

Image Synthesis From A Sparse Set Of Views

Qian Chen and Gérard Medioni *

University of Southern California

Abstract

We present an image synthesis methodology and a system built around it. Given a sparse set of photographs taken from unknown viewpoints, the system generates images from new, different viewpoints with correct perspective, and handles occlusion. It achieves this without requiring any knowledge about the 3-D structure of the scene nor the intrinsic camera parameters. The photo-realistic rendering process is polygon based and can be potentially implemented as real time texture mapping. The system is robust to noise by taking advantage of duplicate information from multiple views. We present results on several example scenes.

Keywords: image-based rendering, epipolar geometry, projective invariant, homography, Constrained Delaunay Triangulation.

1 INTRODUCTION

Conventional VR authoring using solid modeling techniques is time consuming in both the modeling and rendering stages, and the final images look artificial. Recently, people have proposed to use photographs or image sequences directly, with the advantages of modeling in the image space, rendering speed independent of the scene complexity and producing photo-realistic images. While building 3-D models from images has been the theme of computer vision for over two decades, rendering using images has just started. In this paper, we present an image synthesis methodology and a system built around it. The system is purely image based, meaning that at no point during the process is the three dimensional Euclidean information explicitly recovered and/or used. We show that in terms of displaying images, almost everything can be accomplished in the image space, including the correct perspective and occlusion effects which are critical in 3-D perception. We define our problem as follows:

Given images of a static scene taken from different angles, how to reproject and integrate them into a new image as if it were obtained from a viewpoint that is different from any of the source views.

Two fundamental questions need to be addressed: how to warp existing frames, and how to resolve occlusion in the generated frame as a result of changing view point. We shall present algorithms to address both of them with the following assumptions:

- the object is polygonizable.
- reflections can be ignored.
- the camera lens distortions can be ignored.

In the following, we first survey related work in this area. We then introduce the computer vision tools to be used. We show a high

level flow-chart of our system afterwards. In the following section, we describe in detail the algorithms. Finally, we present our experimental results and conclude the paper with a brief discussion. In this paper, we use interchangeably the words image and view. For example, the given images are called *source views*, a synthesized image is called a *novel view*.

2 RELATED WORK

Existing approaches can be taxonomized into three categories: image based, reconstruction based and light field based.

Image based methods

One early paper in this area is [2], where the warping function is simply the scaled down optical flow vector and the visibility is resolved by using the depth value of each pixel which is cached when the frame is rendered. The image planes are parallel to each other and the virtual camera's movement is restricted to be parallel to the image planes. This restriction is relaxed by pre-warping the two source views to a common virtual plane [14]. This step simplifies problems, because warping then becomes the linear interpolation of the pre-warped views, and depth can be estimated from disparity. This method does not extend to multiple views, because the virtual plane is specified in terms of two views, and only two views. In addition, the virtual camera's movement is still limited in between the model views, a result of using linear interpolation. Arbitrary virtual camera placement is allowed in [10] by using projective methods. However, only two views are dealt with. Our work extends the method to multiple views and addresses occlusion. In [3], each location (a node) is associated with a cylindrical map of the environment which is obtained by rotating a camera and then stitching together the pictures. There is no view interpolation. The viewer has to jump to different nodes. Another pitfall is the limit on the vertical field of view. The same idea of cylindrical projection is used in [12]. But the latter allows reprojection to an arbitrary location. It extends the traditional structure-from-motion method to cylindrical projections. It simplifies the correspondence matching problem by manually picking "tie points". Then a non-linear process is carried out to find everything about the camera including its projection center, focus length and structural matrix. With these pieces of information at hand, reprojection is an easy task. During warping, pixels in the reference cylinders are traversed orderly so that the *epipoles* are visited last. This method suffers from the same drawback of limited elevation as [3]. All these image-based methods render the new image in a pixel-by-pixel fashion, not taking advantage of modern graphics hardware, which is mostly polygon based.

Reconstruction based methods

In [5] and [13], the warping and visibility become non-issues because a complete 3-D reconstruction of the scene is built. Source

* PHE 204, Institute for Robotics and Intelligent Systems
University of Southern California, Los Angeles, CA 90089
{chenqian, medioni}@iris.usc.edu
<http://roberts.usc.edu/~chenqian/research.html>

images are then “pasted” to the reconstructed model. In fact, a different terminology - structure from motion - is used in the computer vision community and the underlying problem has been studied from the beginning of the discipline but still there is no robust system. Accurate results are obtained in [5] by cleverly restricting its domain to simple geometric primitives such as box, prism and surfaces of revolution, each of which has inherent constraints. In [13], the scene is divided into voxels and the object boundaries are found by using the marching cube algorithm, which makes the method expensive.

Light field based methods

The main idea in [9] and [11] is borrowed from holography, which is to capture the wavefront of all rays that are emitted from an object. Since a large amount of sample images are needed to produce reasonable quality rendering, both papers deal with the image compression problem. This method is non-geometrical.

We distinguish our method from the above by not performing camera calibration or 3-D reconstruction. Furthermore, it has several other characteristics (which will be detailed later) which sets it apart from previous work:

- It imposes no restrictions on the virtual camera movement by using the *projective depth* concept.
- It takes multiple source views at sparse angles.
- It uses triangles rather than pixels as rendering primitives.
- It handles occlusion by mapping the triangles in a *visibility compatible* order.
- It is a geometrical method, therefore uses relatively a fewer number of images.

In essence, the system works as follows: For each feature point in the source images, a scalar value called *projective depth* is first extracted. It is similar to the disparity in stereo vision but is a projective invariant. It allows us to transfer any feature point to a new view. The images are divided into triangles using *Constrained Delaunay Triangulation* [1, 6, 8]. The triangles perform similar roles as the blocks in [2] in the sense that they carry pixels from the source views to the destination view. But in our case, each triangle corresponds to a planar facet of the real object, hence each pixel inside it can be warped by a 2-D *homography*. Partial occlusion is handled by drawing the triangles in a certain order, similar to the painter’s algorithm, but performed in the image space. Total occlusion is handled by checking the orientation of the polygons. In our setting, correlation based algorithms would be hard to implement as it is difficult to track features across a sparse set of views.

3 COMPUTER VISION TOOLS

3.1 Epipolar Geometry

As stated in [8], given two images of a static scene, if the only information we have about the scene is point correspondence and we have no knowledge about the camera such as focus length, optical center, etc., the strongest constraint we can obtain that relates these two images is the so-called *epipolar geometry*. This is described by the following equation:

$$p_2^T F p_1 = 0 \quad (1)$$

where F is 3x3 and is called the *fundamental matrix*, p_1 and p_2 are images of a common 3-D point and are expressed in their homogeneous coordinates. This equation says that p_2 is on the line defined

by Fp_1 which is called the *epipolar line* corresponding to p_1 . All epipolar lines on an image plane intersect at a point called the *epipole*. When the two planes are coincident, all the epipolar lines are parallel. Said in another way, the epipoles are at the infinity. Fortunately, by using projective geometry, this situation does not have to be treated differently. The epipolar geometry is graphically depicted in Figure 1.

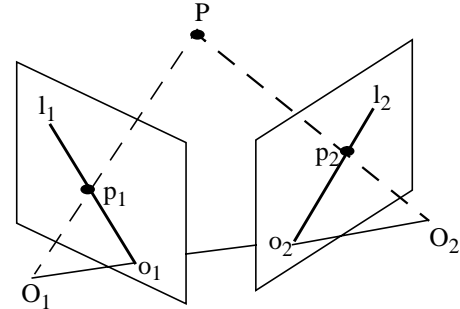


Figure 1: Epipolar geometry.

P - a 3D point; p_1, p_2 - P 's projections; O_1, O_2 - camera centers; o_1, o_2 - epipoles; l_1, l_2 - epipolar lines. When the two image planes are coincident, l_1 and l_2 are parallel, and o_1 and o_2 are at right and left infinity respectively.

In (1), F is defined up to a scale factor. So it has at most 8 independent coefficients. Actually, it is also known that F is of rank 2. Therefore, generally, 7 point correspondences are required to find it. If 4 points are co-planar, then 6 points are sufficient. The recovery of epipolar geometry is a well studied problem in computer vision. For a good overview and state of the art, the readers are referred to [16].

3.2 2-D Projective Transformation

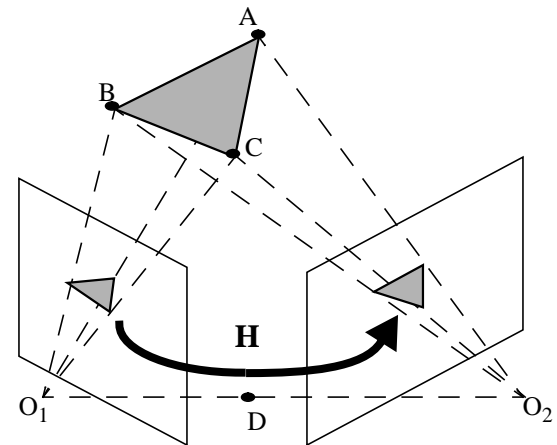


Figure 2: Homography between two images planes. Images of a planar facet are related by a 2-D homography. D is the intersection of ABC and the segment O_1O_2 . The images of D happen to be the epipoles.

As seen in Figure 2, a plane in space introduces a 2-D homography H between the two image planes. H is a 3x3 non-singular matrix subject to a scale factor, hence has only 8 independent coefficients. 4 point correspondences suffice to define the solution. Please be reminded that to compute H , all we need are point correspondences in the source and destination image planes. We do not

require any knowledge about the 3-D points which determine the plane in the 3-D Euclidean space. This implies that, given a set of four matching points, we know how to transfer points inside the trapezoids defined by each of the four point set. Can we transfer points between triangles? The answer is YES, only in this case the fourth pair of matching points is supplied by the epipoles which requires the epipolar geometry to be recovered a priori. The triangle determines a plane which, in projective space, always intersects the line connecting the two camera centers. The epipoles are the images of this intersection point (D in Figure 2). Let $p_{ij} = [u_{ij}, v_{ij}, t_{ij}]^T$, $i=1,2$ and $j=1,2,3,4$ be four point pairs where t_{ij} is 0 if p_{ij} is an epipole at infinity or 1 otherwise. By definition, we have

$$\alpha_j \begin{bmatrix} u_{2j} \\ v_{2j} \\ t_{2j} \end{bmatrix} = H \begin{bmatrix} u_{1j} \\ v_{1j} \\ t_{1j} \end{bmatrix}, j = 1, 2, 3, 4 \quad (2)$$

where α_j is the scale factor. In (2), there are $2 \times 4 + 4 = 12$ unknowns and 12 linear equations, so H is readily solvable.

3.3 Projective Depth

Photogrammetrist established a long time ago that photos could be transferred without extracting the 3-D Euclidean information about the scene. For instance, in Figure 3, the projection of a 3-D point in the central image is the intersection of the epipolar lines corresponding to the point's projections in the left and right images. In the parallel case, however, the intersection cannot be found because the epipolar lines are parallel too.

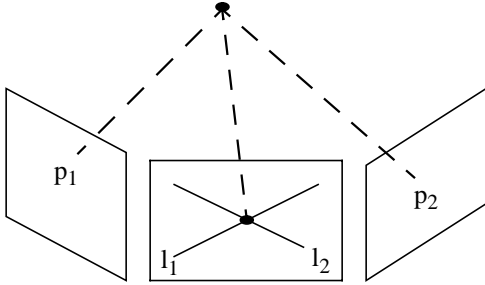


Figure 3: Image transfer.

The correspondent position in the central view is the intersection of the epipolar lines.

Shashua [15] proposed to use the cross-ratio, a projective invariant, to get around this. (We have mentioned that by going to projective space, parallelism does not warrant any special treatment). He named the quantity *projective depth*. In Figure 4, A, B, C, D are four non-coplanar points in the scene. O_1 and O_2 are the camera projection centers. o_1 and o_2 are the epipoles. P is a scene point whose projective depth is to be computed and whose images are p_1 and p_2 . Let the ray O_1P intersect two hypothetical triangular facets ABC and ACD at I_1 and I_2 respectively, then P's projective depth is defined as the cross ratio of O_1, I_1, I_2 and P and is denoted by

$$D(P) = Cross(O_1, I_1, I_2, P) \quad (3)$$

In fact, we do not know the coordinates of any of these points.

However, if we look at their projections on the image plane of O_2 , we immediately see that o_2 and p_2 are already known. Since the cross ratio is preserved under perspective projection, all we need to recover are the image coordinates of I_1 and I_2 in O_2 's image plane (i_1 and i_2 respectively). From the previous section, we know that they are actually the projections of p_1 under the homographies induced by the plane ABC and ACD respectively which is recovered from the images of A, B, C and D. If we denote them by H_1 and H_2 , then P's projective depth can be calculated by

$$D(P) = Cross(o_2, H_1(p_1), H_2(p_1), p_2) \quad (4)$$

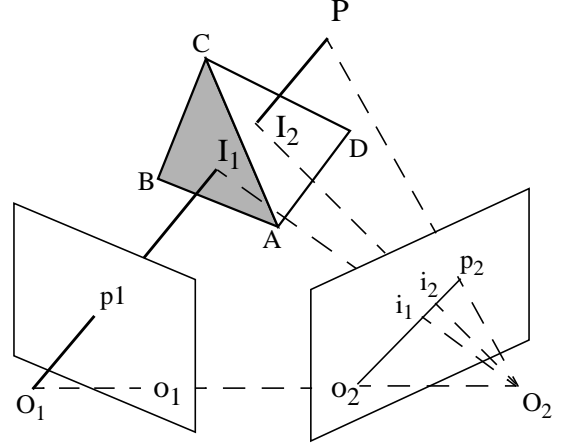


Figure 4: Image warping based on projective depth.

Now suppose O_2 is the new camera position. If o_2 and the projections of A, B, C, D in the new image plane are all known, i_1 and i_2 can be computed the same way as before. Then from P's projective depth, p_2 can be recovered. What this algorithm says is that if enough point correspondences among three views have been established so that the epipolar geometry is computable, then images from two of them can be transferred to the third one. For the convenience of reference, we will call in our paper the simplex ABCD a *projection basis* to denote the fact that it is used in a projective sense and to be different from the common definition of projective basis. It should be pointed out that the projection basis is implicitly specified since only the images of the basis points are known.

4 SYSTEM

Shown in Figure 5 is a high level flow-chart of the system. Currently, the low level work is performed manually assisted by the computer. In the future, we want to reverse the relationship - have them done semi-automatically, assisted by a human.

Stage1 - Corner extraction and matching

Corner points are extracted and matched manually. Epipolar geometry is then computed between each pair of points. The average distance of each point to its correspondent epipolar line is displayed. This helps the operator to refine those with large errors.

Stage2 - Edge detection and labeling

Again, in our current implementation, this step is done manually, by simply connecting points. In the future, this will be replaced by human assisted edge detector which outputs edges as NURBS. We choose to not use fully automatic edge detection algorithms

because, in most cases, they do not produce results that can be used without a lot of cleanup work. An example of using the Canny edge detector is shown in Figure 6. The original image is Figure 14(a).

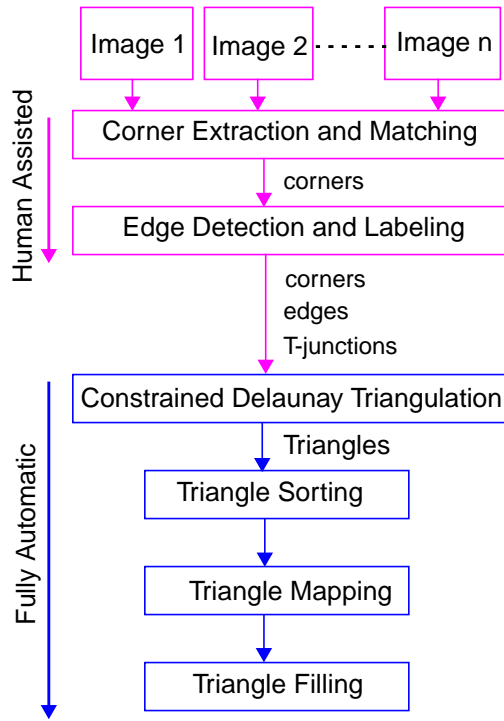


Figure 5: System flow-chart.

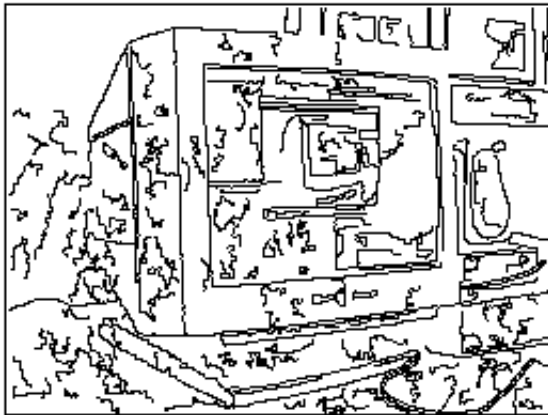


Figure 6: Result of using Canny edge detector.

Stage3 - Constrained Delaunay Triangulation (CDT)

A *Constrained Delaunay Triangulation* is then conducted in each source image to divide the scene into triangles. Later each triangle will be warped to the novel view using the projective depths associated with its vertices. A CDT is a triangulation that preserves the existing edges [1, 6, 8]. Strictly speaking, a CDT is not a Delaunay triangulation - in order to keep the edges, not all triangles are optimal. There are two reasons to perform the triangulation. Firstly, this avoids identifying and matching faces, another tedious and not-so-easy task. Secondly, this establishes the potential to implement the warping as a texture mapping process and to make use of hardware capabilities.

Stage4 - Triangle sorting

As will be detailed in 5.3, triangles have to be filled in a certain order so that occlusion is handled correctly in the novel view.

Stage5 - Triangle mapping

Triangle vertices are mapped to the novel view using the projective depth. Three problems are faced: i) uncertainties associated with the choice of the projection basis. ii) partial occlusion resulting in the formation of T-junctions. One example is given in Figure 7; iii) total occlusion - a face disappears in the new view as a result of changing view point.

Stage6 - Triangle filling

Now that all the vertices are at the right positions (in the novel view), pixels inside them can be filled by warping from source images using a 2-D homography.

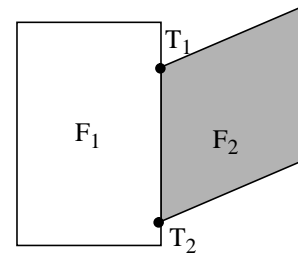


Figure 7: Partial occlusion and T-junction. F_2 is partially occluded by F_1 . T_1 and T_2 are T-junctions.

5 ALGORITHMS

5.1 Image Warping

We have stated in the previous section that the warping process consists of two steps: the mapping of the triangles and the filling of pixels within each triangle. For the first purpose, we use the projective depth concept introduced in 3.3. For the second, the projection matrices are computed using (2). Thanks to projective geometry, we are able to place the virtual camera arbitrarily and still maintain correct perspective effect in the synthesized images. Understandably, this warping implementation is in fact a texture mapping process, only that the mapping function is defined by a projective transformation rather than an affine transformation as provided by most commercial graphics packages. Had it been implemented in hardware, our algorithm should be able to work in real time.

5.2 Dealing With Uncertainties

The accuracy of the recovered projective depth largely depends on that of the chosen basis. We measure the quality of a basis by the average distance of the four points to their corresponding epipolar lines across all views. Among the valid candidates*, we choose the one with the least amount of error. We also take advantage of duplicate information resulted from multiple source views: each point has several estimated mappings, one from each pair of views. Outliers are rejected by thresholding on the variance. The final coordinate of a point is a weighted average of the remaining estimates where the weight is proportional to the inverse of variance.

* The degenerate case happens if a point resides on one of the four facets of the base simplex thus its projective depth is undefined.

Figure 8 shows an example result. The vertices of the irregular polygon around a corner are the candidate points for that corner and their distribution indicates the uncertainty. The outliers have already been removed. The upper-right point comes from two views, and therefore only has one candidate.

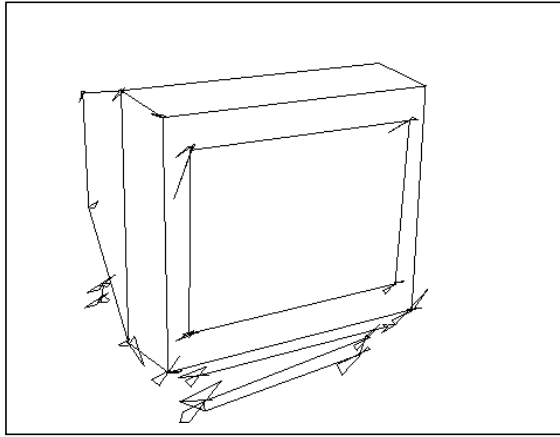


Figure 8: Display of uncertainties.

5.3 Partial Occlusion

As depicted in Figure 7, partial occlusion is characterized by the appearance of T-junctions. A T-junction can be easily identified by checking whether its match is on the corresponding epipolar line. The difficulty arises, however, when trying to recover its projective depth because its peer is not the real correspondent point in the sense that they are not the images of a common 3-D point. This implies that such a point cannot be reprojected directly. We propose our solution in two steps, referring to Figure 9.

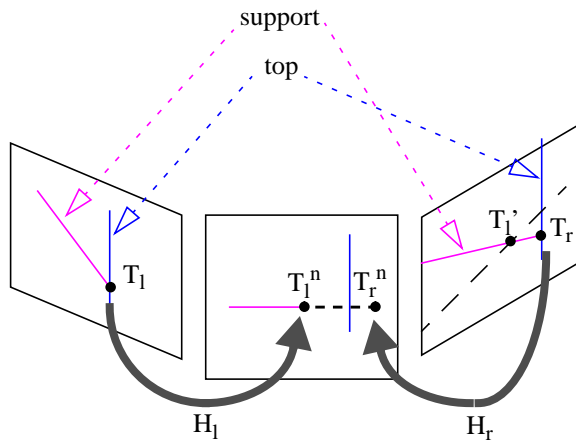


Figure 9: Mapping of T-junctions.

T_1^n forms a gap with the top, while T_r^n goes across it.

5.3.1 Recovery of Projective Depth For T-junctions

Let us name the two edges forming a junction *top* and *support* as labeled in the figure. Let us denote the left junction as T_L and the right one T_R . They were referred to earlier as peer junctions. Let us further denote the correspondent point of T_L in the right image as T_L^n and that of T_R in the left image as T_R^n . Since T_L^n also locates on the epipolar line of T_L in the right image plane, T_L^n is the intersec-

tion of the right support and the epipolar line. Now that we have the pair (T_L, T_L^n) , we can compute the projective depth for T_L and reproject it onto the new view which we denote as T_1^n . If the new view is between the two existing views, there is a gap between T_1^n and the top in the new view. The same thing is done to T_R . But the projected support intersects the top forming an overlap. Consequently, when warping facets from the right view, it must be done in a certain order.

5.3.2 Ordering Of Triangular Facets

If the Euclidean information were available, the order could be easily determined by comparing their depth values. However, what we have is the projective depth, which is a cross ratio, and does not carry any metric information. So how can this be done? We know that occlusion is naturally resolved in all source images. The question then becomes whether the order information is somehow encoded in these images, and, if yes, how to extract it. The answer again lies in the epipolar geometry. Actually, we have seen it in Figure 4: I_1, I_2, P are all projected to the same pixel p_1 in O_1 , but their depth order is preserved in the image plane of O_2 . To see why this is generally true, notice that if I_1 occludes I_2 , then the sequence $O_1-I_1-I_2$ is preserved under any perspective projection as long as the line O_1I_1 (or O_1I_2) is not projected into a point.

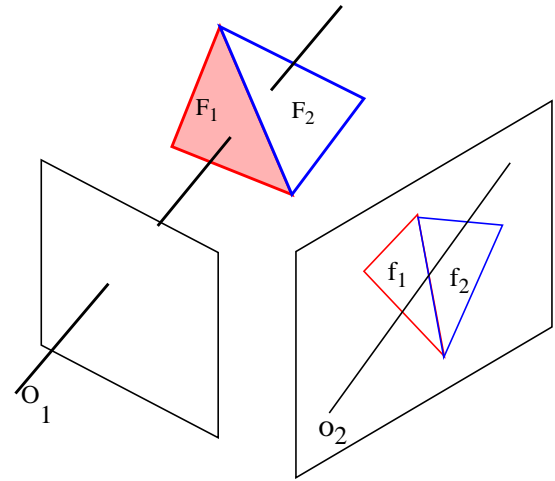


Figure 10: Visibility compatible order.

In Figure 10, F_1 and F_2 are 3-D facets, f_1 and f_2 are their projections on the right image plane. The darkened line emitting from O_1 is an optical ray. The plain line starting from O_2 is its projection in the right image plane. It is also an epipolar line. If F_1 occludes F_2 , there exists at least one point on F_1 which occludes a point on F_2 but not vice versa (otherwise they would intersect resulting an edge which separates each of them into two smaller facets). Consequently, if in the right image plane a point which belongs to f_1 can be found is closer to the epipole than a point belongs to f_2 , it can be concluded that f_1 should be mapped after f_2 . All triangles are sorted based on this relationship and are mapped accordingly. We say the order determined this way is *visibility compatible*, the same terminology used in [12]. Notice that it is only valid with respect to the given views. For a different view, the epipole is changed, thus the order. However, a perturbation to the configuration is merely a perturbation to the existing order which can be quickly rearranged.

This sorting algorithm is therefore very efficient for walk-through type of applications. In implementation, we only compare triangles which share a vertex or an edge. If two triangles share an edge, we compare the opposite vertices. If they share only a vertex, the remaining vertices are checked.

5.4 Total Occlusion

Total occlusion happens when a front facet in a source view goes to the back in the destination view and is completely occluded by other facets. Actually the previous ordering algorithm is also applicable here. The occluded facet is guaranteed to be at the beginning of the display list. However, since totally occluded facets are not observable in the new view anyway, we want to be able to mark them so that they will be ignored during warping. It turns out that these facets can be easily identified: when projected to the new view, the orientation of their boundaries is reversed. We can assign an arbitrary vertex sequence (clockwise or non-clockwise) to each facet, and check if it is preserved after the projection. If it is not, then we know the facet is at the back when looking from the new camera position. As a final note, we would like to mention that the same observation has been made in [4] where the total occlusion is further categorized into orientation-discontinuity occlusion and limb occlusion.

5.5 Summary

We now summarize our algorithm into the following steps:

1. Tessellate each source image.
2. Determine the correct warping sequence with respect to the current new view.
3. Recover projective depth for all vertices including T-junctions.
4. Project all vertices to the new view and remove outliers.
5. Mark those totally occluded facets.
6. Warp all unmarked facets from the source images.

6. RESULTS

Test results are shown in the color plates. To specify the novel view, we need some seed points, at least four of them not co-planar to form the projection basis and a large enough number so that the epipolar geometry can be estimated. We have tried three methods: 1) interpolating points from the rectified views using the method of [14]; 2) interpolating (extrapolating) points directly from source views; 3) picking points from a real view. Figure 11 is an example involving two source views (a) and (b). (c), (d) depict the corners, edges and T-junctions (circled in red). (e) is the synthesized image. The seed points are obtained using 1). Therefore (e) is physically valid. If (a) or (b) were warped individually, there would be gaps (magenta) and overlaps (blue) as shown in (f) and (g). Figure 12 shows four frames extracted from an mpeg movie produced by interpolating three views. The seed points are obtained using 2). Although the frames are not physically valid, they do not show any abnormality visually. Figure 13 is an example of extrapolation. In Figure 14, (d) is generated from (a), (b) and (c). The top face is missing because it does not exist in any of the source views. (g) and (h) are synthesized from (e) and (f). Both miss the front thin face of the terminal support. (g) shows the correct handling of total occlusion. But the shape of the screen is not well preserved. This is because at the current stage we do not handle curved edges. Using all five source views, the resultant view (j) includes all visible faces. Seed points are picked from (i). Therefore (d), (h) and (j) are synthesized copies of (i).

7. DISCUSSION AND FUTURE WORK

In section 2, we taxonomized the existing approaches into three categories: image based, reconstruction based and light field based. Among them, the third category is non-geometrical and, from our perspective, deals with quite different issues. For the first two, the second one is harder. So either the domain is restricted [5] or the method is expensive [13]. Among the approaches in the first category where ours belongs to, there are two subcategories based on whether or not calibration is used. In the extreme case where the environment is synthetic [2], the camera can be thought of as being calibrated exactly and the depth information is perfectly known. In a real environment, camera calibration is a non-linear and unstable process and/or is subject to the range and pattern of the calibration object used. On the other hand, while keeping every thing in the image space avoids the process, it has the shortcoming of not allowing the user to specify the exact viewing parameters. For instance, both [14] and ours can generate a fly-by of a sequence of views, but they cannot generate a view at a specific location and viewing direction. To be able to do this, there is no way but to recover some Euclidean information. Minimally, the camera's projection center and focus length should be recovered. Fortunately there are ways [8] to achieve this without pre-calibrating the camera. We are currently exploring this direction. We are also investigating human assisted edge detection methods which can output edges as NURBS curves. We hope that accomplishing these will allow us to model more complex scenes.

Acknowledgments

Funding was provided in part by a NSF grant to the Integrated Media Systems Center(IMSC) of the University of Southern California. The authors thank Dr. Zhengyou Zhang of INRIA, France, for making the FMatrix software available.

References

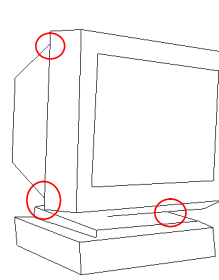
- [1] T. J. Baker. Automatic Mesh Generation for Complex Three Dimensional Regions Using a Constrained Delaunay Triangulation. *Engineering with Computers*, Vol. 5, 1989, pp. 161-175.
- [2] S. Eric Chen and Lance Williams. View Interpolation for Image Synthesis. *Proc. SIGGRAPH 93*, pp. 279-288, ACM SIGGRAPH, 1993.
- [3] S. Eric Chen. QuickTime VR - An Image-Based Approach to Virtual Environment Navigation. *Proc. SIGGRAPH 95*, pp. 29-38, ACM SIGGRAPH, 1995.
- [4] R. C-K. Chung, and R. Nevatia. Recovering LSHGCs and SHGCs from Stereo, *International Journal of Computer Vision*, Vol.20, No. 1/2, 1996, pp. 43-58.
- [5] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and Rendering Architecture from Photographs. *Proc. SIGGRAPH 96*, pp. 11-20, ACM SIGGRAPH, 1996.
- [6] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. The MIT Press, 1993.
- [7] O. Faugeras, S. Laveau, L. Robert. 3-D Reconstruction of Urban Scenes from Sequences of Images. *Research Report 2572*, INRIA Sophia-Antipolis, June 1995.

- [8] L. De Floriani. An On-Line Algorithm for Constrained Delaunay Triangulation. *CVGIP: Graphical Models and Image Processing*, Vol. 54, No. 3, July, 1992, pp. 290-300.
- [9] S. J. Gortler, R. Grzeszczuk, R. Szeliski, M. F. Cohen. Lumi-graph. *Proc. SIGGRAPH 96*, pp. 43-54, ACM SIGGRAPH, 1996.
- [10] P. Havaldar, M. Lee, and G. Medioni. View Synthesis from Unregistered 2-D Images. *Proc. Graphics Interface '96*, pp. 61-69, Toronto, Ontario, Canada, May 1996.
- [11] M. Levoy and P. Hanrahan. Light Field Rendering. *Proc. SIGGRAPH 96*, pp. 31-42, ACM SIGGRAPH, 1996.
- [12] L. McMillan and G. Bishop. Plenoptic Modeling: An Image-Based Rendering System. *Proc. SIGGRAPH 95*, pp. 39-46, ACM SIGGRAPH, 1995.
- [13] S. Moezzi, A. Katkere, D. Y. Kuramura, and R. Jain. Reality Modeling and Visualization from Multiple Video Sequences. *IEEE Computer Graphics & Applications*, Nov. 1996, pp. 58-63.
- [14] S. M. Seitz and C. R. Dyer. View Morphing. *Proc. SIGGRAPH 96*, pp. 21-30, ACM SIGGRAPH, 1996.
- [15] A. Shashua. Projective Depth: A Geometric Invariant for 3D Reconstruction from Two Perspective/Orthographic Views and for Visual Recognition, *Proc. International Conference on Computer Vision*, May 1993, pp. 583-590.
- [16] Z. Zhang. A New Multistage Approach to Motion and Structure Estimation: From Essential Parameters to Euclidean Motion Via Fundamental Matrix. *Research Report 2910*, INRIA Sophia-Antipolis, June 1996.

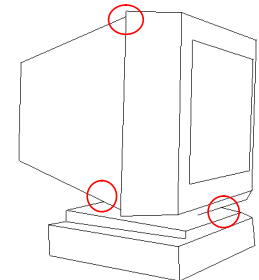


(a) view 1

(b) view 2



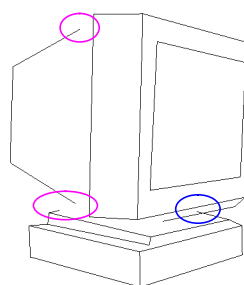
(c)



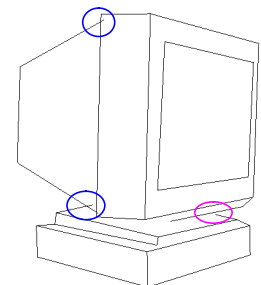
(d)



(e)



(f)



(g)

Figure 11: Handling occlusion



Figure 12: Synthesized views of a monitor

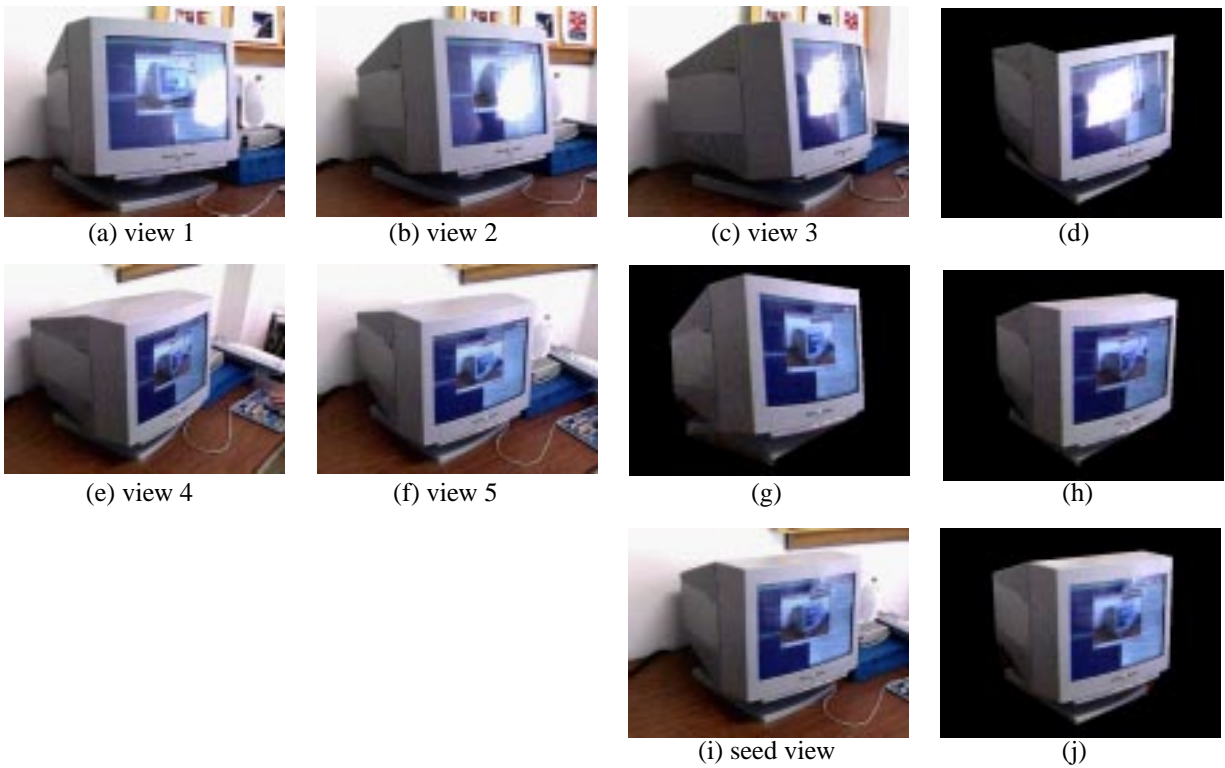


(a) view 1

(b) view 2

(c) synthesized view

Figure 13: Looking into a room



(a) view 1

(b) view 2

(c) view 3

(d)

(e) view 4

(f) view 5

(g)

(h)

(i) seed view

(j)

Figure 14: Fuller synthesis from more than two views