Appeared in: CVPR 2000, Vol. 1, pp. 332-338, June 13-15, Hilton Head, SC

Image-Consistent Surface Triangulation

Daniel D. Morris Takeo Kanade
Robotics Institute,
Carnegie Mellon University,
Pittsburgh, PA 15213-3890
{ddmorris,tk}@ri.cmu.edu

Abstract

Given a set of 3D points that we know lie on the surface of an object, we can define many possible surfaces that pass through all of these points. Even when we consider only surface triangulations, there are still an exponential number of valid triangulations that all fit the data. Each triangulation will produce a different faceted surface connecting the points.

Our goal is to overcome this ambiguity and find the particular surface that is closest to the true object surface. We do not know the true surface but instead we assume that we have a set of images of the object. We propose selecting a triangulation based on its consistency with this set of images of the object. We present an algorithm that starts with an initial rough triangulation and refines the triangulation until it obtains a surface that best accounts for the images of the object. Our method is thus able to overcome the surface ambiguity problem and at the same time capture sharp corners and handle concave regions and occlusions. We show results for a few real objects.

1 Introduction

The shape of a 3D object is often represented as a 3D point cloud in vision literature, and much research has focused on obtaining accurate 3D point clouds in fields such as stereo and Structure from Motion. This, however, is only part of the shape recovery problem. Often what is desired is a full surface model of the object. In many cases the surface model of choice is a triangulation of the features, and this is the model we consider in this paper.

The existence of many possible triangulations, each producing a different surface, poses an immediate difficulty in triangulating 3D points. For example, Figure 1 illustrates two of many possible triangulations of a given set of 3D points. There is obviously insufficient information in the point set alone to define a continuous surface, and choice between these or other solutions is arbitrary. Determining the most "desirable"

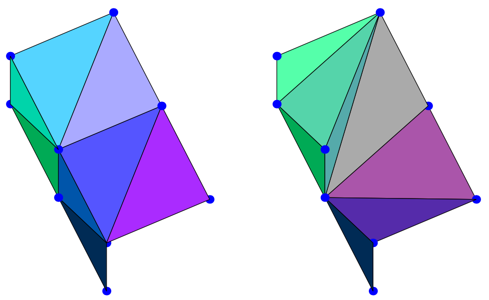
Figure 1: Two possible surface triangulations of a 3D point cloud. There is insufficient information in the 3D points alone to choose between these or a host of other possible surfaces.

solution requires either heuristics or further information. In this paper we examine how the information provided by a set of images of the object can be used to guide selection of a surface triangulation.

Much attention has been paid to the task of obtaining a surface mesh from 3D data points on an unknown surface in the graphics literature [1, 2, 3, 4, 6, 9]. Typically geometric properties of the mesh are optimized, such as relative shape, size and orientation of triangles, in the hope that this will result in faithfulness to an unknown surface. This works well for dense points on smooth surfaces, but when the features are sparse and the object contains sharp corners, mesh properties alone will not necessarily enforce good approximation to the true surface and numerous artifacts are often generated. In many vision problems, such as Structure from Motion, we obtain sparse 3D data and want to create a surface model from this. Currently, surfaces are specified manually [5, 7] or else various heuristics are used to create a suitable triangulation, such as performing a 2D Delaunay triangulation in one of the images and projecting this into 3D. When these methods produce artifacts, manual correction is necessary. Example artifacts resulting from selecting a geometri-

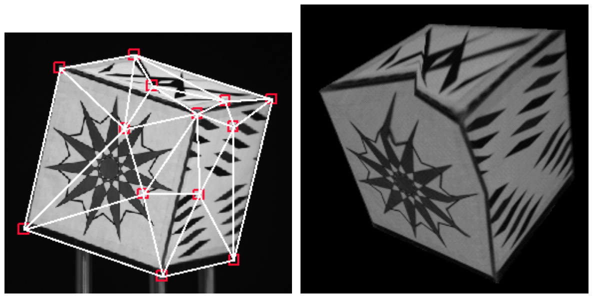

Figure 2: Triangulation obtained using Delaunay triangulation in a projected plane and projected back onto the 3D object. Notice the edges do not correspond to the cube edges, with the result that the texture-mapped shape on the right has large artifacts.

cally optimized triangulation for surface modeling are illustrated in Figure 2. Here some of the triangular faces cut through the true surface, and when the resulting shape is texture-mapped, the deformations are clearly visible.

Vision applications typically have additional surface information in terms of images of the object. Here we propose to use the images for automatically determining the best triangulation of the features. A triangulation determines a texture-mapped surface model of the object, and we "select" a particular triangulation using the error between the images and the reprojected surface model. We will formulate this as a maximum likelihood estimation problem over the space of triangulations.

Our technique requires optimizing over the space of valid triangulations. We use edge swaps [12, 9] to generate new triangulations and depend on a theorem by Ore [13] to show that this enables us to achieve all possible triangulations. Then we use a greedy algorithm to search for the best triangulation. Figure 3 illustrates the goal of our algorithm; namely a refinement of the surface triangulation that is closer to the true surface and as a result removes artifacts in the re-projected images. In this work we do not consider adding or removing vertices of a triangulation, but simply ask what is the best triangulation obtainable from a given vertex set.

2 Surface Modeling

Ideally we would like to find the true surface of the object viewed in a set of images. If we could perfectly model surface shape, reflectance and lighting, we could create a virtual surface whose projection into all the images is identical to the actual images. The optimal virtual surface is consistent with the images, and while not necessarily unique [11], constitutes our surface model. We will make a number of simplifications and approximations to this surface modeling task, and our goal will be to find a restricted surface model whose projection into all of the images is as close as possible to the actual images. We formulate this as a maximum likelihood estimation problem over the space of triangulations.

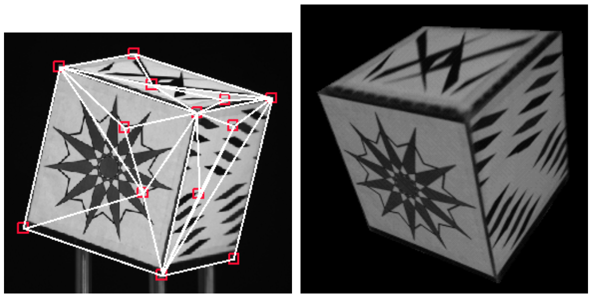

Figure 3: The surface triangulation in Figure 2 is corrected, and the resulting texture-mapped shape on the right does not have large artifacts.

2.1 Assumptions and Approximations

We start with a set of images $\mathcal{I} = \{I_1, \ldots, I_N\}$ of a rigid object taken with a calibrated camera. We know the 3D coordinates of a set of (possibly sparse) feature points on the surface of the object and their projected coordinates in the image taken by a calibrated camera. Equivalently we may just have a registered set of features in an image set, and use Structure from Motion to obtain the 3D coordinates and camera positions. For simplicity and generality we assume a Lambertian model for the surface reflectance and constant lighting. We will triangulate the feature points and hence obtain surface detail up to the size of the triangles. We ignore the possible non-uniqueness of the solution and assume that the images along with the triangulation constraints are sufficient to define a unique surface.

2.2 Surface Estimation

Our surface model is a triangulation, $\mathcal{T}_i$, along with a texture map, $\mathcal{A}_i$, on all of the surface triangles. A particular triangulation determines a set of edges and faces connecting the 3D feature points. In general there are many possible triangulations of a 3D feature set. Given a triangulation and texture map constituting the virtual surface, the predicted image set $\hat{\mathcal{I}}$ is obtained as a projection of this:

$$\hat{\mathcal{I}} = \Pi[\mathcal{T}_i, \mathcal{A}_i] \tag{1}$$

where $\Pi[\ldots]$ is our known projection camera operator onto all of the images.

The texture map, $\mathcal{A}_i$, for a given triangulation can be estimated directly from the actual image set, $\mathcal{I}$, and the triangulation, $\mathcal{T}_i$, by inverting the projection:

$$\mathcal{A}_i = \Pi^{-1}[\mathcal{T}_i, \mathcal{I}] \tag{2}$$

where the inverse projection operator Π^{-1} projects each image onto the triangulation $\mathcal{T}_i$ and averages faces onto which more than one image projects. Thus the predicted image $\hat{\mathcal{I}}$ from equation (1) depends only on the triangulation $\mathcal{T}_i$ and original image sequence $\mathcal{I}$.

Our prior probability on triangulations is assumed to be uniform, and so our goal will be to obtain the maximum likelihood triangulation for the given image set. This is achieved by finding:

$$\underset{\mathcal{T}_i}{argmax} \quad P(\mathcal{I}|\mathcal{T}_i) \tag{3}$$

over all triangulations $\mathcal{T}_i$, where $P(\mathcal{I}|\mathcal{T}_i)$ denotes the likelihood of a triangulation $\mathcal{T}_i$ for image set $\mathcal{I}$. We can derive an expression for this likelihood in terms of the actual images and the predicted images as follows.

Suppose there is a triangulation, $\mathcal{T}_*$, corresponding to the true surface and from it we obtain a predicted image set $\hat{\mathcal{I}}_*$. Let a pixel in an image be denoted as $\mathbf{x}$, and we model each pixel in an actual image as resulting from the back-projected image pixel and a noise term:

$$\mathcal{I}(\mathbf{x}) = \hat{\mathcal{I}}_*(\mathbf{x}) + \Delta\eta. \tag{4}$$

We assume that the noise, η, is independent for all pixels in all images, has zero mean and is Gaussian with variance σ^2. If we have the image set $\mathcal{I}(\mathbf{x})$ and the correct predicted image set $\hat{\mathcal{I}}_*(\mathbf{x})$, the variance of the noise can be estimated directly from this equation as the mean squared error over all M pixels: $\sigma^2 \approx \sum_{\mathbf{x}\in\mathcal{I}}(\mathcal{I}(\mathbf{x}) - \hat{\mathcal{I}}(\mathbf{x})_*)^2/M$.

Hence the likelihood of a triangulation is given by a normal distribution:

$$P(\mathcal{I}|\mathcal{T}_*) = N(\hat{\mathcal{I}}(\mathbf{x}), \Sigma), \tag{5}$$

with variance:

$$\Sigma = \sum_{\mathbf{x}\in\mathcal{I}}(\mathcal{I}(\mathbf{x}) - \hat{\mathcal{I}}(\mathbf{x}))^2. \tag{6}$$

Our predicted images $\hat{\mathcal{I}}$ are obtained using the projection: $\hat{\mathcal{I}} = \Pi[\mathcal{T}_i, \Pi^{-1}[\mathcal{T}_i, \mathcal{I}]]$ from equations (1) and (2). The likelihood is maximized when the variance, Σ, is minimized. If we approximate the likelihood of non-optimal triangulations by analogous expressions, then the maximum likelihood estimate is obtained by finding the triangulation that gives the smallest total variance Σ. Thus our maximum likelihood solution is simply the minimum variance solution. This also corresponds to the triangulation whose actual and predicted images have the minimum squared difference over all pixels.

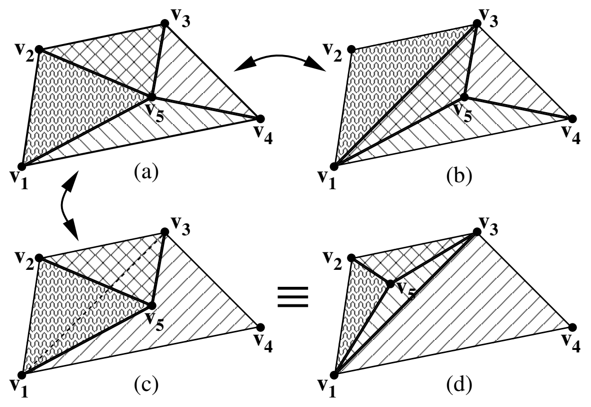

Figure 4: (a) An example triangulation on five vertices. The boundary B contains 4 vertices and their joining edges. The four triangular faces are each shaded with a different pattern. The triangulation in (b) is obtained by swapping edge $\mathbf{e}_{25}$ of (a) which becomes $\mathbf{e}_{13}$. The swap operation on edge $\mathbf{e}_{45}$ of the triangulation in (a) is possible even though the quadrilateral obtained by joining its adjacent faces, $\mathbf{f}_{145}$ and $\mathbf{f}_{345}$, is not convex. Triangulation (c) is the result of this swap, and if viewed as a surface model in 3D, face $\mathbf{f}_{135}$ is hidden. The graph is still planar, however, and can be re-embedded in the plane so that no edges intersect except at vertices, as shown in diagram (d). Finally we note that some edges in (d) cannot be swapped such as $\mathbf{e}_{15}$ since the new edge that would be formed, in this case $\mathbf{e}_{23}$, already exists.

3 Surface Triangulations

We have reduced our surface estimation task to a task of searching for the best triangulation over all the triangulations on a given vertex set. This is still an exponential search problem, but at least one that can be well defined and constrained. In this section we define a triangulation and the space of possible triangulations. We propose using edge swaps as a basis for traversing the space of triangulations, and finally provide a search algorithm to find the best surface triangulation.

3.1 Triangulation Definition

Consider a set of 3D features lying on the surface of an object which we would like to triangulate. For simplicity we also assume that the object has no holes, and thus since it is topologically a sphere or a plane segment, it can be represented by a planar graph $\mathcal{T}_i$ with vertex set $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ corresponding to n feature points and edge set $\{\mathbf{e}_{ij}, \ldots\}$ where edge $\mathbf{e}_{ij}$ joins vertices $\mathbf{v}_i$ and $\mathbf{v}_j$.

Figure 4 (a) illustrates a triangulation on five vertices. The faces in a triangulation all have three edges, except for possibly the one external unbounded face that can have more than 3 edges. The three-edged faces can be drawn as straight-line triangles and are denoted as as $\mathbf{f}_{ijk}$ where vertices $\mathbf{v}_i$, $\mathbf{v}_j$ and $\mathbf{v}_k$ and their joining edges form the face boundary. A triangulation boundary, B, is a circuit on a specified subset of the vertices and edges that can partition the graph into two regions of the plane, one of which is a single face and the other contains all the remaining edges and vertices. To reduce the search space, we assume the boundary on the vertex set is known and for example may correspond to the convex hull of the features in one of the images. Surfaces that are topologically spherical do not need a boundary. Finally we let all triangulations with the same boundary, B, define a boundary class. Assuming we know the surface boundary, our search space consists of all the triangles of this boundary class.

3.2 Edge Swapping

Searching over the triangulation space necessitates that we have a method for finding new triangulations. We propose using the *swap* operator to generate new triangulations from a current triangulation.

The boundaries of two adjacent triangular faces share a common edge and two common vertices. Performing a *swap* on this edge deletes the edge and creates a new edge joining the two vertices of the faces that are not shared, as illustrated in Figure 4. The Figure also illustrates some edges that cannot be swapped, such as edge $\mathbf{e}_{15}$ in (d) and the boundary edges. Unlike 2D and 2.5D applications [12, 9], we permit a swap on an edge even when the two adjacent triangular faces form a non-convex quadrilateral, as in edge $\mathbf{e}_{45}$ of Figure 4(a). We permit this because the particular embedding in the plane of the resulting triangulation is unimportant and will be different for each image. All that is required is that planarity is maintained, and hence that there exists a re-embedding of Figure 4(c) such that there are no intersections of edges, as in Figure 4(d).

We will use our swap operator to search through the space of triangulations, and we would like guarantees both that the swap operator always maintains valid triangulations and also that there are no triangulations in the search space that cannot be reached by edge swaps. It can easily be confirmed that a swap operator always takes a triangulation of one boundary class to a valid triangulation of the same boundary class by showing that the conditions for a triangulation are always met after an edge swap. To show the generality of swap operators we define an equivalence relationship between triangulations of the same boundary class to be: $\mathcal{T}_A \equiv \mathcal{T}_B$ if and only if T_A can be transformed into $\mathcal{T}_B$ by a sequence of swaps. Then we state the following theorem:

Theorem 3.1. *All triangulations of a given boundary class are equivalent under edge swaps.*

This theorem is proved for full triangulations and then extended to boundary cases by Ore [13] page 9. This theorem guarantees that if there is a true triangulation, $\mathcal{T}_*$, then by by using edge swaps we can exhaustively search through the space of triangulations until we find it. It does not, however, preclude an optimization algorithm from being trapped by local minima.

3.3 Finding the Best Triangulation

A triangulation defines a cost or variance Σ from equation (6). Assume we start with some initial triangulation of a vertex set. We are then free to devise various search methods on the space of triangulations to find the triangulation with minimum cost. We formulate a greedy search as follows:

1. Start with an initial triangulation and find the cost of all the faces, the sum of which equals the cost of the triangulation, Σ in equation (6). For each edge that can be swapped find the cost difference between the triangulation before it is swapped, and the triangulation after it is swapped.
2. If there are no swaps that reduce the cost of the triangulation then terminate.
3. Swap the edge that produces the greatest reduction in cost, Σ.
4. Update the costs of all faces as well as the change in cost due to each potential swap.
5. Return to step 2.

In regions of little or no texture, swapping edges may produce complicated overlapping surfaces. These surfaces include hidden faces which increase surface complexity without helping to explain the images. In order to bias the algorithm to simpler surfaces we assume that each triangle of the 3D surface is visible in at least one image. Thus we exclude potential swaps that produce completely hidden faces.

This algorithm converges to a local minimum. Convergence to the true or a “good” surface depends on the starting triangulation and the local costs of swaps. Empirically we found very good convergence for convex corners and regions. Concave regions are harder and occlusions are more likely, and so the chance of being trapped by local minima in the cost is also higher. For the complexity of objects shown in this paper, however, the convergence was good.

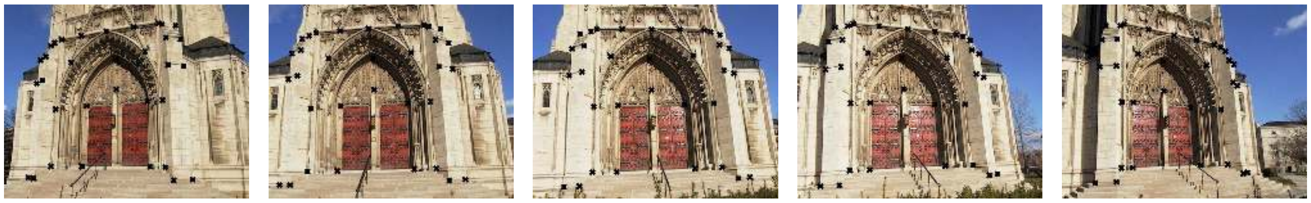
Figure 5: Image sequence of church door with features obtained by tracking by hand and then using a SFM algorithm to recover shape and better feature position estimates. Notice the significant varying specular reflection on the door. Although this effect is unmodeled and creates a larger re-projection error as shown in Figure 7, it does so for all the triangulations and the method remains robust to this.

3.4 Method Simplification

There are a number of factors that significantly effect the efficiency of the algorithm. First we reduce the size of our images by a quarter in each dimension using Gaussian pyramid sub-sampling, and hence reduce the number of pixel calculations by 1/16. If swapping an edge between two triangles does not change the visible area of the two triangles in an image, then the change in cost from that image is only due to the two triangles adjacent to the edge. This is the case when the triangles form a convex quadrilateral in the image. There are cases, however, when swapping an edge reveals portions of triangles that were previously occluded or else covers over previously visible triangles. In these cases the newly revealed or occluded faces must have their costs updated. A fast approximate way of achieving this which we implemented is to add the average variance of a pixel to the cost for each pixel revealed and subtract this for each pixel covered over, rather than explicitly calculating the new costs for each of these faces.

Another simplification we made in our implementation was in the calculation of the re-projected images, $\hat{\mathcal{I}}$, in equation (1). Instead of perspectively projecting the images onto the surface and re-projecting them back, we used affine warping of the triangles. For face f in image $\mathcal{I}_j$, we obtain an estimate of each for its pixels as:

$$\hat{\mathcal{I}}_{fj}(\mathbf{x}_j) = \sum_i \frac{\mathcal{I}_{fi}(\mathbf{x}_i) V_{fi}(\mathbf{x}_i) W_{fi}}{\sum_i V_{fi}(\mathbf{x}_i) W_{fi}}. \qquad (7)$$

Here $\mathbf{x}_i$ is the pixel position in image $\mathcal{I}_i$ that corresponds to pixel $\mathbf{x}_j$ in image $\mathcal{I}_j$. It is found by affinely warping the trianglular face f in image $\mathcal{I}_j$ onto the triangle in image $\mathcal{I}_i$. The visibility map, $V_{fi}(\mathbf{x}_i)$, is 1 if the pixel $\mathbf{x}_i$ on face f is visible and 0 if occluded. The weight, W_{fi}, given to a pixel is equal to the number of visible pixels in the triangle in that image. This averaging scheme gives head-on views of faces more weight than oblique views. It also provides less blurring than a two-step projection followed by inverse projection. The affine approximation itself is valid so long as any given triangle does not have significant perspective distortion.

3.5 Triangle Initialization

If not all the vertices are visible in any one image, it may be necessary to patch together a number of triangulations from different images. For points visible in one image we could start with a Delaunay triangulation of these. Alternatively we could select a set of images in which these points are all visible and greedily grow a triangulation as follows:

1. Start with a closed edge list, E, such as the convex hull, and a set of vertices on and internal to E.
2. Select one of the edges in E, perhaps the longest edge. Find all vertices on and internal to E that form triangles with this edge which do not contain any of the other vertices in any of the images and do not intersect any edges in E. Let S be this set of triangles.
3. Pairwise, compare re-projection costs in overlapping regions of triangles in S and eliminate triangles until the best triangle, T, remains.
4. Remove from E all edges also in T, and add to E the remaining edges in T.
5. If E still contains edges return to step 2.

In some cases, such as the cube in Figure 3, this growing method alone can obtain a correct triangulation. In other cases, applying edge swapping can improve this initial result, as this method does not use all the pixels in the images and does not handle occluded faces.

4 Results

We provide results of triangulation refinement on three real image sequences. First we show a cube in Figures 2 and 3. Features were selected and registered by hand on all seven images of a short sequence. A Structure from Motion algorithm was used to recover the 3D position of the features and the camera positions, orientations and focal length [10]. A standard

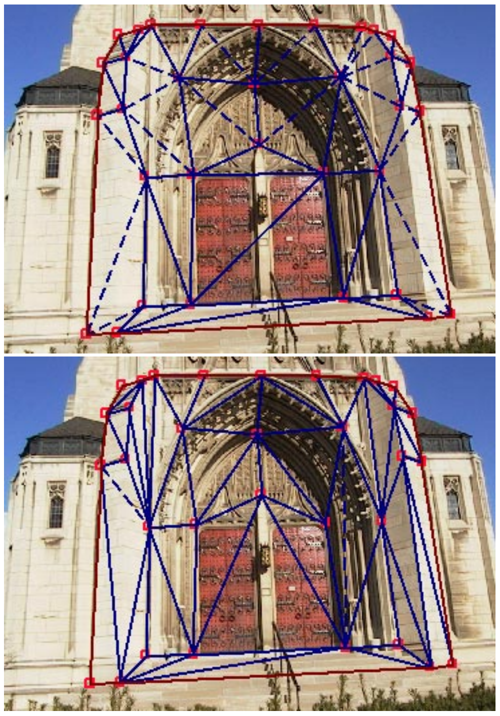

Figure 6: On top is a 2D Delaunay triangulation of the middle image in the sequence. The dashed lines show edges that result in triangles cutting through or lying above the surface. Below is the refined triangulation.

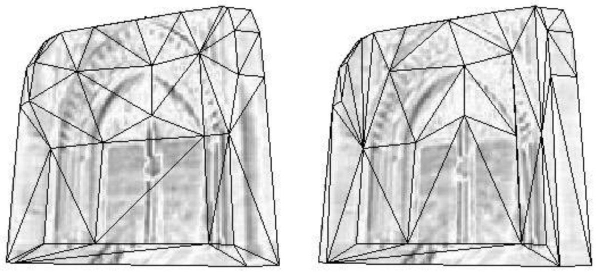

Figure 7: The re-projected error at each pixel in one of the images. On the left is the re-projection error assuming our initial surface triangulation, and on the right is the re-reprojection error with the final refined triangulation.

automated method for triangulating the points is to perform Delaunay triangulation in one of the images, and this was done in the middle image as is illustrated in Figure 2. A number of the faces intersect the true surface, and the resulting predicted surface does not correspond well to the image sequence. After running our refinement algorithm we obtain the triangulation and predicted surface shown in figure 3.

A similar experiment was performed on a sequence of five images of a church door shown in Figure 5 with a set of hand-tracked features. Structure from Motion was used to obtain the shape and camera motion estimates as well as to correct small errors in feature positions. A Delaunay triangulation was performed in the middle image as is illustrated in Figure 6. The feature points and triangle are shown, with the dashed edges corresponding to edges that force faces to cut through the true surface or lie above the true surface. The resulting re-projected error in one of the images is shown in Figure 7. Applying our refinement algorithm to this triangulation reduces this re-projected error and we obtain the triangulation shown in Figures 6 and 7. Only two of the updated edges cause poor faces. One of these is not swapped due to insufficient texture and the other because the surface is not planar. We could apply some texture enhancing operations to the images to overcome the first problem and selecting more points for the second. The resulting texture-mapped 3D surfaces are shown in Figure 8.

Finally we performed the same steps on a sequence of four images of a building. The Delaunay triangulation and our refined triangulation along with the corresponding texture mapped surfaces are shown in Figures 9 and 10. We manually selected the boundary.

We note that our sequences contained significant unmodeled specular reflections and changes in intensity between images, but these effects are similar across triangulations and so fall out in the noise.

5 Conclusion

We have presented an approach to obtaining surface models from 3D feature points that uses the images to guide surface selection. Our method compares the re-projected images of the predicted surface with the actual images and uses this measure to select between competing triangulations and so overcome the surface ambiguity problem. As a result we can capture sharp corners which other methods often fail to capture. We presented an algorithm for searching through the space of triangulations using edge swaps. We showed results of our algorithm refining typical triangulations and removing numerous surface artifacts by selecting a better triangulation. Our method takes us beyond shape recovery to achieving a surface description of an unknown object.

By using calibrated cameras and known 3D feature points, the effects of occlusions can be modeled and used to aid in surface selection. Since each triangulation that is considered must explain all of the image data, our algorithm is robust to significant levels of unmodeled effects such as specular reflection and non-planar surfaces. As long as these unmodeled effects produce similar cost increases in all the triangulations,

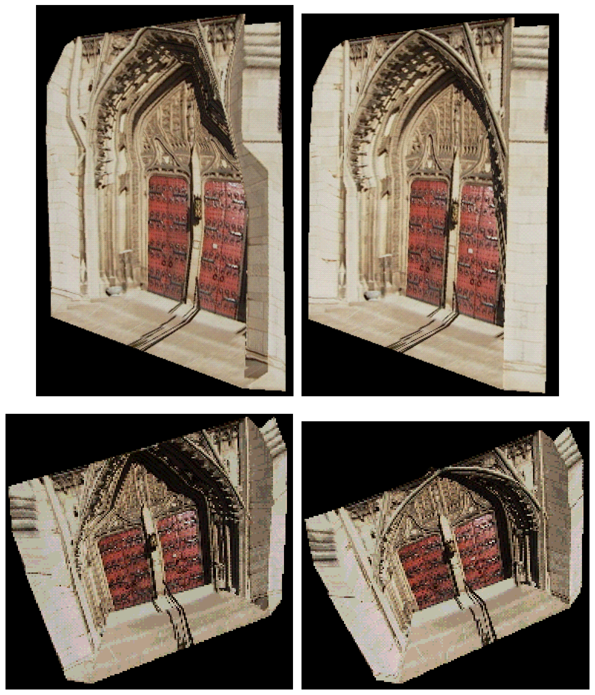

Figure 8: On the left are two views of the texture-mapped surface from the 2D Delaunay triangulation of Figure 6 showing the artifacts caused by incorrect triangles. On the right is the result with our refined triangulation shown in the lower part of Figure 6.

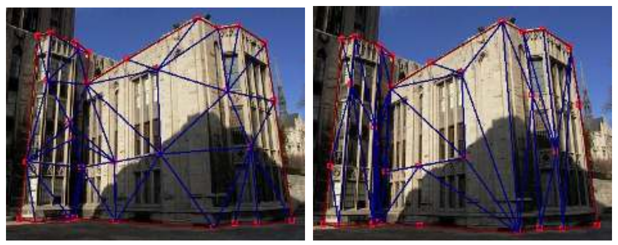

Figure 9: On the left is a 2D Delaunay triangulation and on the right is the refined triangulation.

they do not affect our choice of surface.

There are still limitations and further areas to pursue. We depend on a judicious selection of feature points on edges and corners. If, however, we permitted triangles to split and used other cues such as edges in the images, the algorithm might work with a more general distribution of features on the object surface. While in theory we can handle occlusions, they tend to create local minima in the cost. We plan to investigate other search methods including randomization to tackle these more complicated cases.

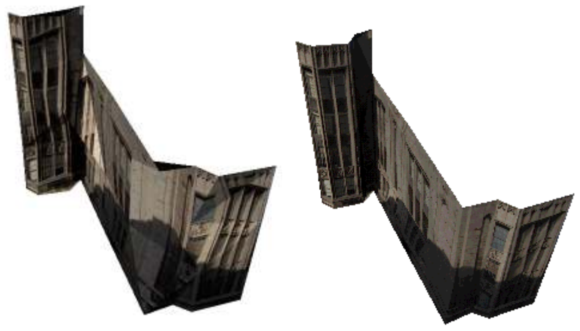

Figure 10: The triangulated surface created from a 2D Delaunay triangulation (left) and our refined triangulation (right).

References

[1] N. Amenta, M. Bern & M. Kamvysselis, A new voronoi-based surface reconstruction algorithm, *Comput. Graph.* (1998), **32**, 415–421, proc. SIGGRAPH '98.

[2] M. Bern, H. Edelsbrunner, D. Eppstein, S. Mitchell & T. S. Tan, Edge-insertion for optimal triangulations, *Proc. Latin Amer. Sympos. Theoret. Inf.*, vol. 583 of *Lect. Notes Comp. Sci.*, Springer-Verlag, Sao Paulo, 1992, 46–60.

[3] M. Bern, D. Eppstein & J. Gilbert, Provably good mesh generation, *J. Comput. Syst. Sci.* (1994), **48**, 384–409.

[4] F. J. Bossen & P. S. Heckbert, A pliant method for anisotropic mesh generation, *Proc. 5th Int. Meshing Roundtable*, Sandia Nat. Lab., NM, 1996, 63–74.

[5] P. E. Debevec, C. J. Taylor & J. Malik, Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach, *SIGGRAPH 96* Addison Wesley, 1996 11–20, New Orleans, 04-09 August 1996.

[6] O. Faugeras, M. Herbert, P. Mussi & J. Boissonnat, Polyhedral approximation of 3-d objects without holes, *CVGIP* (February 1984), **25**(2), 169–183.

[7] O. Faugeras, L. Robert, S. Laveau, G. Csurka, C. Zeller, C. Gauclin & I. Zoghlami, 3-d reconstruction of urban scenes from image sequences, *CVIU* (Mar. 1998), **69**(3), 292–309.

[8] P. Fua & Y. G. Leclerc, Object-centered surface reconstruction: Combining multi-image stereo and shading, *Int. J. Comp. Vision* (1995), **16**(1), 35–56.

[9] M. Garland & P. S. Heckbert, Fast polygonal approximation of terrains and height fields, *Report CMU-CS-95-181*, Carnegie Mellon University, (1995).

[10] A. Heyden & K. Astrom, Euclidean reconstruction from image sequences with varying and unknown focal length and principal point, *Proc. Comp. Vision Patt. Recog.*, Peurto Rico, 1997 438–443.

[11] K. N. Kutulakos & S. M. Seitz, A theory of shape by space carving, *Proc. Seventh Int. Conf. Comp. Vision*, Kerkyra, Greece, 1999 307–314.

[12] C. L. Lawson, Transforming triangulations, *Discrete Math.* (1972), **3**, 365–372.

[13] O. Ore, *The Four-Color Problem*, Academic Press, New York, NY, (1967).

[14] F. P. Preparata & M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, NY, (1985).