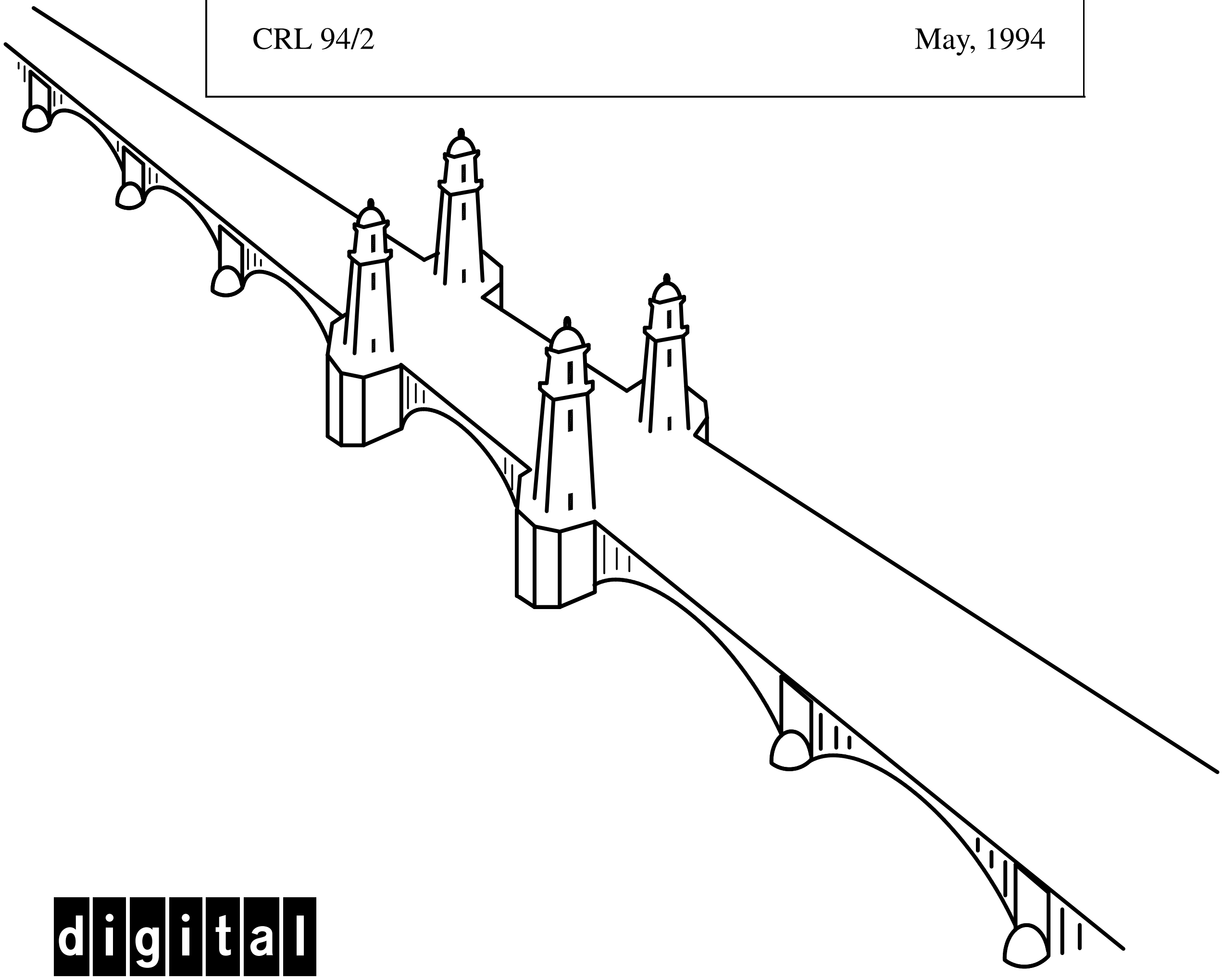
Image Mosaicing for Tele-Reality Applications

Richard Szeliski
Digital Equipment Corporation
Cambridge Research Lab

CRL 94/2 May, 1994



CAMBRIDGE RESEARCH LABORATORY
Technical Report Series

Digital Equipment Corporation has four research facilities: the Systems Research Center and the Western Research Laboratory, both in Palo Alto, California; the Paris Research Laboratory, in Paris; and the Cambridge Research Laboratory, in Cambridge, Massachusetts.

The Cambridge laboratory became operational in 1988 and is located at One Kendall Square, near MIT. CRL engages in computing research to extend the state of the computing art in areas likely to be important to Digital and its customers in future years. CRL's main focus is applications technology; that is, the creation of knowledge and tools useful for the preparation of important classes of applications.

CRL Technical Reports can be ordered by electronic mail. To receive instructions, send a message to one of the following addresses, with the word **help** in the Subject line:

On Digital's EASYnet:	CRL::TECHREPORTS
On the Internet:	techreports@crl.dec.com

Cambridge Research Laboratory
One Kendall Square
Cambridge, Massachusetts 02139

Image Mosaicing for Tele-Reality Applications

Richard Szeliski

Digital Equipment Corporation

Cambridge Research Lab

CRL 94/2 May, 1994

Abstract

While a large number of virtual reality applications, such as fluid flow analysis and molecular modeling, deal with simulated data, many newer applications attempt to recreate true reality as convincingly as possible. Building detailed models for such applications, which we call *tele-reality*, is a major bottleneck holding back their deployment. In this paper, we present techniques for automatically deriving realistic 2-D scenes and 3-D texture-mapped models from video sequences, which can help overcome this bottleneck. The fundamental technique we use is *image mosaicing*, i.e., the automatic alignment of multiple images into larger aggregates which are then used to represent portions of a 3-D scene. We begin with the easiest problems, those of flat scene and panoramic scene mosaicing, and progress to more complicated scenes, culminating in full 3-D models. We also present a number of novel applications based on tele-reality technology.

Keywords: image mosaics, image registration, image compositing, planar scenes, panoramic scenes, 3-D scene recovery, motion estimation, structure from motion, virtual reality, tele-reality.

Contents

List of Figures

1 Introduction

Virtual reality is currently creating a lot of excitement and interest in the computer graphics community. Typical virtual reality systems use immersive technologies such as head-mounted or stereo displays and data gloves. The intent is to convince users that they are interacting with an alternate physical world, and also often to give insight into computer simulations, e.g., for fluid flow analysis or molecular modeling [Earnshaw *et al.*, 1993]. Realism and speed of rendering are therefore important issues.

Another class of virtual reality applications attempts to recreate true reality as convincingly as possible. Examples of such applications include flight simulators (which were among the earliest uses of virtual reality), various interactive multi-player games, and medical simulators, and visualization tools. Since the reality being recreated is usually distant, we use the term *tele-reality* in this paper to refer to such virtual reality scenarios based on real imagery.[1] Tele-reality applications which could be built using the techniques described in this paper include scanning a whiteboard in your office to obtain a high-resolution image, walkthroughs of existing buildings for re-modeling or selling, participating in a virtual classroom, and browsing through your local supermarket aisles from home (see Section 8.)

While most focus in virtual reality today is on input/output devices and the quality and speed of rendering, perhaps the biggest bottleneck standing in the way of widespread tele-reality applications is the slow and tedious model-building process [Adam, 1993]. This also affects many other areas of computer graphics, such as computer animation, special effects, and CAD. For small objects, say 0.2–2m, laser-based scanners are a good solution. These scanners provide registered depth and colored texture maps, usually in either raster or cylindrical coordinates [Cyberware Laboratory Inc, 1990; Rioux and Bird, 1993]. They have been used extensively in the entertainment industry for special effects and computer animation. Unfortunately, laser-based scanners are fairly expensive, limited in resolution (typically 512 × 256 pixels), and most importantly, limited in range (e.g., they cannot be used to scan in a building.)

The image-based ranging techniques which we develop in this paper have the potential to overcome these limitations. Imagine walking through an environment such as a building interior and filming a video sequence of what you see. By registering and compositing the images in the

[1] The term *telepresence* is also often used, especially for applications such as tele-medicine where two-way interactivity (remote operation) is important [Earnshaw *et al.*, 1993].

video together into large mosaics of the scene, image-based ranging can achieve an essentially unlimited resolution.[2] Since the images can be acquired using any optical technology (from microscopy, through hand-held videocams, to satellite photography), the range or scale of the scene being reconstructed is not an issue. Finally, as desktop video becomes ubiquitous in our computing environments—initially for videoconferencing, and later as an advanced user interface tool [Gold, 1993; Rehg and Kanade, 1994; Blake and Isard, 1994]—image-based scene and model acquisition will become accessible to all users.

In this paper, we develop novel techniques for extracting large 2-D textures and 3-D models from image sequences based on image registration and compositing techniques and also present some potential applications. After a review of related work (Section 2) and of the basic image formation equations (Section 3), we present our technique for registering pieces of a flat (planar) scene, which is the simplest interesting image mosaicing problem (Section 4). We then show how the same technique can be used to mosaic panoramic scenes obtained by rotating the camera around its center of projection (Section 5). Section 6 discusses how to recover depth in scenes. Section 7 discusses the most general and difficult problem, that of building full 3-D models from multiple images. Section 8 presents some novel applications of the tele-reality technology developed in this paper. Finally, we close with a comparison of our work with previous approaches, and a discussion of the significance of our results.

2 Related work

While 3-D models acquired with laser-based scanners are now commonly used in computer animations and for special effects, 3-D models derived directly from still or video images are still a rarity. One example of physically-based models derived from images are the dancing vegetables in the "Cooking with Kurt" video [Terzopoulos and Witkin, 1988]. Another, more recent example, is the 3-D model of a building extracted from a video sequences in [Azarbayejani *et al.*, 1993], which was then composited with a 3-D teapot. Somewhat related are graphics techniques which extract camera position information from images, such as [Gleicher and Witkin, 1992].

[2] The traditional use of the term *image compositing* [Porter and Duff, 1984; Foley *et al.*, 1990] is for blending images which are already registered. We use the term *image mosaicing* in this paper for our work to avoid confusion with this previous work, although compositing (as in composite sketches) also seems appropriate.

The extraction of geometric information from multiple images has long been one of the central problems in computer vision [Ballard and Brown, 1982; Horn, 1986] and photogrammetry [Moffitt and Mikhail, 1980]. However, many of the techniques used are based on feature extraction, produce sparse descriptions of shape, and often require manual assistance. Certain techniques, such as stereo, produce depth or elevation maps which are inadequate to model true 3-D objects. In computer vision, the recovered geometry is normally used either for object recognition or for grasping and navigation (robotics). Relatively little attention has been paid to building 3-D models registered with intensity (texture maps), which are necessary for computer graphics and virtual reality applications (but see [Mann, 1993] for recent work which addresses some of the same problems as this paper.)

In computer graphics, compositing multiple image streams together to create larger format (*Omnimax*) images is discussed in [Greene and Heckbert, 1986]. However, in this application, the relative position of the cameras was known in advance. The registration techniques developed in this paper are related to *image warping* [Wolberg, 1990] since once the images are registered, they can be warped into a common reference frame before being composited.[3] While most current techniques require the manual specification of feature correspondences, some recent techniques [Beymer *et al.*, 1993] as well as the techniques developed in this paper can be used to automate this process. Combinations of local image warping and compositing are now commonly used for special effects under the general rubric of *morphing* [Beier and Neely, 1992]. Recently, morphing based on z-buffer depths and camera motion has been applied to view interpolation as a quick alternative to full 3-D rendering [Chen and Williams, 1993]. The techniques we develop in Section 6 can be used to compute the depth maps necessary for this approach directly from real-world image sequences.

3 Basic imaging equations

Many of the ideas in this paper have their simplest expression using projective geometry [Semple and Kneebone, 1952]. However, rather than relying on these results, we will use more standard methods and notations from computer graphics [Foley *et al.*, 1990], and prove whatever simple

[3] One can view the image registration task as an *inverse warping* problem, since we are given two images and asked to recover the unknown warping function.

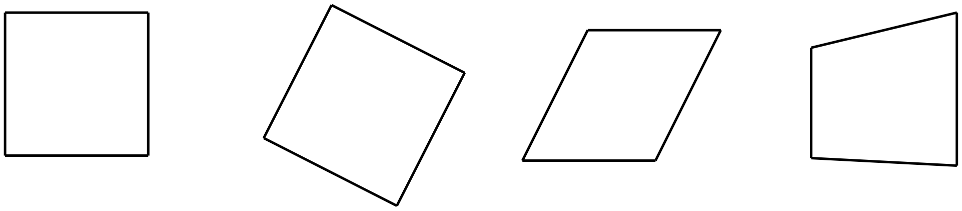


Figure 1: Rigid, affine, and projective transformations

results we require in Appendix A. Throughout, we will use *homogeneous coordinates* to represent points, i.e., we denote 2-D points in the image plane as (x, y, w), with $(x/w, y/w)$ being the corresponding *Cartesian coordinates* [Foley *et al.*, 1990]. Similarly, 3-D points with homogeneous coordinates (x, y, z, w) have Cartesian coordinates $(x/w, y/w, z/w)$.

Using homogeneous coordinates, we can describe the class of 2-D planar transformations using matrix multiplication

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} \quad \text{or} \quad \mathbf{u}' = \mathbf{M}_{\text{2D}}\mathbf{u}. \tag{1}$$

The simplest transformations in this general class are pure translations, followed by translations and rotations (rigid transforms), plus scaling (similarity transforms), affine transforms, and full projective transforms. Figure 1 shows a square and possible rigid, affine, and projective transformations. Rigid and affine transformations have the following forms for the $\mathbf{M}_{\text{2D}}$ matrix,

$$\mathbf{M}_{\text{rigid-2D}} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{M}_{\text{affine-2D}} = \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ 0 & 0 & 1 \end{bmatrix},$$

with 3 and 6 degrees of freedom, respectively, while projective transformations have a general $\mathbf{M}_{\text{2D}}$ matrix with 8 degrees of freedom.[4]

In 3-D, we have the same hierarchy of transformations, with rigid (6 degrees of freedom), similarity (7 dof), affine (12 dof), and full projective (15 dof) transforms. The $\mathbf{M}_{\text{3D}}$ matrices in this

[4] Two $\mathbf{M}_{\text{2D}}$ matrices are equivalent if they are scalar multiples of each other. We remove this redundancy by setting $m_{22} = 1$.

case are 4 × 4. Of particular interest are the rigid (Euclidean) transformation,

$$\mathbf{E} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{2}$$

where $\mathbf{R}$ is a 3×3 orthonormal rotation matrix and $\mathbf{t}$ is a 3-D translation vector, and the 3×4 viewing matrix

$$\mathbf{V} = \begin{bmatrix} \hat{\mathbf{V}} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix}, \tag{3}$$

which projects 3-D points through the origin onto a 2-D projection plane a distance f along the z axis [Foley *et al.*, 1990]. The 3×3 $\hat{\mathbf{V}}$ matrix can be a general matrix in the case where the internal camera parameters are unknown. The last column of $\mathbf{V}$ is always zero for central projection.

The combined equations projecting a 3-D world coordinate $\mathbf{p} = (x, y, z, w)$ onto a 2-D screen location $\mathbf{u} = (x', y', w')$ can thus be written as

$$\mathbf{u} = \mathbf{VEp} = \mathbf{M}_{\text{cam}}\mathbf{p}, \tag{4}$$

where $\mathbf{M}_{\text{cam}}$ is a 3×4 *camera matrix.* This equation is valid even if the camera calibration parameters and/or the camera orientation are unknown.

4 Planar image mosaicing

The simplest possible set of images to mosaic are pieces of a planar scene such as a document, whiteboard, or flat desktop. Imagine that we have a camera fixed directly over our desk. As we slide a document under the camera, different portions of the document are visible. Any two such pieces are related to each other by a translation and a rotation (2-D rigid transformation).

Now imagine that we are scanning a whiteboard with a hand-held video camera. The class of transformations relating two pieces of the board is more general in this case. It is easy to see that this class is the family of 2-D projective transformations (just imagine how a square or grid in one image can appear in another.) These transformations can be computed without any knowledge of the internal camera calibration parameters (such as focal length or optical center) or of the relative camera motion between frames. The fact that 2-D projective transformations capture all such possible mappings is a basic result of projective geometry (see Appendix A.)

Given this knowledge, how do we go about computing the transformations relating the various scene pieces so that we can paste them together? A variety of techniques are possible, some more automated than others. For example, we could manually identify four or more corresponding points between the two views, which is enough information to solve for the eight unknowns in the 2-D projective transformation. Or we could iteratively adjust the relative positions of input images using either a blink comparator or transparency [Carlbom *et al.*, 1991]. These kinds of manual approaches are too tedious to be useful for large tele-reality applications.

4.1 Local image registration

The approach we use in this paper is to directly minimize the discrepancy in intensities between pairs of images after applying the transformation we are recovering. This has the advantage of not requiring any easily identifiable feature points, and of being statistically optimal once we are in the vicinity of the true solution [Szeliski and Coughlan, 1994]. More concretely, suppose we describe our 2-D transformations as

$$x_i' = \frac{m_0 x_i + m_1 y_i + m_2}{m_6 x_i + m_7 y_i + \ 1}, \ y_i' = \frac{m_3 x_i + m_4 y_i + m_5}{m_6 x_i + m_7 y_i + \ 1}. \tag{5}$$

Our technique minimizes the sum of the squared intensity errors

$$E = \sum_i [I'(x_i', y_i') - I(x_i, y_i)]^2 = \sum_i e_i^2 \tag{6}$$

over all corresponding pairs of pixels i which are inside both images $I(x, y)$ and $I'(x', y')$ (pixels which are mapped outside image boundaries do not contribute.) Once we have found the best transformation $\mathbf{M}_{2\mathrm{D}}$, we can *warp* image I' into the reference frame of I using $\mathbf{M}_{2\mathrm{D}}^{-1}$ [Wolberg, 1990] and then blend the two images together. To reduce visible artifacts, we weight images being blended together more heavily towards the center, using a bilinear weighting function.

To perform the minimization, we use the Levenberg-Marquardt iterative non-linear minimization algorithm [Press *et al.*, 1992]. This algorithm requires the computation of the partial derivatives of e_i with respect to the unknown motion parameters $\{m_0 \ldots m_7\}$. These are straightforward to compute, i.e.,

$$\frac{\partial e_i}{\partial m_0} = \frac{x_i}{D_i}\frac{\partial I'}{\partial x'}, \quad \ldots \quad, \frac{\partial e_i}{\partial m_7} = -\frac{y_i}{D_i}\left(x_i'\frac{\partial I'}{\partial x'} + y_i'\frac{\partial I'}{\partial y'}\right), \tag{7}$$

where D_i is the denominator in (5), and $(\partial I'/\partial x', \partial I'/\partial y')$ is the image intensity gradient of I' at (x_i', y_i'). From these partials, the Levenberg-Marquardt algorithm computes an approximate Hessian matrix $\mathbf{A}$ and the weighted gradient vector $\mathbf{b}$ with components

$$a_{kl} = \sum_i \frac{\partial e_i}{\partial m_k} \frac{\partial e_i}{\partial m_l}, \quad b_k = -2 \sum_i e_i \frac{\partial e_i}{\partial m_k}, \tag{8}$$

and then updates the motion parameter estimate $\mathbf{m}$ by an amount $\Delta\mathbf{m} = (\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{b}$, where λ is a time-varying stabilization parameter [Press *et al.*, 1992]. The advantage of using Levenberg-Marquardt over straightforward gradient descent is that it converges in fewer iterations. The complete registration algorithm thus consists of the following steps:

1. For each pixel i at location (x_i, y_i),
   (a) compute its corresponding position in the other image (x_i', y_i') using (5);
   (b) compute the error in intensity between the corresponding pixels $e_i = I'(x_i', y_i') - I(x_i, y_i)$ and the intensity gradient $(\partial I'/\partial x', \partial I'/\partial y')$;
   (c) compute the partial derivative of e_i w.r.t. the m_k using
   $$\frac{\partial e_i}{\partial m_k} = \frac{\partial I'}{\partial x'} \frac{\partial x'}{\partial m_k} + \frac{\partial I'}{\partial y'} \frac{\partial y'}{\partial m_k}$$
   as in (7);
   (d) add the pixel's contribution to $\mathbf{A}$ and $\mathbf{b}$ as in (8).
2. Solve the system of equations $(\mathbf{A} + \lambda\mathbf{I})\Delta\mathbf{m} = \mathbf{b}$ and update the motion estimate $\mathbf{m}^{(t+1)} = \mathbf{m}^{(t)} + \Delta\mathbf{m}$.
3. Check that the error in (6) has decreased; if not, increment λ (as described in [Press *et al.*, 1992]) and compute a new $\Delta\mathbf{m}$,
4. Continue iterating until the error is below a threshold or a fixed number of steps has been completed.

The steps in this algorithm are similar to the operations performed when warping images [Wolberg, 1990; Beier and Neely, 1992], with additional operations for correcting the current warping parameters based on the local intensity error and its gradients (similar to the process in [Gleicher and Witkin, 1992]). For more details on the exact implementation, see [Szeliski and Coughlan, 1994].

4.2 Global image registration

Unfortunately, both gradient descent and Levenberg-Marquardt only find locally optimal solutions. If the motion between successive frames is large, we must use a different strategy to find the best registration. We have implemented two different techniques for dealing with this problem. The first technique, which is commonly used in computer vision, is *hierarchical matching*, which first registers smaller, subsampled versions of the images where the apparent motion is smaller [Quam, 1984; Witkin *et al.*, 1987; Bergen *et al.*, 1992]. Motion estimates from these smaller coarser levels are then used to initialize motion estimates at finer levels, thereby avoiding the local minimum problem (see [Szeliski and Coughlan, 1994] for details.) While this technique is not guaranteed to find the correct registration, we have observed empirically that it works well when the initial misregistration is only a few pixels (the exact domain of convergence depends on the intensity pattern in the image.)

For larger displacements, we use *phase correlation* [Kuglin and Hines, 1975; Brown, 1992]. This technique estimates the 2-D translation between a pair of images by taking 2-D Fourier Transforms of each image, computing the phase difference at each frequency, performing an inverse Fourier Transform, and searching for a peak in the magnitude image. In our experiments, this technique has proven to work remarkably well, providing good initial guesses for image pairs which overlap by as little as 50%. The technique will not work, however, if the inter-frame motion is not mostly translational (large rotations or zooms), but this does not occur in practice with our current image sequences.

4.3 Results

To demonstrate the performance of our algorithm, we digitized an image sequence with a camera panning over a whiteboard. Figure 2a shows the final mosaic of the whiteboard, with the locations of the constituent images in the mosaic shown as black outlines. Figure 2b is a portion of the final mosaic without these outlines. This mosaic is 1300×2046 pixels, based on compositing 39 NTSC (640×480) resolution images.

To compute this mosaic, we developed an interactive image mosaicing tool which lets the user coarsely position successive frames relative to each other. The tool also has an *automatic mosaicing* option which computes the initial rough placement of each image with respect to the previous one using phase correlation. Our algorithm then refines the location of each image by minimizing (6) using the current mosaic thus far as $I(x, y)$ and the input frame being adjusted as $I'(x', y')$. The images in Figure 2 were automatically composited without user intervention using the middle frame (center of the image) as the *anchor* image (no deformations.) As we can see, our technique works well on this example. Section 9 discusses some imaging artifacts which can cause difficulties.

5 Panoramic image mosaicing

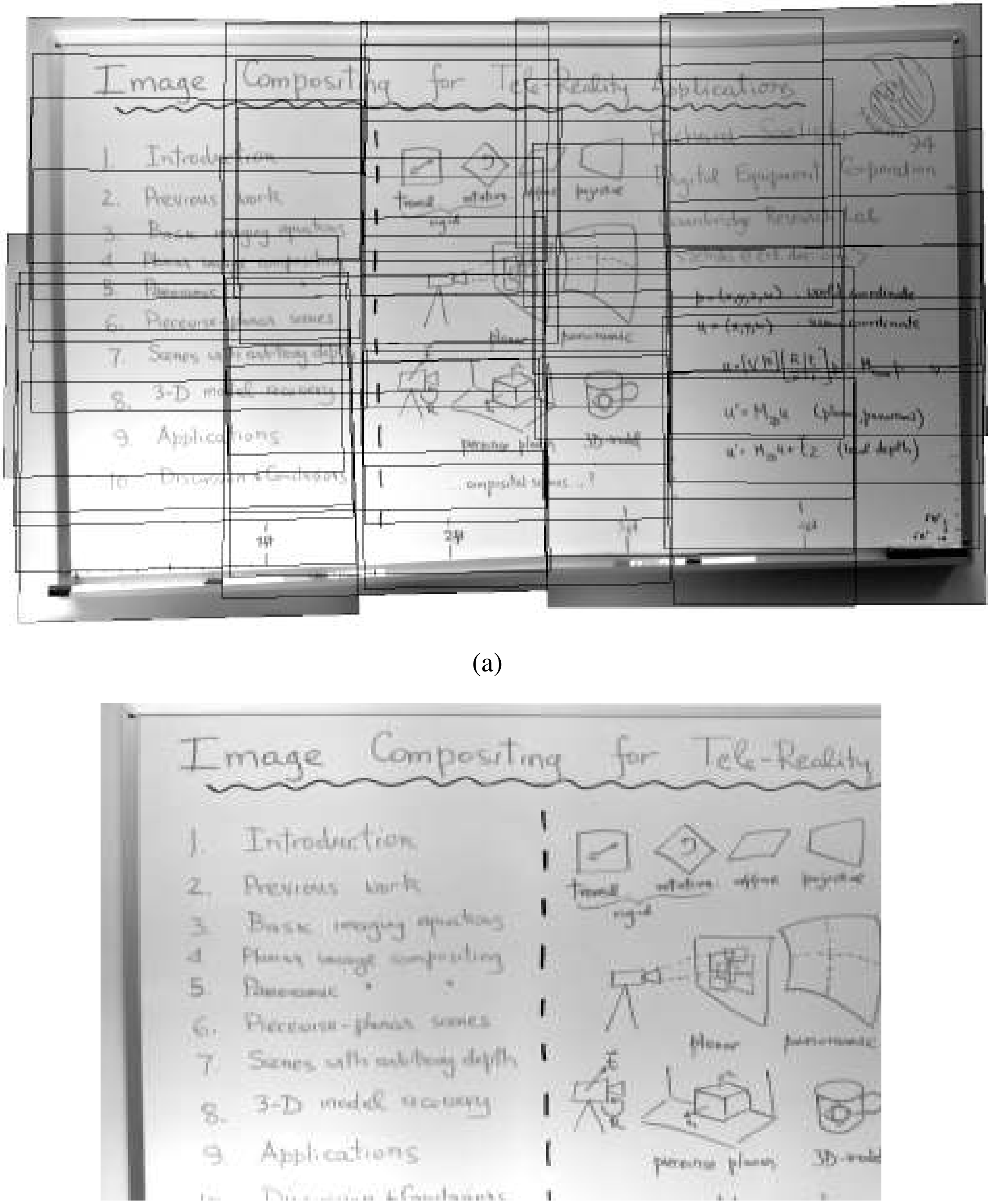
Another image mosaicing problem which turns out to be equally simple to solve is panoramic image mosaicing. In this scenario, we rotate a camera around its optical center in order to create a full "viewing sphere" of images (in computer graphics, this representation is sometimes called an *environment map* [Greene, 1986].) This is similar to the action of panoramic still photographic cameras where the rotation of a camera on top of a tripod is mechanically coordinated with the film transport, the anamorphic lens used in [Lippman, 1980], and to "cinema in the round" movie theaters [Greene and Heckbert, 1986]. In our case, however, we can mosaic multiple 2-D images of arbitrary detail and resolution, and we need not know the camera motion. Examples of applications include constructing true scenic panoramas (say of the view at the rim of the Grand Canyon), or limited virtual environments (recreating a meeting room or office as seen from one location.)

It turns out that images taken from the same viewpoint (stationary optical center) are related by 2-D projective transformations, just as in the planar scene case (see Appendix A for a quick proof.) Intuitively, we cannot tell the relative depth of points in the scene as we rotate (there is no *motion parallax*), so they may as well be located on any plane (in projective geometry, we could say they lie on the *plane at infinity*, $w = 0$.) For some applications, this property is usually viewed as a deficiency, i.e., we cannot recover scene depth from rotational camera motion. For image mosaicing, this is actually an advantage since we just want to composite a large scene and be able to "look around" from a static viewpoint.

More formally, the 2-D transformation denoted by $\mathbf{M}_{\mathrm{2D}}$ is related to the viewing matrices $\hat{\mathbf{V}}$ and $\hat{\mathbf{V}}'$ and the inter-view rotation $\mathbf{R}$ by

$$\mathbf{M}_{\mathrm{2D}} = \hat{\mathbf{V}}'\mathbf{R}\hat{\mathbf{V}}^{-1} \tag{9}$$

(see Appendix A.) In the case of a calibrated camera (known center of projection), this has a



(a)

(b)

Figure 2: Whiteboard image mosaic example

(a) mosaic with component locations shown as black outlines, (b) portion of mosaic in greater detail.

particularly simple form,

$$\mathbf{M}_{2D} = \begin{bmatrix} r_{00} & r_{01} & fr_{02} \\ r_{10} & r_{11} & fr_{12} \\ r_{20}/f' & r_{21}/f' & fr_{22}/f' \end{bmatrix}, \tag{10}$$

where f and f' are the scaled focal lengths in the two views, and r_{kl} are the entries in the rotation matrix $\mathbf{R}$.[5] In this case, we only have to recover five independent parameters (or three if the f values are known) instead of the usual eight.

How do we represent a panoramic scene composited using our techniques? One approach is to divide the viewing sphere into several large, potentially overlapping regions, and to represent each region with a plane onto which we paste the images [Greene, 1986]. Another approach is to compute the relative position of each frame relative to some base frame, and to then re-compute an arbitrary view on the fly from all visible pieces, given a particular view direction $\mathbf{R}$ and zoom factor f. In our current implementation, we adopt the former strategy since it is a simple extension of the planar case.

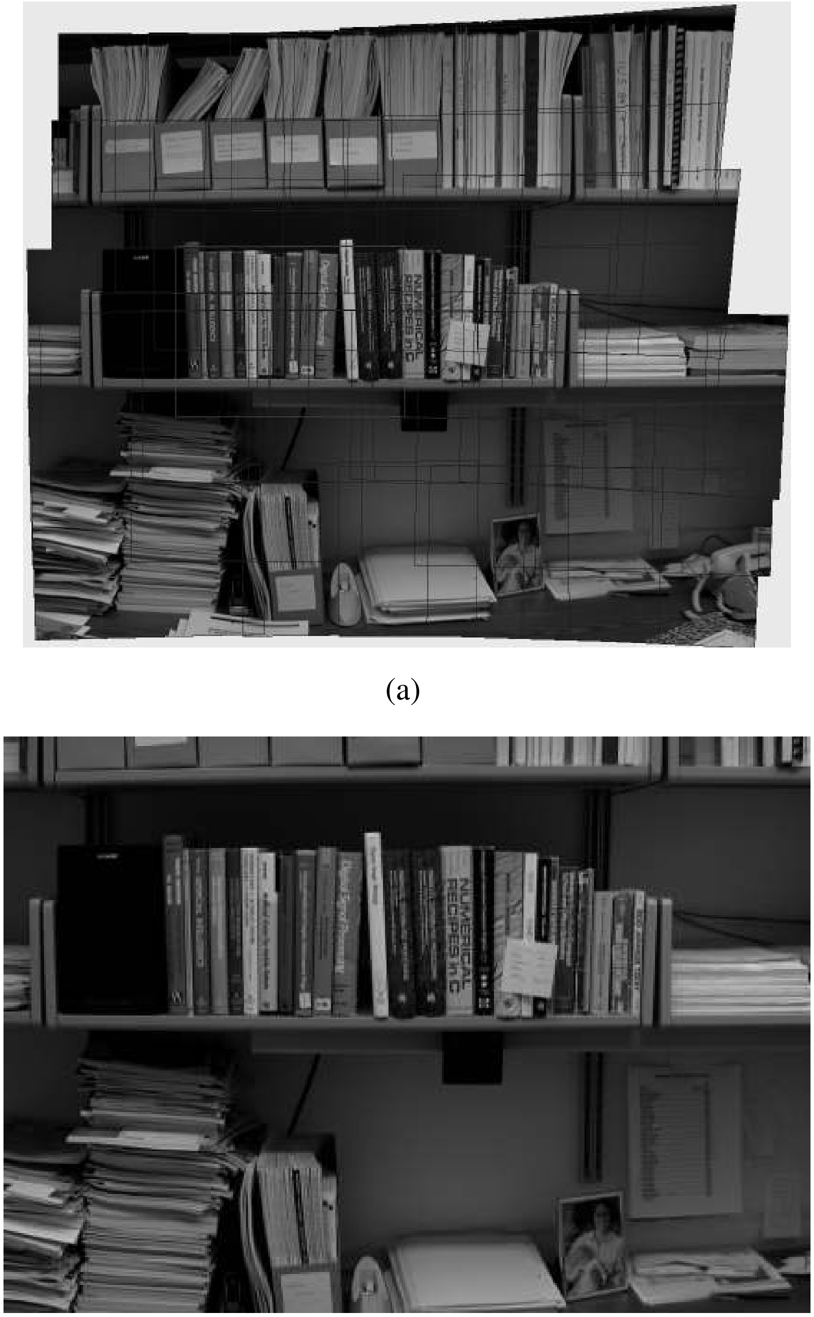
5.1 Results

Figure 3 shows a mosaic of a bookshelf and cluttered desk, which is obviously a non-planar scene. The images were obtained by tilting and panning a video camera mounted on a tripod, without any special steps taken to ensure that the rotation was around the true center of projection. The complete scene is registered quite well.

6 Scenes with arbitrary depth

Mosaicing flat scenes or panoramic scenes may be interesting for certain tele-reality applications, e.g., transmitting whiteboard or document contents, or viewing an outdoor panorama. However, for most applications, we must recover the depth associated with the scene to give the illusion of a 3-D environment, e.g., through nearby view synthesis [Chen and Williams, 1993]. Two possible approaches are to model the scene as piecewise-planar or to recover dense 3-D depth maps.

[5] At large focal lengths and small rotations, $\mathbf{M}_{2D}$ resembles a pure translation matrix.



(a)

(c)

Figure 3: Panoramic image mosaic example (bookshelf and cluttered desk)

(a) mosaic with component locations shown as black outlines, (b) portion of mosaic in greater detail.

The first approach is to assume that the scene is piecewise-planar, as is the case with many man-made environments such as building exteriors and office interiors. The image mosaicing technique developed in Section 4 can then be applied to each of the planar regions in the image. The segmentation of each image into its planar components can either be done interactively (e.g., by drawing the polygonal outline of each region to be registered), or automatically by associating each pixel with one of several global motion hypotheses [Jepson and Black, 1993; Wang and Adelson, 1993]. Once the independent planar pieces have been composited, we could, in principle, recover the relative geometry of the various planes and the camera motion (see Appendix A.) However, rather than pursuing this approach in this paper, we will purse the second, more general solution which is to recover a full depth map, i.e., to infer the missing z component associated with each pixel in a given image sequence.

When the camera motion is known, the problem of depth map recovery is called *stereo reconstruction* (or multi-frame stereo if more than two views are used [Matthies *et al.*, 1989; Okutomi and Kanade, 1993].) This problem has been extensively studied in photogrammetry and computer vision [Moffitt and Mikhail, 1980; Barnard and Fischler, 1982; Horn, 1986; Dhond and Aggarwal, 1989]. When the camera motion is unknown, we have the more difficult *structure from motion* problem [Horn, 1986; Weng *et al.*, 1993; Azarbayejani *et al.*, 1993; Szeliski and Kang, 1994]. In this section, we present our solution to this latter problem based on recovering *projective depth*, which is particularly simple and robust and fits in well with the material already developed in this paper. Readers interested in alternative techniques should consult the above references.

6.1 Formulation

To formulate the projective structure from motion recovery problem, we note that the coordinates corresponding to a pixel $\mathbf{u}$ with projective depth w in some other frame can be written as

$$\mathbf{u}' = \mathbf{V}'\mathbf{E}\mathbf{p} = \hat{\mathbf{V}}'\mathbf{R}\hat{\mathbf{V}}^{-1}\mathbf{u} + w\hat{\mathbf{V}}'\mathbf{t} = \mathbf{M}_{\mathrm{2D}}\mathbf{u} + w\hat{\mathbf{t}}, \tag{11}$$

where $\mathbf{V}$, $\mathbf{E}$, $\mathbf{R}$, and $\mathbf{t}$ are defined in (2–3), and $\mathbf{M}_{\mathrm{2D}}$ and $\hat{\mathbf{t}}$ are the computed planar projection matrix and epipole direction (see (25) in Appendix A.) To recover the parameters in $\mathbf{M}_{\mathrm{2D}}$ and $\hat{\mathbf{t}}$ for each frame along with the depth values w (which are the same for all frames), we use the same Levenberg-Marquardt algorithm as before.

In more detail, we write the projection equation as

$$x'_i = \frac{m_0 x_i + m_1 y_i + t_0 w_i + m_2}{m_6 x_i + m_7 y_i + t_2 w_i + \ 1}, \ y'_i = \frac{m_3 x_i + m_4 y_i + t_1 w_i + m_5}{m_6 x_i + m_7 y_i + t_2 w_i + \ 1}. \tag{12}$$

We compute the partial derivatives of x'_i and y'_i w.r.t. the m_k and t_k (which we concatenate into the motion vector $\mathbf{m}$) as before in (7.) We similarly compute the partials of x'_i and y'_i with respect to w_i, i.e.,

$$\frac{\partial x'_i}{\partial w_i} = \frac{D_i t_0 - t_2}{D_i^2}, \frac{\partial y'_i}{\partial w_i} = \frac{D_i t_0 - t_2}{D_i^2}, \tag{13}$$

where D_i is the denominator in (12).

To estimate the unknown parameters, we alternate iterations of the Levenberg-Marquardt algorithm over the motion parameters $\{m_0, \ldots, t_2\}$ in $\mathbf{m}$ and the depth parameters $\{w_i\}$, using the partial derivatives defined above to compute the approximate Hessian matrices $\mathbf{A}$ and the weighted error vectors $\mathbf{b}$ as in (8). In our current implementation, in order to reduce the total number of parameters being estimated, we represent the depth map using a tensor-product spline, and only recover the depth estimates at the spline control vertices (the complete depth map is available by interpolation) [Szeliski and Coughlan, 1994].

6.2 Results

Figure 4 shows an example of using our projective depth recovery algorithm. The image sequence was taken by moving the camera up and over the scene of a table with stacks of papers (Figure 4a). The resulting depth-map is shown in Figure 4b as intensity-coded range values. Figures 4c–d show the original intensity image texture mapped onto the surface (an example of view extrapolation), and Figures 4c–d show a set of grid-lines overlayed on the recovered surface. The shape is recovered reasonably well in areas where there is sufficient texture (uniform intensity areas give no depth cues), and the extrapolated view looks reasonable. Figure 5a shows results from another sequence, this time from an outdoor scene of some trees. Again, the geometry of the scene is recovered reasonably well, as indicated in the depth map in Figure 5b.

7 Full 3-D model recovery

Recovering depth maps may be adequate for many tele-reality applications, e.g., scanning library bookshelves or supermarket aisles, but for other applications, e.g., when we need to see the back

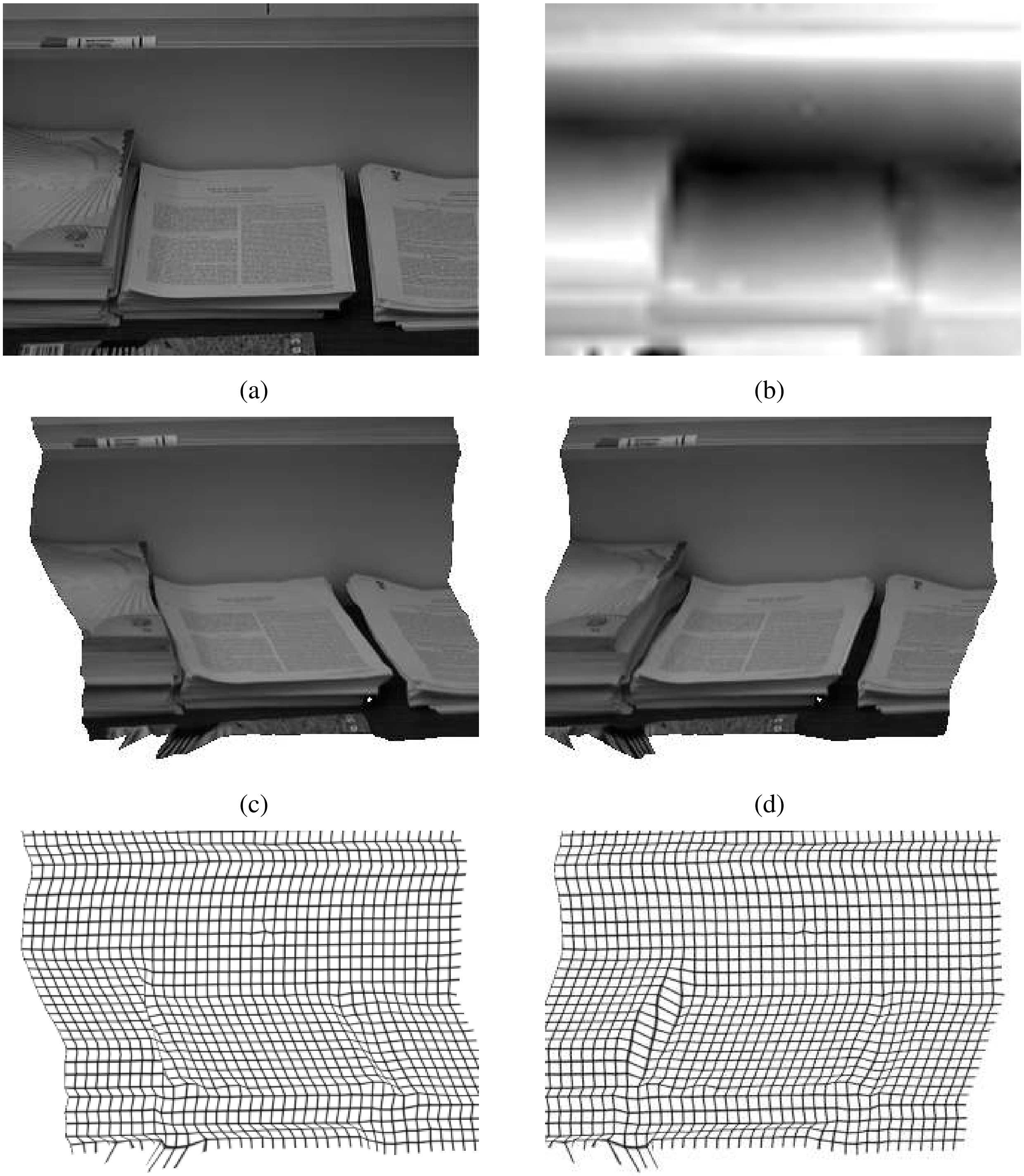
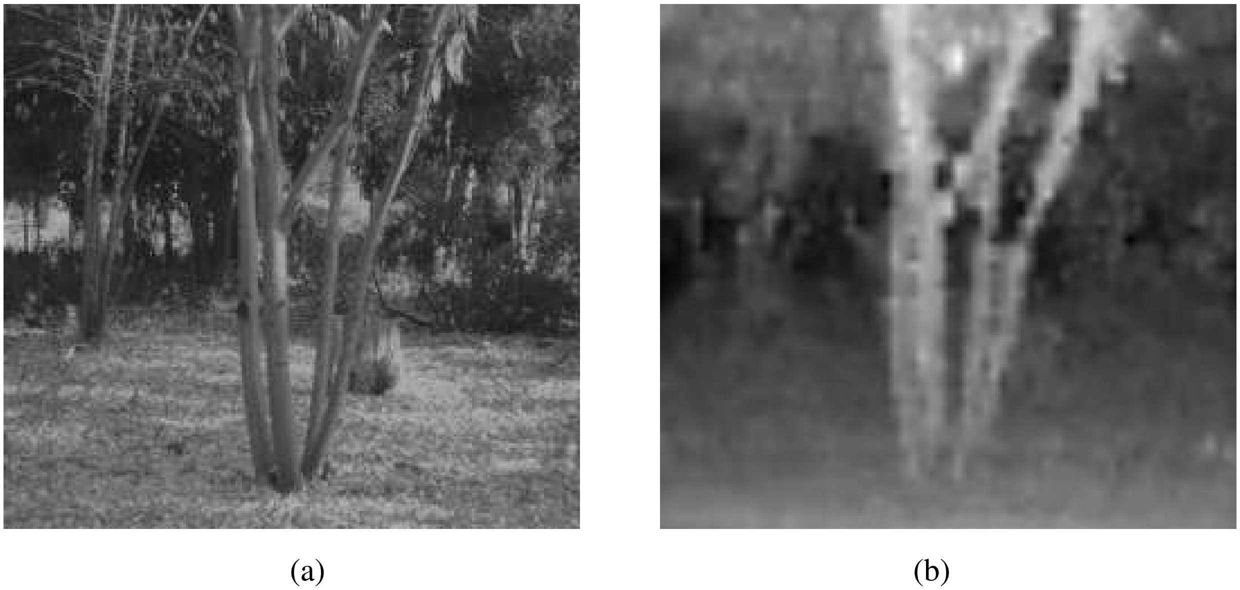


Figure 4: Depth recovery example: table with stacks of papers

(a) input image, (b) intensity-coded depth map (dark is farther back), (c–d) texture-mapped surface seen from novel viewpoints, (e–f) gridded surface.



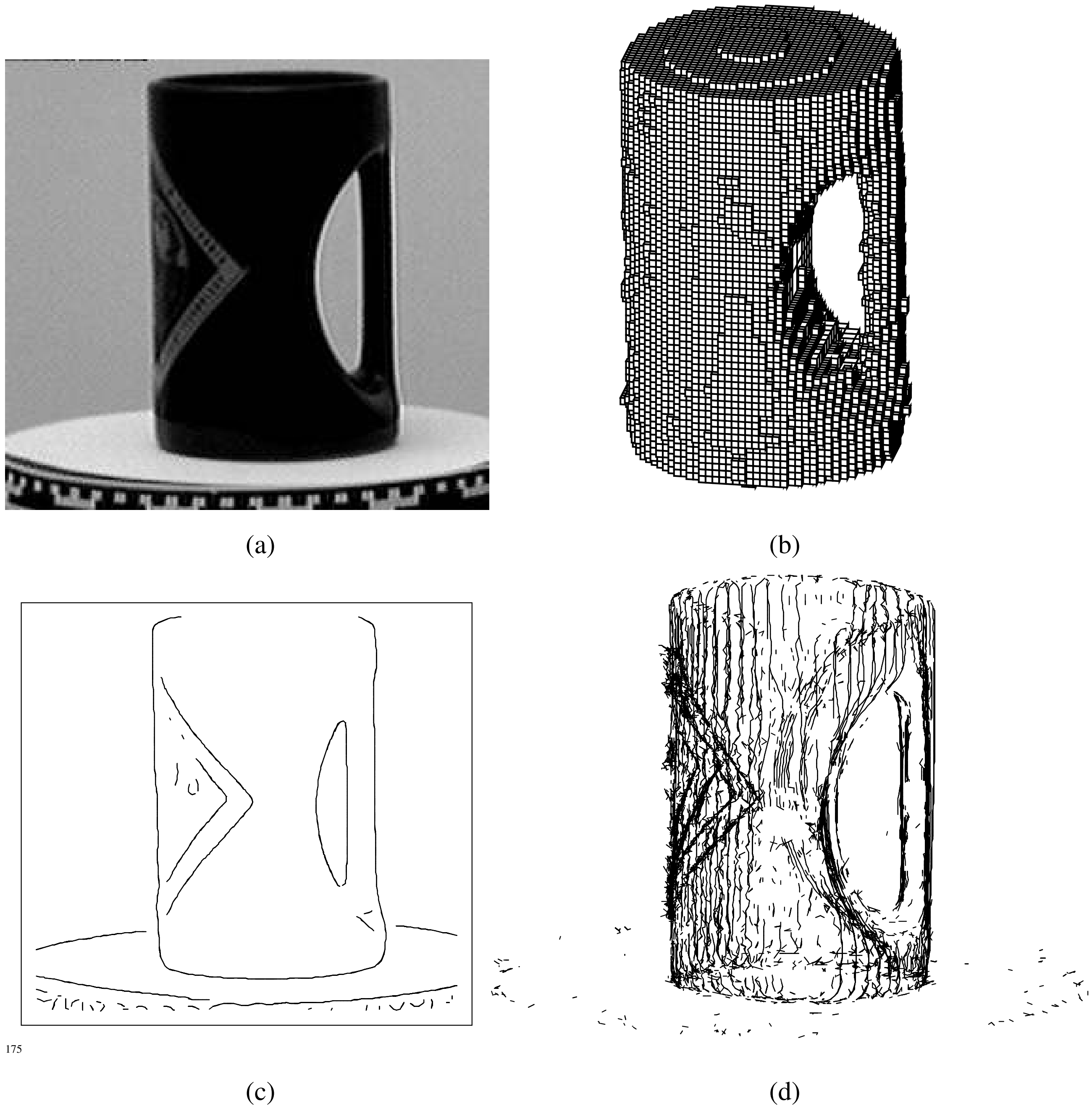
(a) (b)

Figure 5: Depth recovery example: outdoor scene with trees
(a) input image, (b) intensity-coded depth map (dark is farther back).

side of an object, full 3-D models may be required. This is the most difficult image mosaicing problem, since not only do we have to recover depth, but we also have to merge (register and composite) multiple depth maps, and represent objects given no *a priori* knowledge about their rough shape or topology.

For simple topologies and shapes, deformable physically-based models can do a good job of recovering the unknown geometry [Terzopoulos *et al.*, 1987; Pentland and Sclaroff, 1991]. For general topologies, the problem is more difficult. Since many current shape recovery techniques produce incomplete or sparse geometric descriptions, a general 3-D surface interpolation technique may be required to generate a smooth continuous surface [Hoppe *et al.*, 1992; Szeliski and Tonnesen, 1992].

There exist many techniques for extracting 3-D shape from multiple views. For example, we can recover a volumetric description from the binary *silhouettes* of an object against its background [Szeliski, 1993], compute local optic flow (pixel motion) estimates and convert these into sparse 3-D point estimates [Szeliski, 1991], or track the occluding contours of an object to generate 3-D space curves [Szeliski and Weiss, 1993]. A complete survey of 3-D shape extraction techniques is beyond the scope of this paper. Instead, we present the results of two of our previously developed algorithms applied to an image sequence of a cup rotating on a calibrated turntable (Figure 6a).



(a)

(b)

(c)

(d)

Figure 6: 3-D model recovery example

(a) input image, (b) octree recovered from silhouettes (c) 2-D edges, (d) 3-D extremal contours

Our first technique converts each image into a binary silhouette by differencing the image with an empty background image. Each cube in the octree volumetric 3-D model is then projected (using the known camera position) into the silhouette, and cubes that fall outside the silhouette are culled. Cubes which fall partially into the silhouette are marked for later subdivision, and the process is repeated after each complete revolution at successively finer resolutions [Szeliski, 1993]. The resulting octree is shown in Figure 6b.

Our second technique extracts silhouette (extremal) edges from the image sequence, and uses a multi-frame stereo algorithm to reconstruct their 3-D position (internal albedo edges are also extracted and reconstructed) [Szeliski and Weiss, 1993]. Figure 6c shows one of the edge images used by the algorithm, and Figure 6d shown the collection of 3-D curves estimated by the algorithm.

These examples demonstrate some of our algorithms for reconstructing an isolated object undergoing known motion. Similar techniques can be used to solve the more general 3-D scene recovery problem where the camera motion is unknown. Methods for determining the motion include the projective motion algorithm presented in the previous section, as well as techniques described in [Horn, 1986; Weng *et al.*, 1993; Azarbayejani *et al.*, 1993]. Compared to the 2-D and depth-map models developed in Sections 4–6, the recovery of 3-D models is more difficult and less robust, but holds the promise of more realistic and more general object and scene models.

8 Applications

Given automated techniques for building 2-D and 3-D scenes from video sequences, what can we do with these models? In this section, we describe a number of potential applications, including whiteboard and document scanning, 3-D model acquisition for inverse CAD, model acquisition for computer animation and special effects, supermarket shopping at home, interactive walkthroughs of historical buildings, and live tele-reality (telepresence) applications.

8.1 Planar mosaicing applications

The most straightforward application is scanning whiteboards or blackboards as an aid to videoconferencing or as an easy way to capture ideas. Scanning can produce images of much greater resolution than single wide-angle lens shots. This facility could be combined with a video projection system to create a *virtual whiteboard* similar to the *DigitalDesk* proposed by Wellner [Wellner, 1993] (see [Gold, 1993] for other examples of ubiquitous computing and augmented reality.) The techniques developed in this paper enable any video camera attached to a computer to be used. In specialized situations, e.g., in classrooms, a computer-controlled camera could be used to perform the scanning, removing the need for automatic registration of the images.

Another obvious application of this technology is for document scanning. Hand-held scanners currently perform this function quite well. However, since they are based on linear CCDs, they are subject to “skewing” problems in each strip of the scanned image which must be corrected manually. Using 2-D video images as input removes some of these problems. A final global skew may still be needed to make the document square, but this can be performed automatically by detecting document edges or internal horizontal and vertical edges (e.g., column borders).

8.2 3-D model building applications

The 3-D model building technology, while more difficult to automate reliably, has even greater potential. For example, we could construct a *3-D fax* which would scan 3-D objects at one end using a video camera and either mechanical or user-controlled motion, and a 3-D graphics display at the other end [Carlbom *et al.*, 1992]. This fax could be used to transmit 3-D models to a remote site for viewing and design revision, or to input rough CAD models for reverse engineering or clay mock-up applications. Of course, this technology could also be combined with stereolithography or numerically controlled milling to make this a true solid 3-D fax [Bresenham, 1993].

Rapid 3-D model building is also critical in the computer animation and special effects industries. Currently, it can take days or weeks to build by hand each computer model used in a feature-length film. The ability to build such models rapidly from real objects or physical mock-ups would be of great utility, especially when freed from the range and resolution limitations of active rangefinding systems. The techniques we use for computing 3-D geometry could also be modified to automatically track the motions of objects or actors in a scene (this is called *motion capture*, and is currently done using specially placed markers.)

8.3 End-user applications

A more mass-market application is home shopping applied not to single objects (either demonstrated by a model or available in a catalog), but to complete stores, such as your local supermarket. This has the advantage of having a familiar look and organization, and enables the pre-planning of your next shopping trip. The images of the aisles (with their current contents) can be digitized by rolling a video camera through the store. More detailed geometric models of individual items can be acquired either by a 3-D model building process, or, in the future, directly from the manufacturer. The shopper can then stroll down the aisles, pick out individual items, and look at their ingredients and prices.

A similar scenario is possible with other buildings. Architectural walkthroughs have long been a part of computer graphics [Greenberg, 1974], but only for buildings currently under design. Walkthrough of existing building can have a number of applications. For example, interactive 3-D walkthroughs of your home, built by walking a video camera through the rooms and mosaicing the image sequences, could be used for selling your house (an extension of existing still-image based systems), or for re-modeling or renovating its design. Walkthroughs of historic building (e.g., palaces or museums) can be used for educational and entertainment purposes. A museum scenario might include the ability to look at individual 3-D objects such as sculptures, and to bring up related information in a hypertext system [Miller *et al.*, 1991].

Walkthroughs (or fly-throughs) can also be done in general (e.g., outdoor) 3-D scenes. An early example of this was the Movie-Maps project [Lippman, 1980] which was based on videodisc technology. Since this system was based on choosing from a number of pre-stored motion sequences, the range of possible viewpoints (and detail) was limited. The basic Movie-Maps idea can be extended by building true 3-D scenes from the input motion sequences. Such 3-D tele-reality models have the potential of being of extremely high complexity and will require solving problems in representation, partial 3-D models [Chen and Williams, 1993], and switching between different levels of resolution [Funkhouser and Séquin, 1993].

The ultimate in tele-reality systems is dynamic tele-reality (sometimes called *telepresence*), which composites video from multiple source in real-time to create the illusion of being in a dynamic (and perhaps reactive) 3-D environment. An example of such an application might be to view a 3-D version of a concert with control over the camera shots, even being able to see the concert from the musicians' point of view. Other examples might be to participate or consult in a surgery from a remote location (*tele-medicine*), or to remotely participate in a *virtual classroom*. Building such dynamic 3-D models at frame rates is beyond the processing power of today's high-performance superscalar workstations, but it could be achieved using a collection of such machines (or special-purpose stereo hardware). In such applications, we need to use multiple cameras, since in moving scenes we cannot rely on the motion parallax from a single camera to give us reliable shape information.

9 Discussion

Image mosaicing provides a powerful new way of creating the detailed 3-D models and scenes needed for tele-reality applications. By registering multiple images together, we can create scenes of extremely high resolution and at the same time recover partial or full 3-D geometric information. While we use techniques from computer vision to perform the registration, our focus is different: traditional vision techniques are designed for inspection, recognition, and robot control, while our techniques are designed to produce realistic 3-D models for computer graphics and virtual reality applications.

The approach we use, namely direct minimization of intensity differences between warped images, has a number of advantages over more traditional vision techniques, which are based on tracking features from frame to frame [Tomasi and Kanade, 1992; Azarbayejani *et al.*, 1993; Szeliski and Kang, 1994]. Our techniques produce dense estimates of shape, work in highly textured areas where features may not be reliably observed, and make statistically optimal use of all the information [Szeliski and Coughlan, 1994]. Our approach is similar to [Bergen *et al.*, 1992], who also use intensity differences. However, we use full 2-D projective models of motion instead of instantaneous quadratic flow fields, and we also use a projective formulation of structure from motion, which eliminates the need for calibrated cameras. It is also very similar to [Mann, 1993], who also use planar projective motion estimates. Furthermore, [Mann and Picard, 1994] have also demonstrated superresolution results, i.e., the ability to obtain higher resolution images by simply jittering the camera rather than panning [Irani and Peleg, 1991].

While our techniques have worked well in the scenes in which we have tried them, we must be cautious about their general applicability. The intensity-based techniques we use are sensitive to image intensity variation, such as those caused by video camera gain control and vignetting (darkening of the corners at wide lens apertures); working with band-pass filtered images can remove most of these problems. All vision techniques are also sensitive to geometric distortions (deviations from the pinhole model) in the optics, so careful calibration is necessary for optimal accuracy (the results in this paper were obtained with uncalibrated cameras.)

The depth extraction techniques we use rely on the presence of texture in the image. In areas of sufficient texture, it is still possible for the registration/matching algorithm to compute an erroneous depth estimate (such gross errors are less common in active rangefinders.) Where texture is absent, interpolation must be used, and this can lead to erroneous (hallucinated) depth estimates. Other visual cues, such as occluding contours or shading [Horn, 1986], can be used to mitigate these problems in some cases, but a general-purpose reliable vision-based ranging system remains very much an open research problem.

The size and structure of the models produced for tele-reality applications have interesting implications for specialized graphics hardware and software interfaces. Current graphics hardware stores the intensity images (texture maps) separately from the geometry, which is stored as polygons usually spanning many pixels. A more appropriate architecture might be registered multi-resolution color images and depth maps. View interpolation, rather than 3-D rendering, could then be used to synthesize novel 3-D views [Chen and Williams, 1993]. These kinds of architectures and algorithms point towards a tighter coupling between computer graphics and image processing [Pickering, 1993].

10 Conclusions

In this paper, we have presented a hierarchy of scene models, ranging from 2-D planar and panoramic mosaics through full 3-D object models, and developed a collection of associated scene recovery algorithms. The creation of realistic high-resolution 3-D scenes from video imagery opens up many new applications for tele-reality technology. These include office applications such as whiteboard, document, and bookshelf scanning, simulated meeting spaces, engineering applications such as reverse engineering and collaborative design, and computer graphics applications such as 3-D model building. More importantly, this technology enables mass market applications such as home shopping, education (the virtual classroom), and entertainment (virtual travel). Ultimately, as processing speeds and reconstruction algorithms improve further, we will see dynamic real-time 3-D scene and model recovery being used to provide an even more exciting range of telepresence and tele-reality applications.

Acknowledgements

Bob Thomas introduced me to the term *tele-reality*, which he used to describe the live concert telepresence scenario. James Coughlan developed the image registration code. David Tonnesen, Demetri Terzopoulos, Sing Bing Kang, and Richard Weiss were collaborators on some of the 3-D reconstruction algorithms described in this paper. William Hsu helped me produce the results in this paper. Ingrid Carlbom and Gudrun Klinker provided useful comments on the manuscript.

References

[Cyberware Laboratory Inc, 1990] Cyberware Laboratory Inc. *4020/RGB 3D Scanner with color digitizer*. Monterey, 1990.

[Adam, 1993] J. A. Adam. Virtual reality is for real. *Spectrum*, :22–29, October 1993.

[Azarbayejani *et al.*, 1993] A. Azarbayejani, B. Horowitz, and A. Pentland. Recursive estimation of structure and motion using relative orientation constraints. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pages 294–299, New York, New York, June 1993.

[Ballard and Brown, 1982] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[Barnard and Fischler, 1982] S. T. Barnard and M. A. Fischler. Computational stereo. *Computing Surveys*, 14(4):553–572, December 1982.

[Beier and Neely, 1992] T. Beier and S. Neely. Feature-based image metamorphosis. *Computer Graphics (SIGGRAPH'92)*, 26(2):35–42, July 1992.

[Bergen *et al.*, 1992] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Second European Conference on Computer Vision (ECCV'92)*, pages 237–252, Springer-Verlag, Santa Margherita Liguere, Italy, May 1992.

[Beymer *et al.*, 1993] D. Beymer, A. Shashua, and T. Poggio. *Example Based Image Analysis and Synthesis*. A. I. Memo 1431, Massachusetts Institute of Technology, November 1993.

[Blake and Isard, 1994] A. Blake and M. Isard. 3d position, attitude, and shape input using video tracking of hands and lips. *Computer Graphics (SIGGRAPH'94)*, July 1994.

[Bresenham, 1993] J. Bresenham. Real virtuality: Stereolithography – rapid prototyping in 3-D. *Computer Graphics (SIGGRAPH'93)*, :377–378, August 1993.

[Brown, 1992] L. G. Brown. A survey of image registration techniques. *Computing Surveys*, 24(4):325–376, December 1992.

[Carlbom *et al.*, 1992] I. Carlbom and others. Modeling and analysis of empirical data in collaborative environments. *Communications of the ACM*, 35(6):74–84, April 1992.

[Carlbom *et al.*, 1991] I. Carlbom, D. Terzopoulos, and K. M. Harris. Reconstructing and visualizing models of neuronal dendrites. In N. M. Patrikalakis, editor, *Scientific Visualization of Physical Phenomena*, pages 623–638, Springer-Verlag, New York, 1991.

[Chen and Williams, 1993] S. Chen and L. Williams. View interpolation for image synthesis. *Computer Graphics (SIGGRAPH'93)*, :279–288, August 1993.

[Dhond and Aggarwal, 1989] U. R. Dhond and J. K. Aggarwal. Structure from stereo—a review. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1489–1510, November/December 1989.

[Earnshaw *et al.*, 1993] R. A. Earnshaw, M. A. Gigante, and H. Jones, editors. *Virtual Reality Systems*. Academic Press, London, 1993.

[Faugeras, 1992] O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Second European Conference on Computer Vision (ECCV'92)*, pages 563–578, Springer-Verlag, Santa Margherita Liguere, Italy, May 1992.

[Foley *et al.*, 1990] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, Reading, MA, 2 edition, 1990.

[Funkhouser and Séquin, 1993] T. A. Funkhouser and C. H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. *Computer Graphics (SIGGRAPH'93)*, :247–254, August 1993.

[Gleicher and Witkin, 1992] M. Gleicher and A. Witkin. Through-the-lens camera control. *Computer Graphics (SIGGRAPH'92)*, 26(2):331–340, July 1992.

[Gold, 1993] R. Gold. Ubiquitous computing and augmented reality. *Computer Graphics (SIGGRAPH'93)*, :393–394, August 1993.

[Greenberg, 1974] D. P. Greenberg. Computer graphics in architecture. *Scientific American*, :98–106, May 1974.

[Greene, 1986] N. Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications*, 6(11):21–29, November 1986.

[Greene and Heckbert, 1986] N. Greene and P. Heckbert. Creating raster Omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Computer Graphics and Applications*, 6(6):21–27, June 1986.

[Hartley, 1994] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, Washington, June 1994.

[Hoppe *et al.*, 1992] W. Hoppe and others. Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH'92)*, 26(2):71–78, July 1992.

[Horn, 1986] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, Massachusetts, 1986.

[Irani and Peleg, 1991] M. Irani and S. Peleg. Improving resolution by image registration. *Graphical Models and Image Processing*, (3), May 1991.

[Jepson and Black, 1993] A. Jepson and M. J. Black. Mixture models for optical flow computation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pages 760–761, New York, New York, June 1993.

[Kuglin and Hines, 1975] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *IEEE 1975 Conference on Cybernetics and Society*, pages 163–165, New York, September 1975.

[Lippman, 1980] A. Lippman. Movie maps: An application of the optical videodisc to computer graphics. *Computer Graphics (SIGGRAPH'80)*, 14(3):32–43, July 1980.

[Mann, 1993] S. Mann. Compositing multiple pictures of the same scene: Generalized large-displacement 8-parameter motion. In *Proceedings of the 46th Annual IS&T Conference*, The Society of Imaging Science and Technology, Cambridge, Massachusetts, May 1993.

[Mann and Picard, 1994] S. Mann and R. W. Picard. The projective group model for multi-frame resolution enhancement. In *First IEEE International Conference on Image Processing (ICIP'94)*, (submitted) 1994.

[Matthies *et al.*, 1989] L. H. Matthies, R. Szeliski, and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209–236, 1989.

[Miller *et al.*, 1991] G. Miller and others. The virtual museum: Interactive 3D navigation of a multimedia database. *Journal of Visualization and Computer Animation*, 3(3):183–198, 1991.

[Moffitt and Mikhail, 1980] F. H. Moffitt and E. M. Mikhail. *Photogrammetry*. Harper & Row, New York, 3 edition, 1980.

[Okutomi and Kanade, 1993] M. Okutomi and T. Kanade. A multiple baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.

[Pentland and Sclaroff, 1991] A. Pentland and S. Sclaroff. Closed-form solutions for physically-based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, July 1991.

[Pickering, 1993] W. R. Pickering. Merging 3-D graphics and imaging – applications and issues. *Computer Graphics (SIGGRAPH'93)*, :395–396, August 1993.

[Porter and Duff, 1984] T. Porter and T. Duff. Compositing digital images. *Computer Graphics (SIGGRAPH'84)*, 18(3):253–259, July 1984.

[Press *et al.*, 1992] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, second edition, 1992.

[Quam, 1984] L. H. Quam. Hierarchical warp stereo. In *Image Understanding Workshop*, pages 149–155, Science Applications International Corporation, New Orleans, Louisiana, December 1984.

[Rehg and Kanade, 1994] J. Rehg and T. Kanade. Visual tracking of high dof articulated structures: an application to human hand tracking. In *Third European Conference on Computer Vision (ECCV'94)*, pages 35–46, Springer-Verlag, Stockholm, Sweden, May 1994.

[Rioux and Bird, 1993] M. Rioux and T. Bird. White laser, synced scan. *IEEE Computer Graphics and Applications*, 13(3):15–17, May 1993.

[Semple and Kneebone, 1952] J. G. Semple and G. T. Kneebone. *Algebraic Projective Geometry*. Clarendon Press, Oxford, England, 1952.

[Szeliski, 1991] R. Szeliski. Shape from rotation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'91)*, pages 625–630, IEEE Computer Society Press, Maui, Hawaii, June 1991.

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

[Szeliski, 1994] R. Szeliski. A least squares approach to affine and projective structure and motion recovery. 1994. In preparation.

[Szeliski and Coughlan, 1994] R. Szeliski and J. Coughlan. Hierarchical spline-based image registration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, Washington, June 1994.

[Szeliski and Kang, 1994] R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[Szeliski and Tonnesen, 1992] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. *Computer Graphics (SIGGRAPH'92)*, 26(2):185–194, July 1992.

[Szeliski and Weiss, 1993] R. Szeliski and R. Weiss. Robust shape recovery from occluding contours using a linear smoother. In *Image Understanding Workshop*, pages 939–948, Morgan Kaufmann Publishers, Washington, D. C., April 1993. A longer version is available as CRL TR 93/7.

[Terzopoulos and Witkin, 1988] D. Terzopoulos and A. Witkin. Physically-based models with rigid and deformable components. *IEEE Computer Graphics and Applications*, 8(6):41–51, 1988.

[Terzopoulos *et al.*, 1987] D. Terzopoulos, A. Witkin, and M. Kass. Symmetry-seeking models and 3D object reconstruction. *International Journal of Computer Vision*, 1(3):211–221, October 1987.

[Tomasi and Kanade, 1992] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[Wang and Adelson, 1993] J. Y. A. Wang and E. H. Adelson. Layered representation for motion analysis. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pages 361–366, New York, New York, June 1993.

[Wellner, 1993] P. Wellner. Interacting with paper on the DigitalDesk. *Communications of the ACM*, 36(7):86–96, July 1993.

[Weng *et al.*, 1993] J. Weng, T. S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences.* Springer-Verlag, Berlin, 1993.

[Witkin *et al.*, 1987] A. Witkin, D. Terzopoulos, and M. Kass. Signal matching through scale space. *International Journal of Computer Vision*, 1:133–144, 1987.

[Wolberg, 1990] G. Wolberg. *Digital Image Warping.* IEEE Computer Society Press, Los Alamitos, California, 1990.

A 2-D projective transformations

Planar scene views are related by projective transformations. This is true in the most general case, i.e., for any 3-D to 2-D mapping,

$$\mathbf{u} = \mathbf{M}_{\text{cam}}\mathbf{p}, \tag{14}$$

where $\mathbf{p}$ is a 3-D world coordinate, $\mathbf{u}$ is the 2-D screen location, and $\mathbf{M}_{\text{cam}}$ is a 3×4 *camera matrix* defined in (4). Since $\mathbf{M}_{\text{cam}}$ is of rank 3, we have

$$\mathbf{p} = \mathbf{M}^{*}\mathbf{u} + s\mathbf{m} \tag{15}$$

where $\mathbf{M}^{*}$ is a left inverse of $\mathbf{M}_{\text{cam}}$ and $\mathbf{m}$ is in the null space of $\mathbf{M}_{\text{cam}}$, i.e., $\mathbf{M}_{\text{cam}}\mathbf{m} = 0$. The equation of a plane in world coordinates can be written as

$$ax + by + cz = d \quad \text{or} \quad \mathbf{n} \cdot \mathbf{p} = 0 \tag{16}$$

from which we can conclude that

$$\mathbf{n}^{T}\mathbf{M}^{*}\mathbf{u} + s\mathbf{n} \cdot \mathbf{m} = 0 \quad \text{or} \quad s = -(\mathbf{n}^{T}\mathbf{M}^{*}\mathbf{u})/(\mathbf{n} \cdot \mathbf{m}) \tag{17}$$

and hence

$$\mathbf{p} = (\mathbf{I} - (\mathbf{n} \cdot \mathbf{m})^{-1}\mathbf{m}\mathbf{n}^{T})\mathbf{M}^{*}\mathbf{u} = \tilde{\mathbf{M}}\mathbf{u}. \tag{18}$$

From any other viewpoint, we have

$$\mathbf{u}' = \mathbf{M}'_{\text{cam}}\tilde{\mathbf{M}}\mathbf{u} = \mathbf{M}_{\text{2D}}\mathbf{u}, \tag{19}$$

which is a 2-D planar projective transformation.

For the remainder of this appendix, it is convenient to assume that the world coordinate system coincides with the first camera position, i.e., $\mathbf{E} = \mathbf{I}$ and $\mathbf{M}_{\text{cam}} = \mathbf{V} = \begin{bmatrix} \hat{\mathbf{V}} & \mathbf{0} \end{bmatrix}$, and therefore $\mathbf{M}^* = \hat{\mathbf{V}}^{-1}$ and $\mathbf{m} = (0,0,0,1)^T$ in (15–18). An image point $\mathbf{u}$ then corresponds to a world coordinate $\mathbf{p}$,

$$\mathbf{p} = \begin{bmatrix} \hat{\mathbf{V}}^{-1}\mathbf{u} \\ w \end{bmatrix}, \tag{20}$$

where w is the unknown fourth coordinate of $\mathbf{p}$ (called *projective depth*) which controls how far the point is from the origin.

For panoramic image mosaicing, we rotate the point $\mathbf{p}$ around the camera optical center to obtain

$$\mathbf{p}' = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{p} = \begin{bmatrix} \mathbf{R}\hat{\mathbf{V}}^{-1}\mathbf{u} \\ w \end{bmatrix}, \tag{21}$$

with a corresponding screen coordinate

$$\mathbf{u}' = \hat{\mathbf{V}}'\mathbf{R}\hat{\mathbf{V}}^{-1}\mathbf{u} = \mathbf{M}_{\text{2D}}\mathbf{u}. \tag{22}$$

Thus, the mapping between the two screen coordinate systems can be described by a 2-D projective transformation.

For piecewise-planar scenes, the point $\mathbf{p}$ will lie on some plane k whose equation we can write as $\mathbf{n}_k \cdot \mathbf{p} = (\hat{\mathbf{n}}_k, -1) \cdot \mathbf{p} = 0$.[6] Then, (18) reduces to

$$\mathbf{p}_k = \begin{bmatrix} \mathbf{I} \\ \mathbf{n}_k^T \end{bmatrix} \hat{\mathbf{V}}^{-1}\mathbf{u} \tag{23}$$

as the 3-D position on plane k corresponding to the screen position $\mathbf{u}$. In a second camera position, the world coordinates have undergone a Euclidean transformation $\mathbf{p}'_k = \mathbf{E}\mathbf{p}_k$ with rotation $\mathbf{R}$ and translation $\mathbf{t}$. The new screen coordinates are therefore

$$\mathbf{u}'_k = \mathbf{V}'\mathbf{E}\mathbf{p}_k = \hat{\mathbf{V}}'(\mathbf{R} + \mathbf{t}\mathbf{n}_k^T)\hat{\mathbf{V}}^{-1}\mathbf{u} = \mathbf{M}_k\mathbf{u} \tag{24}$$

Thus, we see that 2-D projective motion matrices for the various planar pieces are interrelated, and are in fact determined by the plane equations ($\mathbf{n}_k$), the rigid motion ($\mathbf{R}$ and $\mathbf{t}$), and the viewing matrices ($\hat{\mathbf{V}}$ and $\hat{\mathbf{V}}'$). For methods for recovering these unknown parameters from the $\mathbf{M}_k$, see [Faugeras, 1992; Hartley, 1994].

[6] We exclude planes passing through the first camera center in this formulation.

For general depth recovery, each image point $\mathbf{u}$ corresponds to a 3-D point $\mathbf{p}$ with an unknown projective depth w as given in (20). In some other frame, whose relative orientation to the first frame is denoted by $\mathbf{E}$, we have

$$\mathbf{u}' = \mathbf{V}'\mathbf{E}\mathbf{p} = \hat{\mathbf{V}}'\mathbf{R}\hat{\mathbf{V}}^{-1}\mathbf{u} + w\hat{\mathbf{V}}'\mathbf{t} = \mathbf{M}_{\text{2D}}\mathbf{u} + w\hat{\mathbf{t}}. \tag{25}$$

Thus, the motion between the two frames can be described by our familiar 2-D planar projective motion, plus an amount of motion proportional to the projective depth along the direction $\hat{\mathbf{t}}$ (which is called the *epipole*) [Faugeras, 1992]. Note that the values of $\mathbf{M}_{\text{2D}}$, $\hat{\mathbf{t}}$, and w are not unique: there is a scale ambiguity between w and $\hat{\mathbf{t}}$, and multiples of $\hat{\mathbf{t}}$ can be added to $\mathbf{M}_{\text{2D}}$ with the appropriate adjustments to w. To remove these ambiguities, we must enforce the rigidity constraints $\mathbf{M}_{\text{2D}} = \hat{\mathbf{V}}'\mathbf{R}\hat{\mathbf{V}}^{-1}$ [Szeliski, 1994].