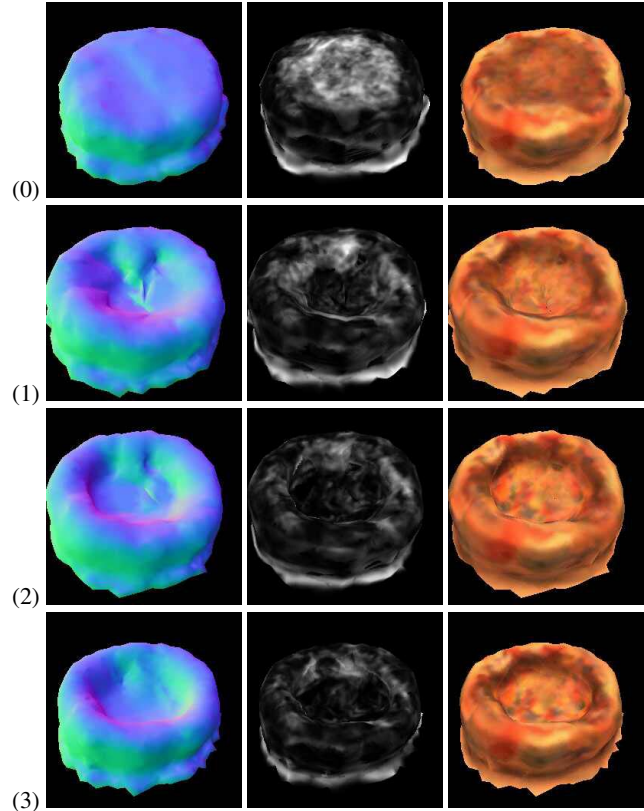

Figure 6: Evolution of the bowl shape during model reconstruction: (0) initial visual hull mesh, (1) mesh after first iteration, (2) second, etc. Each row shows the mesh rendered via Gouraud shading, the error texture applied to the model (brighter regions signify higher error), and the reconstructed texture applied to the model. The corresponding average pixel error for each iteration is 0.144, 0.113, 0.093, and 0.076 respectively.

Another topic for future investigation is adaptive subdivision strategies that produce higher geometric detail (more triangles) only where it is needed: in regions of the model where the reprojection error is higher. Thus, the vertex density could be updated from iteration to iteration.

References

[1] A. Broadhurst, T. W. Drummond, and R. Cipolla. A probabilistic framework for space carving. In *Proc. ICCV*, pages I:388–393, 2001.

[2] I. Csiszar and G. Tusnady. Information geometry and alternaning minimization procedures. *Statistics and Decisions*, Supplement Issue 1.

[3] W. B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. In *ICCV Workshop on Vision Algorithms and Theory and Practice*, pages 100–115, 1999.

[4] J. S. de Bonet and P. Viola. Roxels: Responsibility weighted 3D volume reconstruction. In *Proc. ICCV*, pages 418–425, 1999.

[5] C. Esteban and F. Schmitt. Multi-stereo 3d object reconstruction. In *International Symposium on 3D Processing, Visualization, and Transmission (3DPVT)*, pages 159–166, 2002.
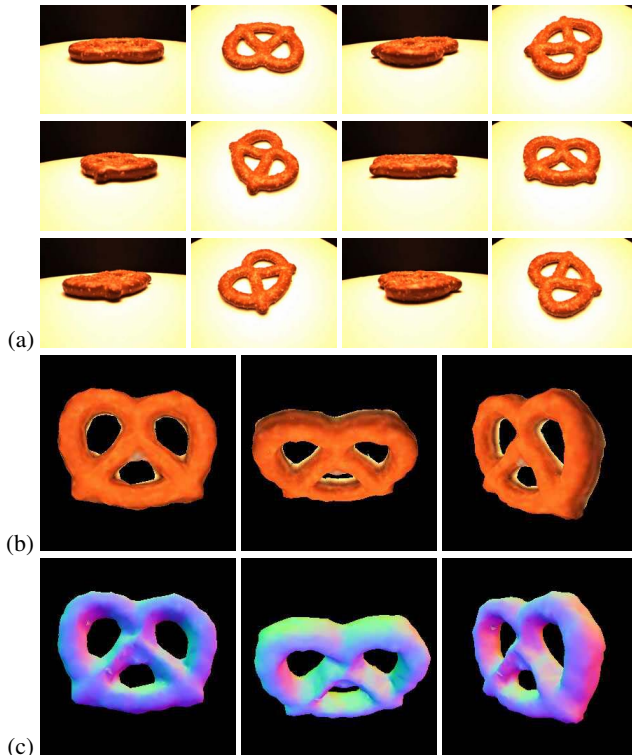
Figure 7: A pretzel reconstructed from 12 calibrated images: (a) input images, (b) three views of final reconstructed object rendered with texture applied, (c) final mesh rendered using Gouraud shading. As a result of three iterations of the algorithm, the average pixel error was reduced from 0.083 to 0.049.
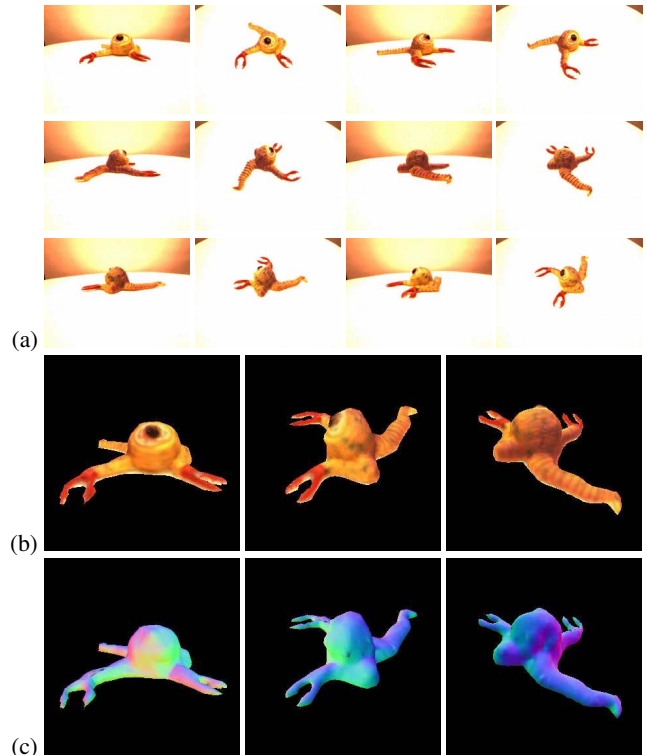

Figure 8: A toy "alien creature" reconstructed from 12 calibrated images: (a) input images, (b) three views of final reconstructed object rendered with texture applied, (c) final mesh rendered using Gouraud shading. As a result of three iterations of the algorithm, the average pixel error was reduced from 0.129 to 0.069.

[6] C. Everitt, A. Rege, and C. Cebenoyan. Hardware shadow mapping. Technical report, NVIDIA, Santa Clara, CA, 2002.

[7] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Model construction for turn-table sequences. In *Proc. SMILE'98*, pages 155–170. Lecture Notes in Computer Science 1506, 1998.

[8] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *Proc. SIGGRAPH*, pages 209–216, 1997.

[9] M. Irani, T. Hassner, and P. Anandan. What does the scene look like from a scene point? In *Proc. ECCV*, pages xx–yy, 2002.

[10] J. Isidoro and S. Sclaroff. Stochastic mesh-based multiview reconstruction. volume 1, pages 568–577, 2002.

[11] K. N. Kutulakos. Approximate N-view stereo. In *Proc. ECCV*, pages 67–83, 2000.

[12] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International J. of Computer Vision*, 38(3):197–216, 2000.

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

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

[15] V. Milenkovic. Densest translational lattice packing of non-convex polygons (extended abstract). In *Symposium on Computational Geometry*, pages 280–289, 2000.

[16] S. Nobuhara, T. Wada, and T. Matsuyama. 3d shape from multi-viewpoint images using deformable mesh model (in japanese). Technical Report No.131 - 009, IPSJ SIGNotes Computer Vision and Image Media, March 2002.

[17] J. Porrill and S. Pollard. Curve matching and stereo calibration. *Image and Vision Computing*, 9(1):45–50, 1991.

[18] A. Prock and C. Dyer. Towards real-time voxel coloring. In *Proc. DARPA Image Understanding Workshop*, 1998.

[19] D. Ruprecht and H. Muller. Spatial free form deformation with scattered data interpolation methods. *Computers and Graphics*, 19:63–72, 1995.

[20] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. CVPR*, pages 1067–1073, 1997.

[21] G. Slabaugh. *Novel Volumetric Scene Reconstruction Methods for New View Synthesis*. PhD thesis, Georgia Institute of Technology, November 2002.

[22] R. Szeliski. Real-time octree generation from rotating objects. Technical Report 90/12, Digital Equipment Corporation, Cambridge Research Lab, 1990.

[23] J. VonNeumann. in a letter to s. ulam in 1947. *Los Alamos Science.*, page 135, 1987.

[24] L. Zhang and S. Seitz. Image-based multiresolution modeling by surface deformation. Technical Report CMU-RI-TR-00-07, Carnegie Mellon University, January 2000.