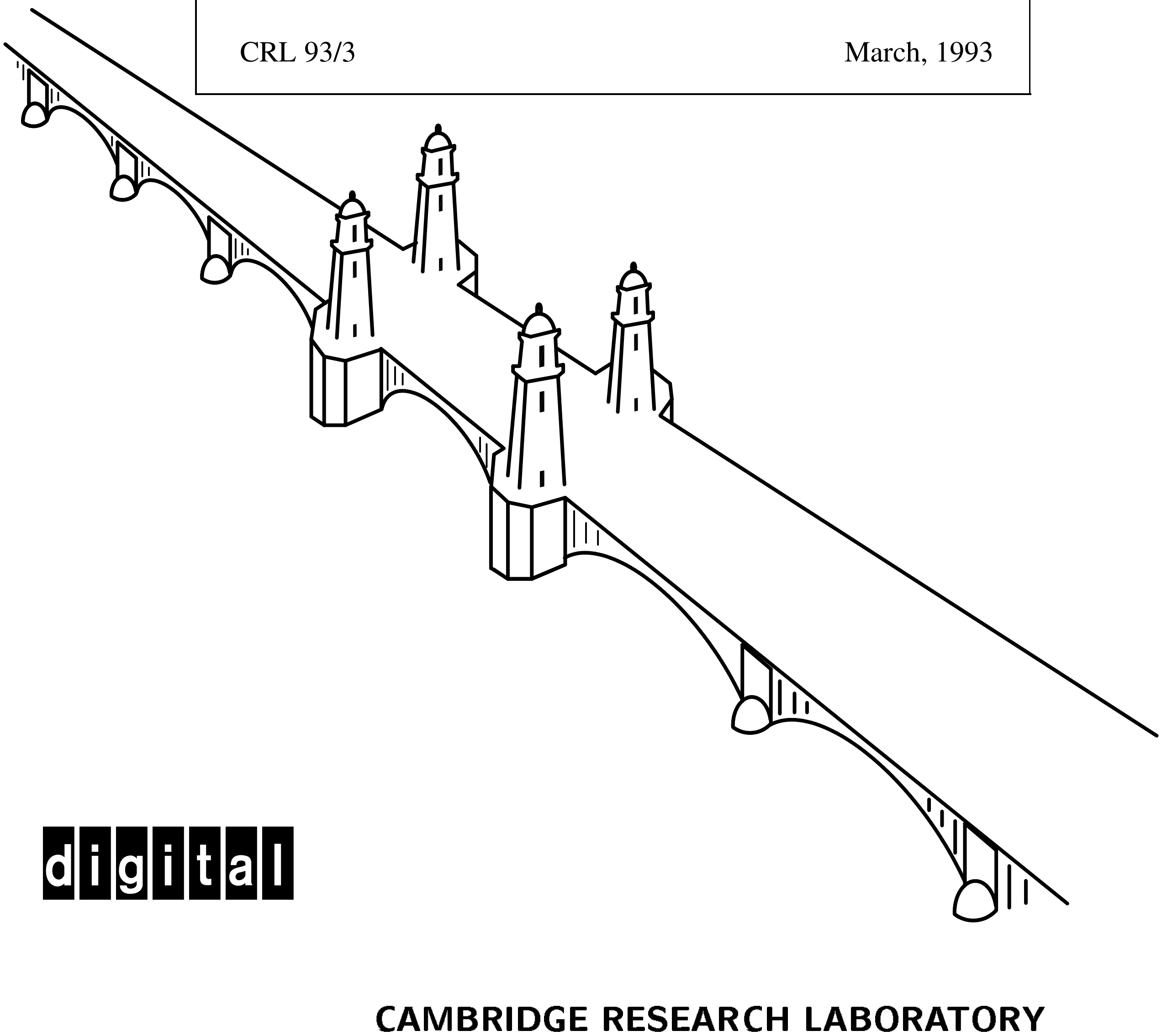
Recovering 3D Shape and Motion from Image Streams using Non-Linear Least Squares

Richard Szeliski and Sing Bing Kang
Digital Equipment Corporation
Cambridge Research Lab

CRL 93/3 March, 1993



digital

CAMBRIDGE RESEARCH LABORATORY
Technical Report Series

digital™

Cambridge Research Laboratory
One Kendall Square
Cambridge, Massachusetts 02139

Recovering 3D Shape and Motion from Image Streams using Non-Linear Least Squares

Richard Szeliski and Sing Bing Kang[1]

Digital Equipment Corporation

Cambridge Research Lab

CRL 93/3 March, 1993

Abstract

The simultaneous recovery of 3D shape and motion from image sequences is one of the more difficult problems in computer vision. Classical approaches to the problem rely on using algebraic techniques to solve for these unknowns given two or more images. More recently, a batch analysis of *image streams* (the temporal tracks of distinguishable image features) under orthography has resulted in highly accurate reconstructions. We generalize this approach to perspective projection and partial or uncertain tracks by using a non-linear least squares technique. While our approach requires iteration, it quickly converges to the desired solution, even in the absence of *a priori* knowledge about the shape or motion. Important features of the algorithm include its ability to handle partial point tracks, to use line segment matches and point matches simultaneously, and to use an object-centered representation for faster and more accurate structure and motion recovery. We also show how a projective (as opposed to scaled rigid) structure can be recovered when the camera calibration parameters are unknown.

Keywords: Computer vision, image sequence analysis, structure from motion, non-linear least squares fitting.

[1]The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213-3890

Contents

List of Figures

List of Tables

1 Introduction

This paper addresses the problem of extracting both 3D structure (shape) and object or camera motion simultaneously from a given image sequence. Recovering shape and motion is a difficult but important task, and has wide applicability in many areas such as robot navigation, non-tactile parts inspection, manipulation, and CAD.

Approaches to this problem range from the classical methods which use only two frames and a few points [Ullman, 1979; Longuet-Higgins, 1981; Tsai and Huang, 1984] to methods which use many frames and points [Debrunner and Ahuja, 1990; Tomasi and Kanade, 1991; Taylor *et al.*, 1991]. Tomasi and Kanade [1991] have obtained highly accurate results using a factorization method to extract object-centered shape and motion under orthography. More recently, Taylor, Kriegman, and Anandan [1991] developed a non-linear least squares fitting algorithm for 2D shape and motion recovery under perspective.

Our approach applies a similar non-linear least squares technique to recover 3D shape and motion from *image streams* (the temporal tracks of point-like image features) without *a priori* information about the shape or motion. Least squares makes optimal use of each measurement and guarantees a statistically optimal estimate in the vicinity of the true solution. Furthermore, it avoids the potentially unlimited noise amplification which may occur with arbitrary algebraic manipulation. The least squares formulation also enables us to deal easily with perspective or arbitrary camera models, partial and/or uncertain tracks, and even to simultaneously use point and line correspondences. Finally, it has the virtue of simplicity, since we use a general-purpose optimization technique (Levenberg-Marquardt) [Press *et al.*, 1992] which only requires an error computation and error gradient backpropagation at each step.

Our results on both synthetic and real data indicate that the algorithm normally converges even when no *a priori* information about shape or motion is given. However, the choice of parametrization and order of solution can be important. We have also found that simultaneously solving for structure and motion (unlike [Taylor *et al.*, 1991]) increases the convergence rate towards the solution.

We begin the paper with a brief review of previous work in structure from motion (Section 2). In Section 3, we present the equations governing how image points depend on the structure and motion parameters. Section 4 reviews the Levenberg-Marquardt algorithm and derives the equations necessary for its implementation. Section 5 extends our formulation to include line correspondences. Section 6 presents the tracking algorithms we use to compute the image streams. In Section 7, we demonstrate how our algorithm performs on the easier problem of camera calibration (i.e., known structure). We then present results on synthetic data for pure rotation and general motion (Section 8), and on real image sequences (Section 9). We close with a discussion of the algorithm's performance and topics for future research.

2 Previous work

The structure from motion problem has been extensively studied in computer vision. Early papers on this subject [Longuet-Higgins, 1981; Tsai and Huang, 1984] develop algorithms to compute the structure and motion from a small set of points matched in two frames. Longuet-Higgins [1981] presents a closed-form solution to the problem using eight 3D points and two frames with the assumption that the correspondence problem has been solved. Tsai and Huang [1984] present similar work, but in addition consider the uniqueness issues in the determination of the motion parameters. Others have extended the essential parameter approach to lines [Faugeras *et al.*, 1987; Spetsakis and Aloimonos, 1990], performed more detailed error analyses [Weng *et al.*, 1989a; Weng *et al.*, 1993], and developed non-linear least squares (*optimal estimation*) techniques for the two-frame problem [Weng *et al.*, 1989b].

Recent research focuses on extraction of shape and motion from longer image sequences [Kumar *et al.*, 1989; Debrunner and Ahuja, 1990; Cui *et al.*, 1990; Tomasi and Kanade, 1990; Tomasi and Kanade, 1991; Chen and Tsuji, 1992]. Debrunner and Ahuja [1990] provide closed-form expressions for shape and motion assuming that motion is constant over the sequence (see also [Broida and Chellappa, 1991; Kumar *et al.*, 1989; Weng *et al.*, 1993]). Incremental solutions for multiple motions are computed by taking advantage of the redundancy of measurements. Cui, Weng, and Cohen [1990] use an optimal estimation technique (non-linear least squares) between each pair of frames, and an extended Kalman filter to accumulate information over time.

Tomasi and Kanade [1991] use a factorization method which extracts shape and motion from an image stream without computing camera-centered depth. Their approach formulates the shape from motion problem in object-centered coordinates, unlike the more conventional camera-centered formulation. Tomasi and Kanade's method is based on the assumption of orthography and processes all of the frames simultaneously, i.e., it is a *batch* approach. Chen and Tsuji [1992] relax the assumption of orthography by analyzing the image sequence through its temporal and spatial subparts.

Taylor, Kriegman and Anandan [1991] formulate the shape from motion task as a non-linear least squares problem in which the Euclidean distance between the estimated and actual positions of the points in the image sequence is minimized. They restrict their analysis to a 2-D environment (a mobile robot moving horizontally in a room) and a 1-D retina, and use the Levenberg-Marquardt algorithm to compute the locally optimal solution starting with motion estimates based on odometry.

Our approach is a combination of several of the above techniques. We apply least squares (optimal estimation) directly to the whole image sequence and use an object-centered representation [Tomasi and Kanade, 1991]. We use perspective projection [Taylor *et al.*, 1991] and partial point tracks, extend the formulation to include lines, and show that the algorithm converges without requiring an algebraic reconstruction technique for initialization (as in [Weng *et al.*, 1989b]).

Within our framework, camera calibration can be viewed as a simplified version of structure from motion, where the structure component is known *a priori*. Tsai [1987] presents a good review of the camera calibration literature. Our approach is related to [Gennery, 1979; Tsai, 1987; Gennery, 1991], which all employ iterative least squares to recover camera parameters.

3 General problem formulation

The problem addressed in the paper is the recovery of a set of 3-D structure parameters $\mathbf{p}_i$ and time-varying motion parameters T_j from a set of observed image features $\mathbf{u}_{ij}$. In this section, we present the forward equations, i.e., the rigid body and perspective transformations which map 3D points into 2D image points, and discuss potential ambiguities in the recovered parameters. In Section 4 we will discuss how to estimate the $\mathbf{p}_i$ and T_j which best satisfy these forward equations.

3.1 Rigid body transformations

The general equation linking a 2D image feature location $\mathbf{u}_{ij}$ in frame j to its 3D position $\mathbf{p}_i$ (i is the track index) is

$$\mathbf{u}_{ij} = \mathcal{P}\left(T_j^{(K)} \ldots T_j^{(1)} \mathbf{p}_i\right) \tag{1}$$

where the perspective projection transformation $\mathcal{P}()$ (defined in 3.2) is applied to a cascaded series of rigid transformation $T_j^{(k)}$. Each transformation is in turn defined by

$$T_j^{(k)} \mathbf{x} = \mathbf{R}_j^{(k)} \mathbf{x} + \mathbf{t}_j^{(k)} \tag{2}$$

where $\mathbf{R}_j^{(k)}$ is a rotation matrix and $\mathbf{t}_j^{(k)}$ is a translation applied after the rotation. We represent each rotation by a quaternion $\mathbf{q} = [w, (q_0, q_1, q_2)]$ with a corresponding rotation matrix

$$\mathbf{R}(\mathbf{q}) = \begin{pmatrix} 1 - 2q_1^2 - 2q_2^2 & 2q_0q_1 + 2wq_2 & 2q_0q_2 - 2wq_1 \\ 2q_0q_1 - 2wq_2 & 1 - 2q_0^2 - 2q_2^2 & 2q_1q_2 + 2wq_0 \\ 2q_0q_2 + 2wq_1 & 2q_1q_2 - 2wq_0 & 1 - 2q_0^2 - 2q_1^2 \end{pmatrix} \tag{3}$$

(alternative representations for rotations are discussed in [Ayache, 1991]). The cascaded nature of equations (1) and (2) allow for both object- and camera-centered transformation.

Within each of the cascaded transforms, the motion parameters may be time-varying (the j subscript is present) or fixed (the subscript is dropped). A less general form of (1) that we use in many of our experiments is given by

$$\mathbf{u}_{ij} = \mathcal{P}\left[\mathbf{R}\left(\mathbf{R}_j(\mathbf{p}_i - \mathbf{c}) + \mathbf{t}_j\right)\right) + \mathbf{t}] \tag{4}$$

where $\mathbf{R}$ gives the camera tilt relative to the object, $\mathbf{c}$ is the object's displacement from the axis of rotation, and $\mathbf{t}$ is the (fixed) distance to the intermediate motion frame (Figure 1). The time varying motion parameters $\mathbf{R}_j$ and $\mathbf{t}_j$ are represented relative to an intermediate frame (Figure 1), but this could easily be modified to a strictly object-centered or camera-centered reference frame. As an example, if we are trying to calibrate a camera by looking at a known object rotating on a turntable (Section 7, Figure 4), we can set $\mathbf{t}_j = 0$ and restrict $\mathbf{R}_j$ to have rotation only around the z axis.

3.2 Perspective projection

The standard perspective projection equation used in computer vision is

$$\begin{pmatrix} u \\ v \end{pmatrix} = \mathcal{P}_1 \begin{pmatrix} x \\ y \\ z \end{pmatrix} \equiv \begin{pmatrix} f\frac{x}{z} \\ f\frac{y}{z} \end{pmatrix} \tag{5}$$

where f is a product of the focal length of the camera and the pixel array scale factor (we assume that pixels are square, since this has been verified experimentally for our camera).

An alternative formulation which we use in this paper is

$$\begin{pmatrix} u \\ v \end{pmatrix} = \mathcal{P}_2 \begin{pmatrix} x \\ y \\ z \end{pmatrix} \equiv \begin{pmatrix} s\frac{x}{1+\eta z} \\ s\frac{y}{1+\eta z} \end{pmatrix} \tag{6}$$

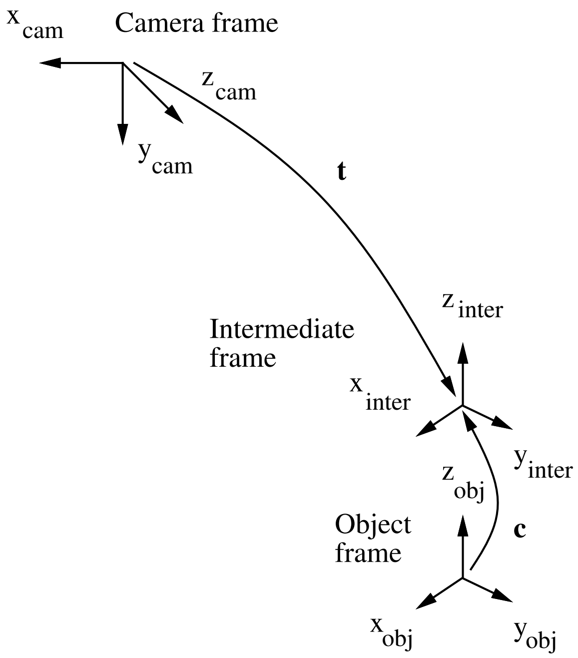


Figure 1: Transformation between camera and object coordinate frames

Here, we assume that the (x, y, z) coordinates before projection are with respect to a reference frame that has been displaced away from the camera by a distance t_z along the optical axis,[1] with $s = f/t_z$ and $\eta = 1/t_z$. The projection parameter s can be interpreted as a *scale factor* and η as a *perspective distortion factor*. Our alternative perspective formulation results in a more robust recovery of camera parameters under weak perspective, where $\eta \ll 1$ and $\mathcal{P}(x, y, z)^T \approx (sx, sy)^T$, since s can be much more reliably recovered than η (in the old formulation, f and t_z are very highly correlated).

3.3 General projection equations

An alternative to the rigid motion plus fixed perspective projection equations presented above is a formulation which only attempts to recover the *projective* structure of the world [Mohr *et al.*, 1992; Faugeras, 1992; Shashua, 1992; Shashua, 1993]. In this formulation, we use the imaging

[1]If we wish, we can view t_z as the z component of the original global translation $\mathbf{t}$ which is absorbed into the projection equation, and then set the third component of $\mathbf{t}$ to zero.

equations

$$\mathbf{u}_{ij} = \mathcal{P}_2\left(\mathbf{M}_j\mathbf{p}_i + \mathbf{t}_j\right) = \begin{pmatrix} \frac{m_{00}x_i+m_{01}y_i+m_{02}z_i+t_x}{m_{20}x_i+m_{21}y_i+m_{22}z_i+1} \\ \frac{m_{10}x_i+m_{11}y_i+m_{12}z_i+t_y}{m_{20}x_i+m_{21}y_i+m_{22}z_i+1} \end{pmatrix} \tag{7}$$

where $\mathbf{M}_j = [m_{pq}]$ are arbitrary (non-orthogonal) matrices, $\mathbf{t}_j = (t_x, t_y, 0)^T$, and $s = \eta = 1$ in (6). The structure in this case is recovered to within an unknown projective transform of the true structure. This alternative formulation may be acceptable in certain applications (e.g., recognition, or the testing of co-planarity). The projective structure can be converted into the true Euclidean structure given sufficient domain knowledge (e.g., the knowledge about the angles between recovered planes and/or distances between points). Alternatively, the rigid structure might also be recovered by applying a *post hoc* rigidity constraint to the recovered (non-rigid) motion parameters. A more detailed description of our projective reconstruction algorithm can be found in [Szeliski, 1993].

3.4 Ambiguities in the recovered parameters

The set of imaging equations introduced in sections 3.1 to 3.3 are very general. This allows us to easily specialize the equations to a particular setting (e.g., the calibration problem) and to study the tradeoffs between various parametrizations (by fixing or eliminating certain parameters). However, it is often the case that there are more free parameters than can theoretically be recovered from the data. Since we are using a stabilized local gradient descent, the extra degrees of freedom should not affect the quality of the final solution, and, in our experience, often help speed the convergence to a good estimate. If such ambiguities are not acceptable in a given application, the equations can always be specialized until a unique solution is guaranteed (up to unavoidable ambiguities, such as the scale ambiguity in pure structure from motion [Longuet-Higgins, 1981]).

4 Least squares minimization

To solve for the structure and motion parameters, we use the iterative Levenberg-Marquardt algorithm. While Levenberg-Marquardt will only find a locally optimal solution, our experiments indicate that it normally converges to the correct solution even with a very simple initialization (Sections 8 and 9). In this section, we present the Levenberg-Marquardt algorithm, the equations for the required derivatives, our matrix layout and inversion algorithm, and techniques to analyze and use the uncertainty in the estimates.

4.1 The Levenberg-Marquardt algorithm

The Levenberg-Marquardt method is a standard non-linear least squares technique [Press *et al.*, 1992] that works very well in a wide range of situations. It provides a way to vary smoothly between the inverse-Hessian method and the steepest descent method.

The merit or objective function that we minimize is

$$\mathcal{C}(\mathbf{a}) = \sum_i \sum_j c_{ij} \left| \mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij}) \right|^2, \tag{8}$$

where $\mathbf{f}()$ is given in (1) and

$$\mathbf{a}_{ij} = \left(\mathbf{p}_i^T, \mathbf{m}_j^T, \mathbf{m}_g^T \right)^T \tag{9}$$

is the vector of structure and motion parameters which determine the image of point i in frame j. The vector $\mathbf{a}$ contains all of the unknown structure and motion parameters, including the 3D points $\mathbf{p}_i$, the time-dependent motion parameters $\mathbf{m}_j$, and the global motion/calibration parameters $\mathbf{m}_g$. The weight c_{ij} in (8) describes our confidence in measurement $\mathbf{u}_{ij}$, and is normally set to the inverse variance σ_{ij}^{-2}.

An inherent feature of this formulation is its ability to cope with partial information (by simply setting $c_{ij} = 0$ when information is unavailable). This is important, since in processing image streams, only partial tracks corresponding to the motion of 3D points may be available, both because the points may not be visible in all frames and because the tracking may be unreliable.

The Levenberg-Marquardt algorithm first forms the approximate Hessian matrix

$$\mathbf{A} = \sum_i \sum_j c_{ij} \frac{\partial \mathbf{f}^T(\mathbf{a}_{ij})}{\partial \mathbf{a}} \frac{\partial \mathbf{f}(\mathbf{a}_{ij})}{\partial \mathbf{a}^T} \tag{10}$$

and the weighted gradient vector

$$\mathbf{b} = -\sum_i \sum_j c_{ij} \frac{\partial \mathbf{f}^T(\mathbf{a}_{ij})}{\partial \mathbf{a}} \mathbf{e}_{ij}, \tag{11}$$

where $\mathbf{e}_{ij} = \mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij})$ is the image plane error of point i in frame j. Given a current estimate of $\mathbf{a}$, it computes an increment $\delta\mathbf{a}$ towards the local minimum by solving

$$(\mathbf{A} + \lambda \mathbf{I})\delta\mathbf{a} = -\mathbf{b}, \tag{12}$$

where λ is a stabilizing factor which varies over time [Press *et al.*, 1992]. Note that the matrix $\mathbf{A}$ is an approximation to the Hessian matrix, as the second-derivative terms are left out. As mentioned in [Press *et al.*, 1992], inclusion of these terms can be destabilizing if the model fits badly or is contaminated by outlier points.

To compute the required derivatives for (10) and (11), we compute derivatives with respect to each of the fundamental operations (perspective projection, rotation, translation) and apply the chain rule. The equations for each of the basic derivatives are given in Appendix A.

To solve the set of equations (12), we arrange the parameter vector $\mathbf{a}$ with all of the structure terms $\mathbf{p}_i$ followed by the time varying motion terms $\mathbf{m}_j$ followed by the global transform (camera) parameters $\mathbf{m}_g$. This leads to a sparse and partially block diagonal Hessian matrix $\mathbf{A}$ (Appendix B) which we store in the space-efficient skyline form [Bathe and Wilson, 1976]. We then use LDU decomposition [Press *et al.*, 1992], since this minimizes the amount of *fill-in* during the system solution [Bathe and Wilson, 1976].

4.2 Statistical interpretation and robust estimation

The weighted least squares formulation produces the minimum variance (and maximum likelihood) estimate for the unknown parameters under the assumption that each measurement is contaminated with additive Gaussian noise [Bierman, 1977]. In our structure from motion application, we can quantify the amount of error in the tracked feature locations by analyzing the response of the tracker (e.g., the shape of the correlation surface [Anandan, 1989]).

The statistical formulation enables us to make our technique more *robust*, by discarding or down-weighting measurements whose *residual errors* are too large [Huber, 1981]. In our implementation of the Levenberg-Marquardt algorithm, after an initial convergence has been achieved, we robustify the estimates by computing all of the residuals, throwing out measurements where $|\mathbf{e}_{ij}| > 3\sigma_i$, and performing some more iteration.[2] The statistical formulation can also be used to compute $\mathbf{P}$, the covariance matrix of $\mathbf{a}$, by simply inverting the Hessian matrix $\mathbf{A}$ (assuming that equations were properly weighted with $c_{ij} = \sigma_{ij}^{-2}$).

[2] This technique is called using *metrically Winsorised residuals* [Huber, 1981].

5 Using line segment correspondences

Recovering structure and motion from line segment correspondences provides another powerful mechanism for recovering the environment in situations where straight lines are preponderant (e.g., in man-made indoor environments) [Faugeras *et al.*, 1987; Spetsakis and Aloimonos, 1990; Ayache, 1991]. In many cases, matching or tracking lines may be significantly easier than identifying reliably trackable point features such as corners.

The least squares formulation developed in the previous section can easily be extended to take advantage of tracked line segments. While many representations are possible for line segments [Ayache, 1991], we choose to represent each segment by its two endpoints $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$. We replace selected terms in our weighted cost measure (8) with

$$c_{ij}[\hat{\mathbf{n}}_{ij} \cdot (\mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij}))]^2 \tag{13}$$

where $\hat{\mathbf{n}}_{ij}$ is normal to the 2D line segment orientation observed for line segment $(i, i+1)$ in frame j. Thus, only the motion perpendicular to the line segment is used, which provides only a single constraint (rather than the usual two) on the structure and motion parameters,[3] making this problem more difficult than point-based recovery.

A slightly more general form of (13) which can also work with (8) is

$$(\mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij}))^T \mathbf{C}_{ij} (\mathbf{u}_{ij} - \mathbf{f}(\mathbf{a}_{ij})) \tag{14}$$

where $\mathbf{C}_{ij}$ is the inverse covariance matrix for each measurement. In the case of line segment matches, we can make the constraint along the line segment to be much weaker (i.e., $\sigma_{||}^{-2} \ll \sigma_{\perp}^{-2}$), and still achieve some (weak) localization of 3D line segment endpoints.

6 Image streams: feature detection and tracking

To track point features from frame to frame, we use a relatively simple algorithm based on the *monotonicity operator* [Kories and Zimmermann, 1986], which computes the number of neighboring pixels whose intensity is less than that of the central pixel. The monotonicity operator maps each pixel in an intensity image into one of the nine classes. Pixels of the same class within the

[3] Under known motion, it constrains the endpoint to lie in a plane rather than along a line.

same vicinity tend to form blobs. The centroids of these blobs can then be tracked for motion detection, and in our case, for structure and motion recovery.

As in [Kories and Zimmermann, 1986], the image is first bandpass filtered to suppress both high and low frequencies, which increases the stability of the blobs detected. In addition, we impose a *deadband* of a few pixels (n_{dead}) to reduce the effects of noise, i.e., we do not rely on a pixel if the difference in the intensity between itself and its neighbors is less than n_{dead}. While this results in fewer blobs to track, most of the blobs created by noise are eliminated as well. Two examples of point feature tracking using the monotonicity operator are shown in Figures 2 and 3.

7 Camera calibration

The general structure from motion algorithm developed in this paper can easily be specialized to the simpler camera calibration problem by simply setting the 3-D point locations to their known values. Camera calibration is often a necessary first step in many computer vision applications, including structure from motion (but see, e.g., [Mohr *et al.*, 1992; Faugeras, 1992; Demey *et al.*, 1992; Shashua, 1992; Shashua, 1993; Szeliski, 1993] for techniques that do not require calibration). This simplified problem is also a good way to test out the general recovery algorithm and to determine limits on its performance. Below, we present our methods for tracking the calibration pattern, determining the focal length and distance to pattern (intrinsic parameters) and for computing object motion (extrinsic parameters).

7.1 Experimental setup

The setup for camera calibration is shown in Figure 4. The camera is facing the turntable at an oblique angle with the whole calibration pattern in its field of view. Referring to (4), the unknown parameters are $\mathbf{R}$, $\mathbf{R}_j$ for each frame, $\mathbf{c}$, $\mathbf{t}$ and f, the focal length. The 3D point positions ($\mathbf{p}_i$'s) are known *a priori* while the translations $\mathbf{t}_j$'s are set to zero, since only rotations are expected in this procedure. To initialize the non-linear least squares, we compute a guess for the translations and tilt angle from the locations of the center and corner dots.

The calibration pattern that we use is a flat 7 by 7 square grid of points with a spacing of 1.9 cm (Figure 5). Prior to detecting and tracking each grid point, the image is first histogram equalized to increase contrast between the dots and the white background. Detection of each dot is done using

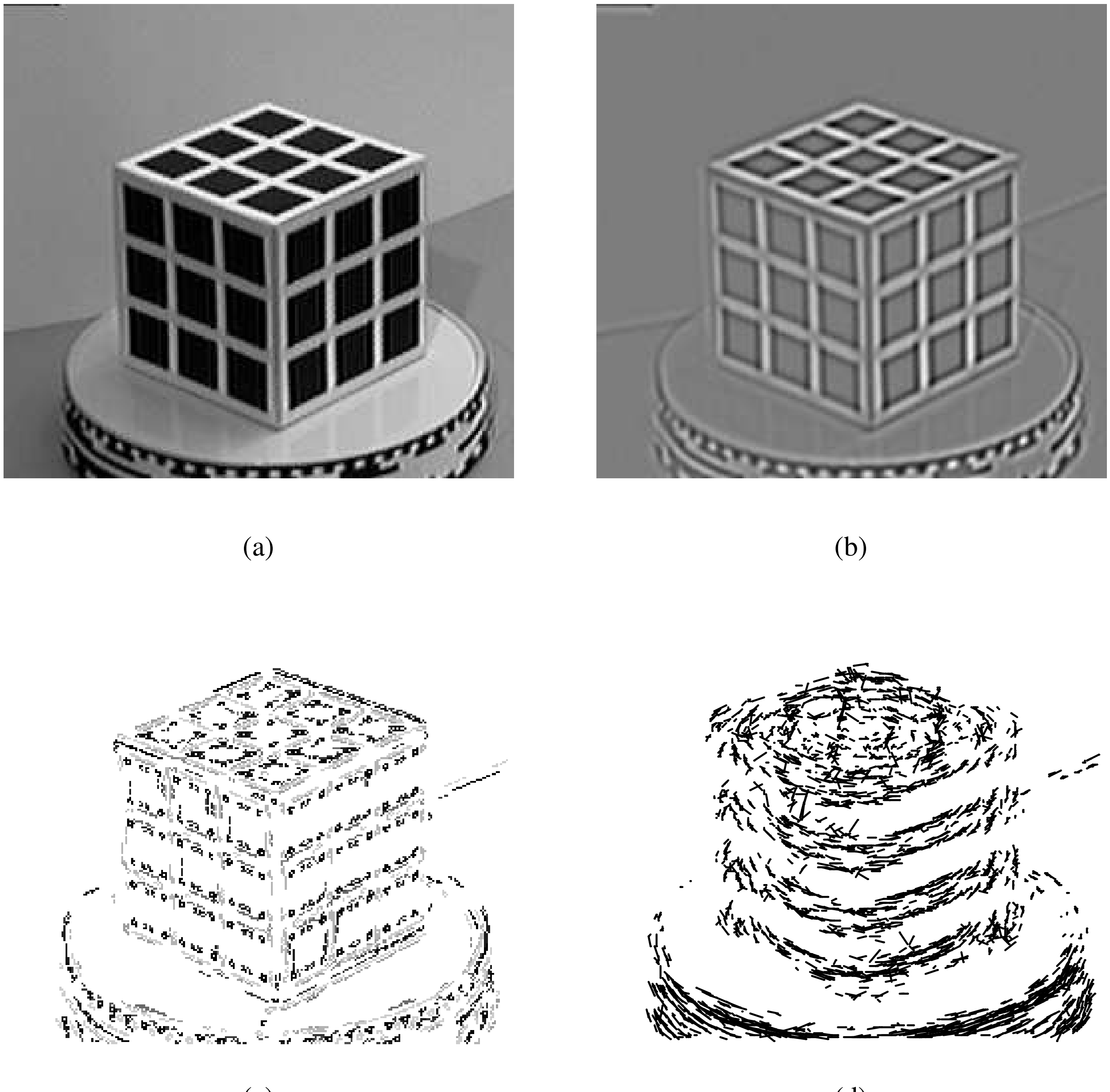
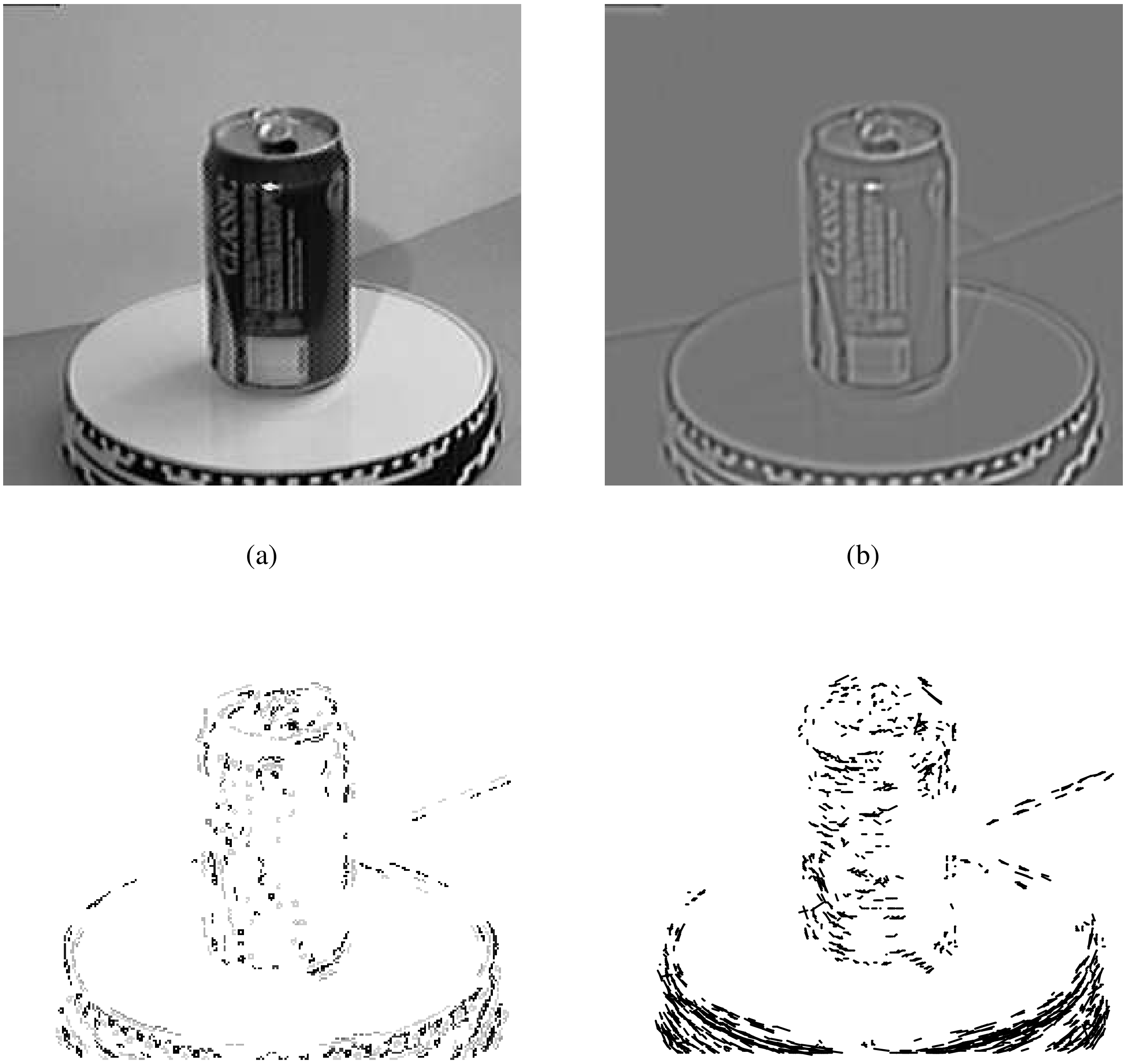


Figure 2: Application of the monotonicity operator - cube scene
(a) Original image at frame 10 (b) Bandwidth filtered image (c) Output of monotonicity operator (d) Tracks found for the first 10 frames



(c) (d)

Figure 3: Application of the monotonicity operator - coke can scene
(a) Original image at frame 10 (b) Bandwidth filtered image (c) Output of monotonicity operator (d) Tracks found for the first 10 frames

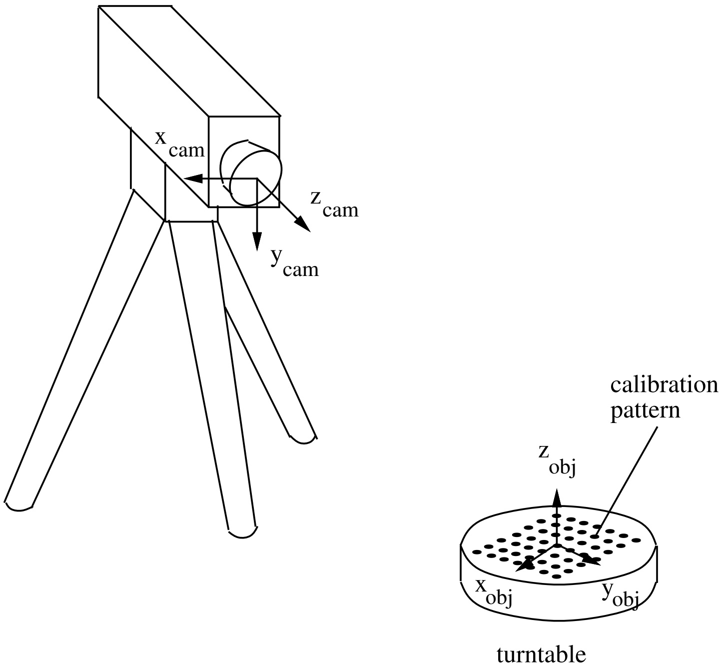


Figure 4: Setup for camera calibration

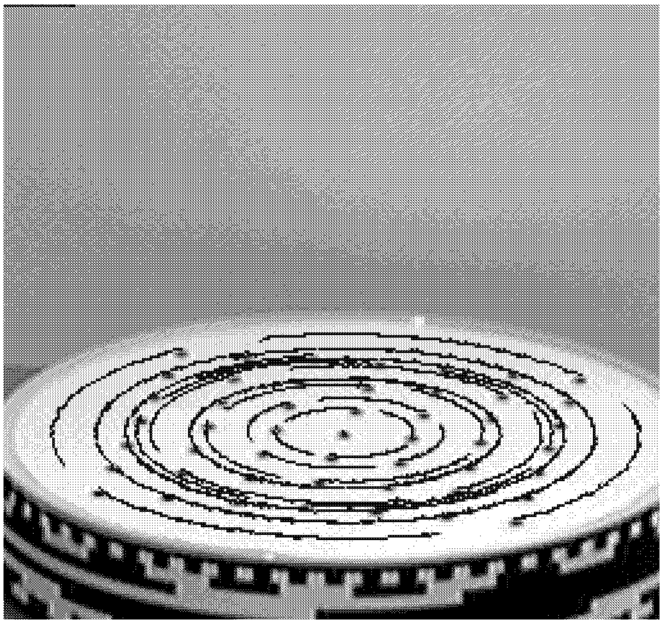


Figure 5: Calibration pattern and the track paths for 50 frames

a star-shaped template whose conical weight distribution is such that the weight is the smallest at the center (weight of -8) and linearly increasing to 1 away from the center:

$$
\mathrm{T}_{star} = \frac{1}{5}\begin{bmatrix}
5 & & & & & 5 & & & & & 5 \\
 & 3 & & & & 3 & & & & 3 & \\
 & & 1 & & & 1 & & & 1 & & \\
 & & & -1 & & -1 & & -1 & & & \\
 & & & & -3 & -3 & -3 & & & & \\
5 & 3 & 1 & -1 & -3 & -40 & -3 & -1 & 1 & 3 & 5 \\
 & & & & -3 & -3 & -3 & & & & \\
 & & & -1 & & -1 & & -1 & & & \\
 & & 1 & & & 1 & & & 1 & & \\
 & 3 & & & & 3 & & & & 3 & \\
5 & & & & & 5 & & & & & 5
\end{bmatrix}.
$$

In the first frame, the whole image is searched for potential dots. Those which are too small or too large are filtered out. In subsequent frames, every potential dot is searched for within a neighborhood of its previous position. The result of this procedure is shown in Figure 5. In this figure, the tracks are drawn over the first image of the calibration pattern and turntable.

7.2 Results

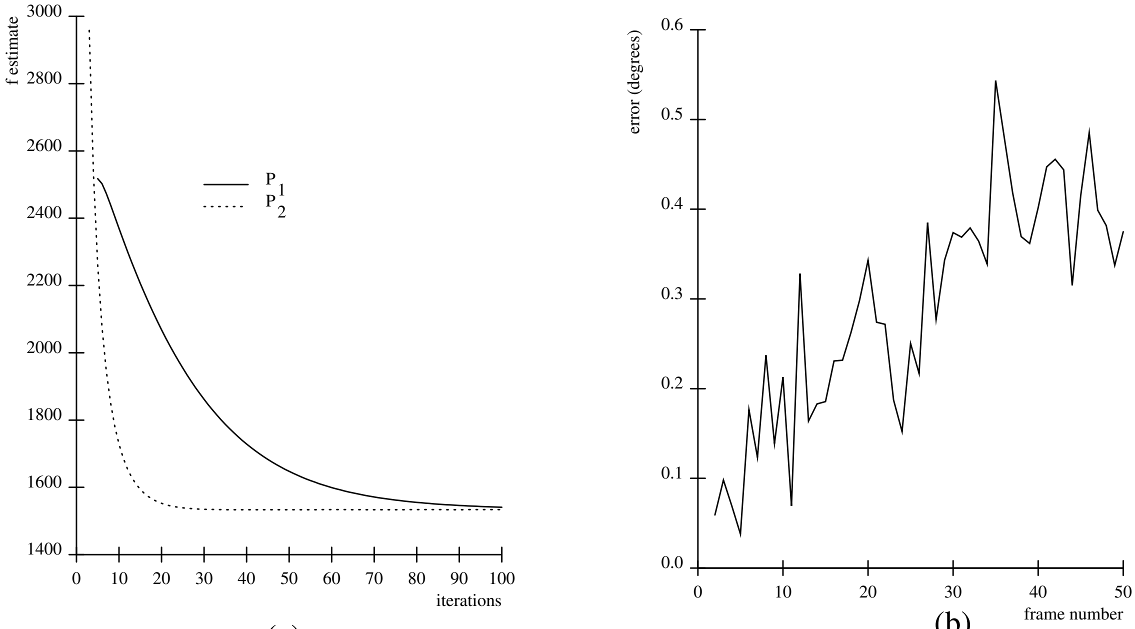
The results of our calibration algorithm applied to the tracked point centers are given in Table 1. From these results, we can see that the tilt and distance to the turntable are both recovered accurately, even though the field of view is small (5.2” pattern / 59.4” distance $\approx 5^\circ$). Our experiments also indicate that the type of perspective projection function used significantly influences the rate of convergence. This can be seen from Figure 6a, which compares how quickly the estimated focal length settles to a constant value.

We compared the recovered rotations using the least squares formulation with those extracted using a program that tracks and interprets the Gray coded pattern on the side of the turntable [Szeliski, 1990]. Our results indicate that the root mean squared (RMS) error in angle estimates between the two techniques is about 0.2° (Figure 6b). It is unclear from these results which algorithm has a greater error in motion estimation, but the worst-case RMS error of either algorithm is at most 0.2°.

Parameter	1 frame	48 frames
f	1569.0	1536.6
$\mathbf{t}$	(0.33, 2.29, 58.30)	(0.31, 2.29, 57.22)
d = $\lvert\mathbf{t}\rvert$ (estimated)	58.3"	57.3"
d (measured)	59.4"	59.4"
$\mathbf{c}$	(0, 0, 0)	(-0.02, 0.03, 0)
$\mathbf{q}_x$	[0.537, (0.843, -0.022, 0)]	[0.536, (0.844, -0.022, 0)]
θ_x (estimated)	115.0°	115.2°
θ_x (measured)	117.9°	117.9°

Table 1: Comparison of camera calibration results for 1 frame and 48 frames.

Note that the parameter $\mathbf{c}$ for one frame calibration has been set inactive, otherwise there will be redundant degrees of freedom. The z-component of $\mathbf{c}$ for the multiple frame calibration has been set inactive for the same reason. The x- and y-components are not set inactive as there may be some offset along the z-plane in the object frame.



(a) (b)

Figure 6: Convergence rates of focal length estimation and estimation errors.
(a) comparing convergence rates of focal length vs. iteration number for f and t_z ($\mathcal{P}_1$), and s and η ($\mathcal{P}_2$) perspective projection representations (48 frames used); (b) difference between Gray code and calibration pattern angle estimates (angles range over 90°).

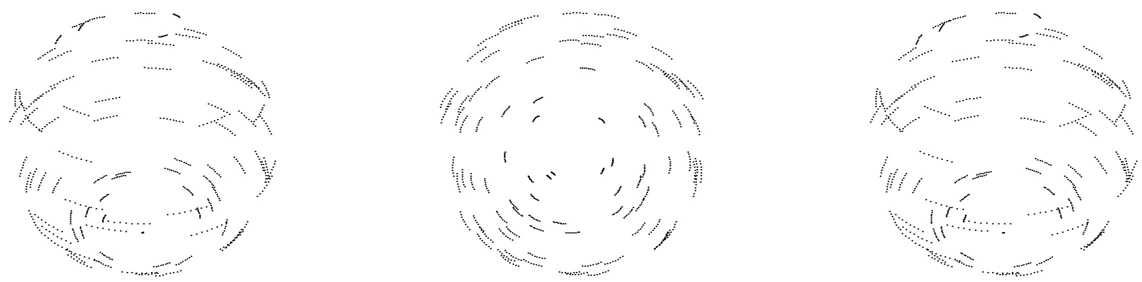

(a) (b) (c)

Figure 7: Synthetic point tracks
(a) observed tracks; current track estimates (b) after 1 iteration and (c) after 5 iterations. 96 points, 8 frames, $\Delta\theta = 2°$.

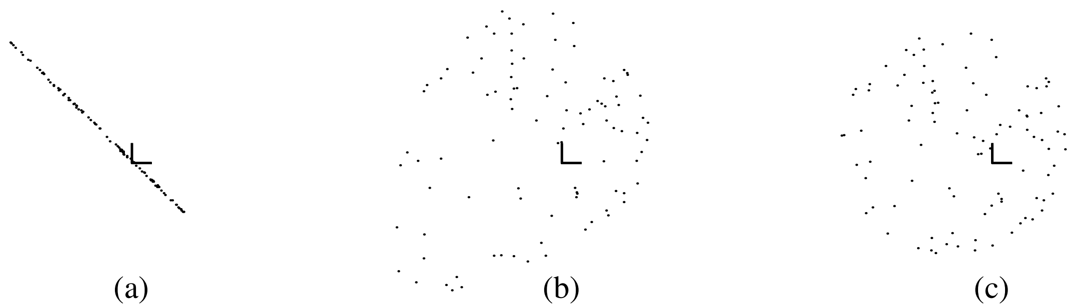
8 Experimental results: synthetic data

We now present the results of our algorithm running on synthetic motion sequences of objects undergoing pure object-centered rotation and general motion. The case of pure rotation corresponds to the problem solved (under orthography) by Tomasi and Kanade [1991] (see also [Sawhney *et al.*, 1990] for the analysis of rotational motion). The 3D point locations are initialized by projecting the 2D image point locations in the middle frame to a constant depth plane which passes through the object coordinate frame. The rotation quaternions are set to unit scalars and the translations are set to zero.

8.1 Pure rotation

Our synthetic data set consists of a set of n points randomly distributed over the surface of a sphere. The tracks resulting from rotating the sphere about its z axis, with the camera located 45° above the x-y plane, are shown in Figure 7a. Figures 7b and 7c show the image plane trajectories corresponding to the current shape and motion estimates. The minimization of the squared distance between the predicted and observed tracks is what drives our shape and motion recovery algorithm. By observing the two sets of tracks simultaneously, we can gauge the speed of convergence of the algorithm and observe potential local minima.

The reconstruction process can also be examined by viewing the 3D positions of the true and estimated point locations using an interactive 3D display window. The initial estimate of the 3D point locations is a plane, as can be seen from the side view in Figure 8a. When we start the



(a) (b) (c)

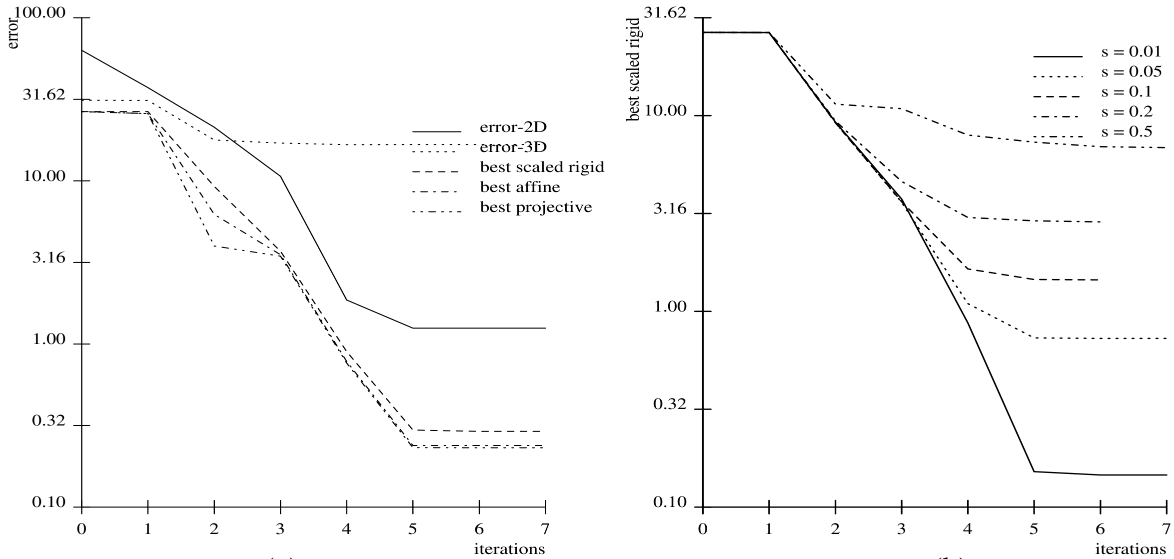
Figure 8: Side view of 3D point position estimates
After (a) 0 iterations, (b) 2 iterations, and (c) 20 iterations. 96 points, 8 frames, $\Delta\theta = 2°$.

reconstruction algorithm, the 3D shape quickly approaches the correct spherical shape (Figures 8b and 8c).

To obtain a more quantitative measure of the algorithm's convergence, we can plot both the 2D image plane error (8) and the 3D error between the true and estimated point positions (Figure 9a). The errors plotted are the average RMS errors per point, in pixels for 2D errors, and in world coordinates for 3D errors (the sphere diameter is 100 units). While the true object motion is purely rotational, we estimate both a rotation and translation for each frame, as in traditional structure from motion. The recovered shape cannot therefore be recovered unambiguously, and is related to the true shape by a scaled rigid transform [Longuet-Higgins, 1981]. The error after computing the best scaled rigid match between the estimated and true point positions is shown in Figure 9a, along with the errors after computing the best affine and projective transforms between the two data sets.

In this figure, we notice that the projective and affine structures are recovered better than the scaled rigid structure, even though our algorithm specifically is designed to recover the rigid structure (this difference is less obvious when more frames or larger displacements are used). We also notice that the image plane error reduces to its final value relatively quickly (5–10 iterations), while the 3D structure continues to improve. It is also interesting that the 3D structure does not "wander away" too far from its initial planar estimate (Figure 8), and does not therefore acquire a grossly erroneous estimate of scale.

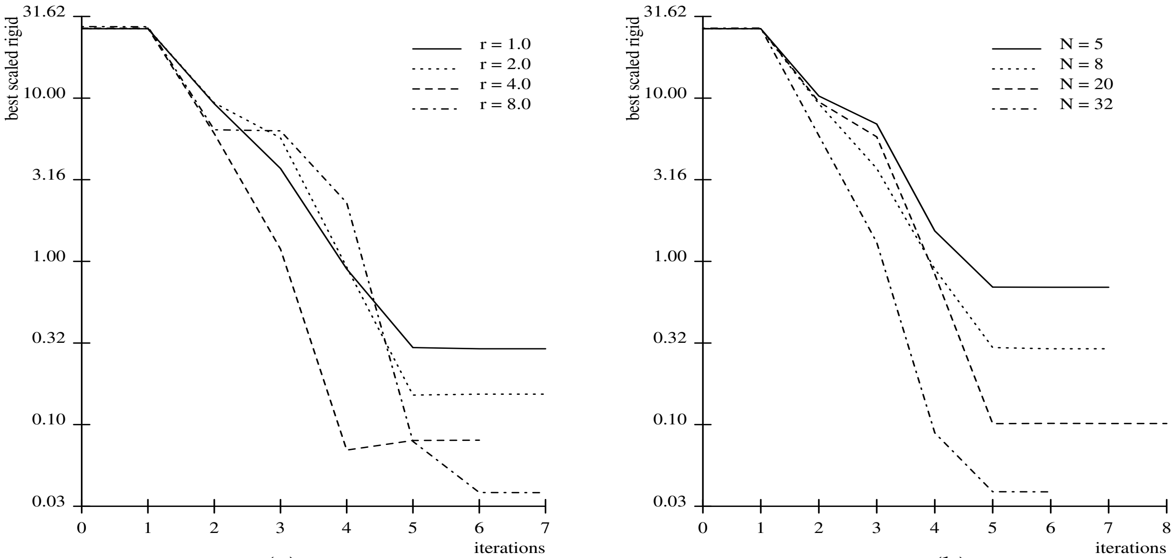
To determine the error sensitivity of our method, we can vary the amount of noise added to the image plane point positions (tracks). Figure 9b shows the scaled rigid matching error as a function of iteration number and image plane noise. As expected, the amount of reconstruction error increases with imaging noise.



(a) (b)

Figure 9: Reconstruction error vs. iteration number

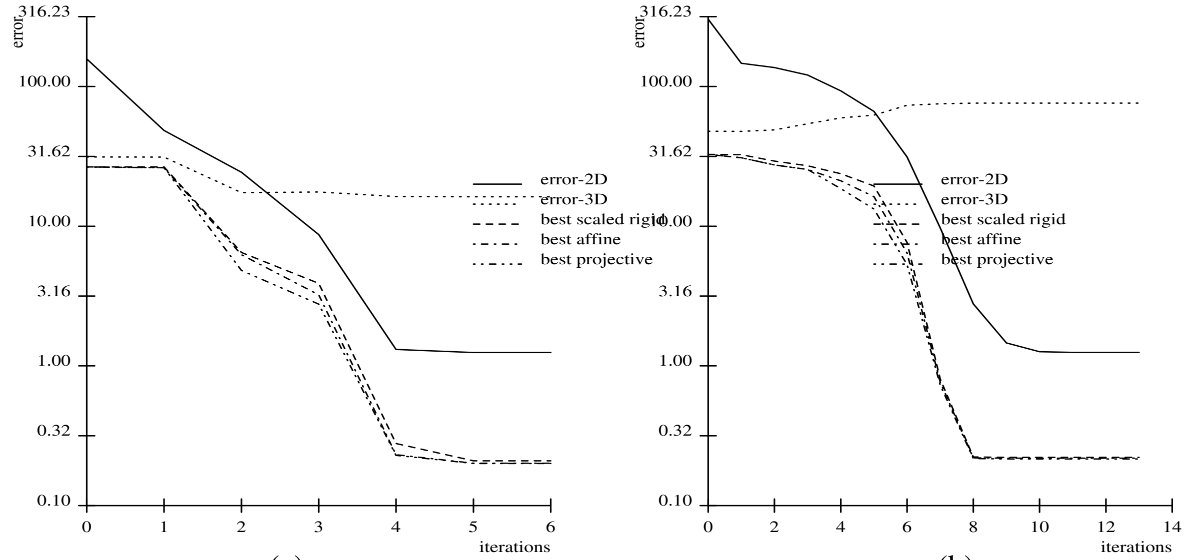
(a) different error measures (b) varying image noise σ (standard deviation in pixels). 96 points, 8 frames, $\Delta\theta = 1°$, 90×90 pixels projected image size.



(a) (b)

Figure 10: Reconstruction error vs. iteration number

(a) increasing rotation angles $\Delta\theta = r = 1°$-$8°$, $N = 8$; (b) increasing number of frames $\Delta\theta = 1°$, $N = 5$-32.



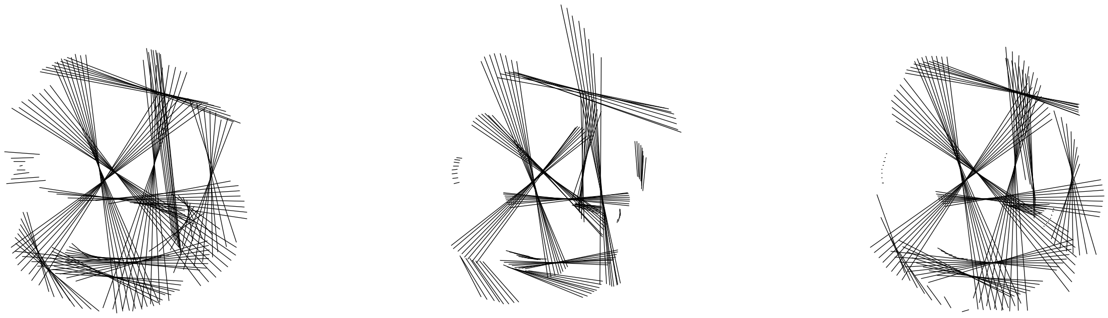
(a) (b)

Figure 11: Reconstruction error vs. iteration number
(a) rotation and translation, (b) pure translation. 96 points, 8 frames, $\Delta\theta = 1°$ and $0°$, $\Delta\mathbf{t} = (0.41, -0.61, 1.22)$, sphere size = 100.

Up to a certain point, we can improve the accuracy of our estimates by using a wider range of viewpoints, either using larger inter-frame displacements (Figure 10a), or using more frames (Figure 10b). However, for a very large range of viewpoints (typically $> 60°$), batch least squares minimization sometimes becomes unstable. This is because the initial estimates of the rotation and translation in the later frames are far from the true values. To alleviate this problem, we start the algorithm with a smaller number of frames and points and incrementally add more frames and points. The motions associated with the additional frames are estimated by linearly extrapolating the estimated motion in the earlier frames. By adopting this technique, the modified least squares technique has so far always converged on both simulated and real data to reasonable values. The incremental frame/point scheme is used for the real image sequences analyzed in the next section.

8.2 General motion

Adding moderate amounts of translation to the object's rotational motion does not affect the quality of the reconstructed solution (Figure 11a, where the translational component of the image plane motion $\approx 2\times$ rotational component). The algorithm also deals well with pure translation (Figure



(a) (b) (c)

Figure 12: Synthetic line tracks

(a) observed tracks; current track estimates (b) after 1 iteration and (c) after 5 iterations. 15 segments, 8 frames, $\Delta\theta = 4°$.

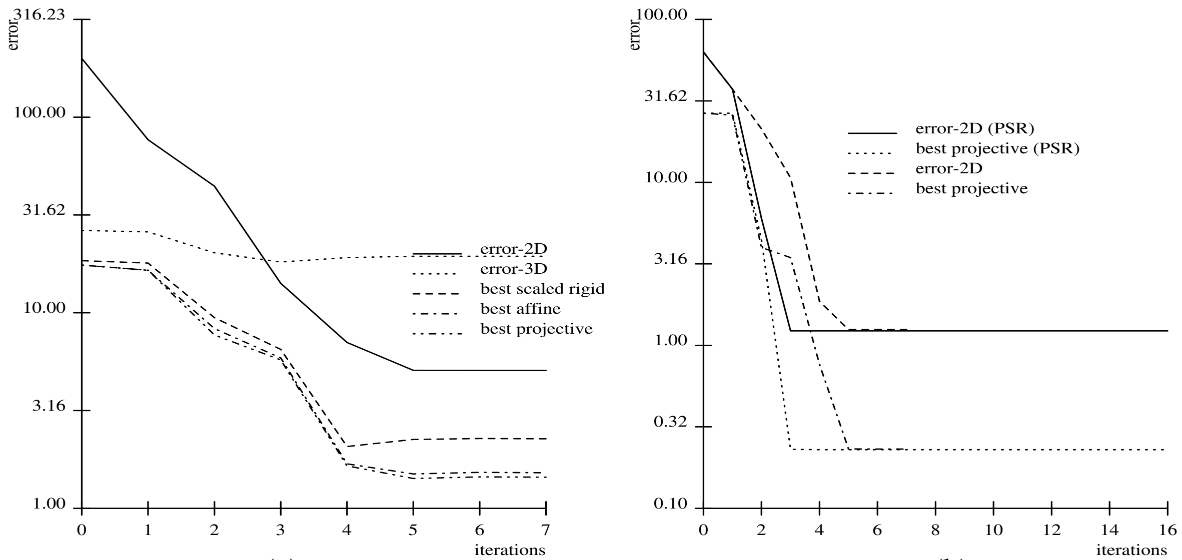
11b), as long as there is sufficient motion parallax.

8.3 Line matching

Our algorithm has also been tested on a collection of line segments. Here, we connect successive pairs in our original point list into lines, and use only the error terms perpendicular to the projected line segments in our least squares formulation (actually, to make convergence faster, we also include the error component along the line, but weighted so that $c_{||} < 0.005c_{\perp}$, where $c_{||}$ and $c_{\perp}$ are the minimum and maximum eigenvalues of $\mathbf{C}_{ij}$ in (14)). The viewed collection of line segments, and the views of the reconstructed segments are shown in Figure 12. As can be seen, only the projections of the line segments onto the measured lines are recovered. To measure the 2D and 3D errors, we modified our algorithm to only compute the error terms perpendicular to the line segments. A plot of the convergence of the algorithm is given in Figure 13a.

8.4 Projective structure recovery

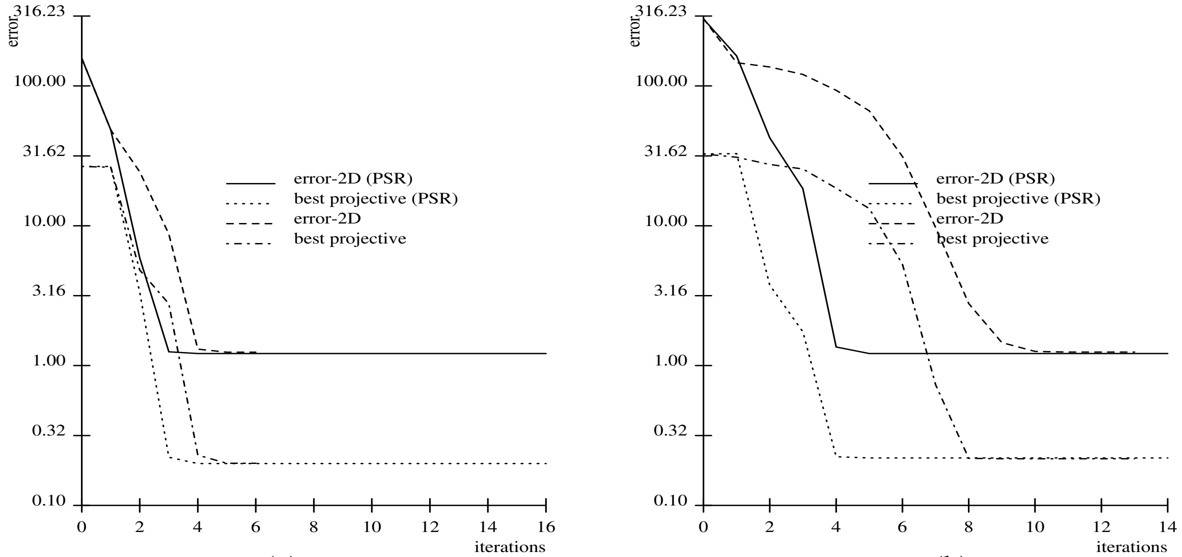
We have performed experiments in recovering projective structure and motion estimates using the formulation presented in Section 3.3. In this case, it only makes sense to compute the image plane (2D) error and the projective structure error, since the scaled rigid and affine errors can be arbitrarily large. Using the same input data as in the previous experiments, we were able to very quickly reduce the 2D errors and recover good projective structure estimates. Figure 13b shows the errors corresponding to the same data as Figure 9a, while Figures 14a and 14b correspond to



(a) (b)

Figure 13: Reconstruction error vs. iteration number

(a) matching line segments, (b) projective structure recovery (PSR). 96 points (48 segments), 8 frames, $\Delta\theta = 1°$. Results from Euclidean structure recovery and projective structure recovery (PSR) are overlayed.



(a) (b)

Figure 14: Reconstruction error vs. iteration number for projective structure recovery (PSR)

(a) rotation and translation, (b) pure translation. 96 points, 8 frames, $\Delta\theta = 1°$ and $0°$, $\Delta\mathbf{t} = (0.41, -0.61, 1.22)$, sphere size = 100. Results from Euclidean structure recovery and projective structure recovery (PSR) are overlayed.

Figures 11a and 11b. From all of these plots, we can conclude that the projective structure recovery is more rapid than with rigid structure estimation. A more detailed presentation of these results can be found in [Szeliski, 1993].

9 Experimental results: real image sequences

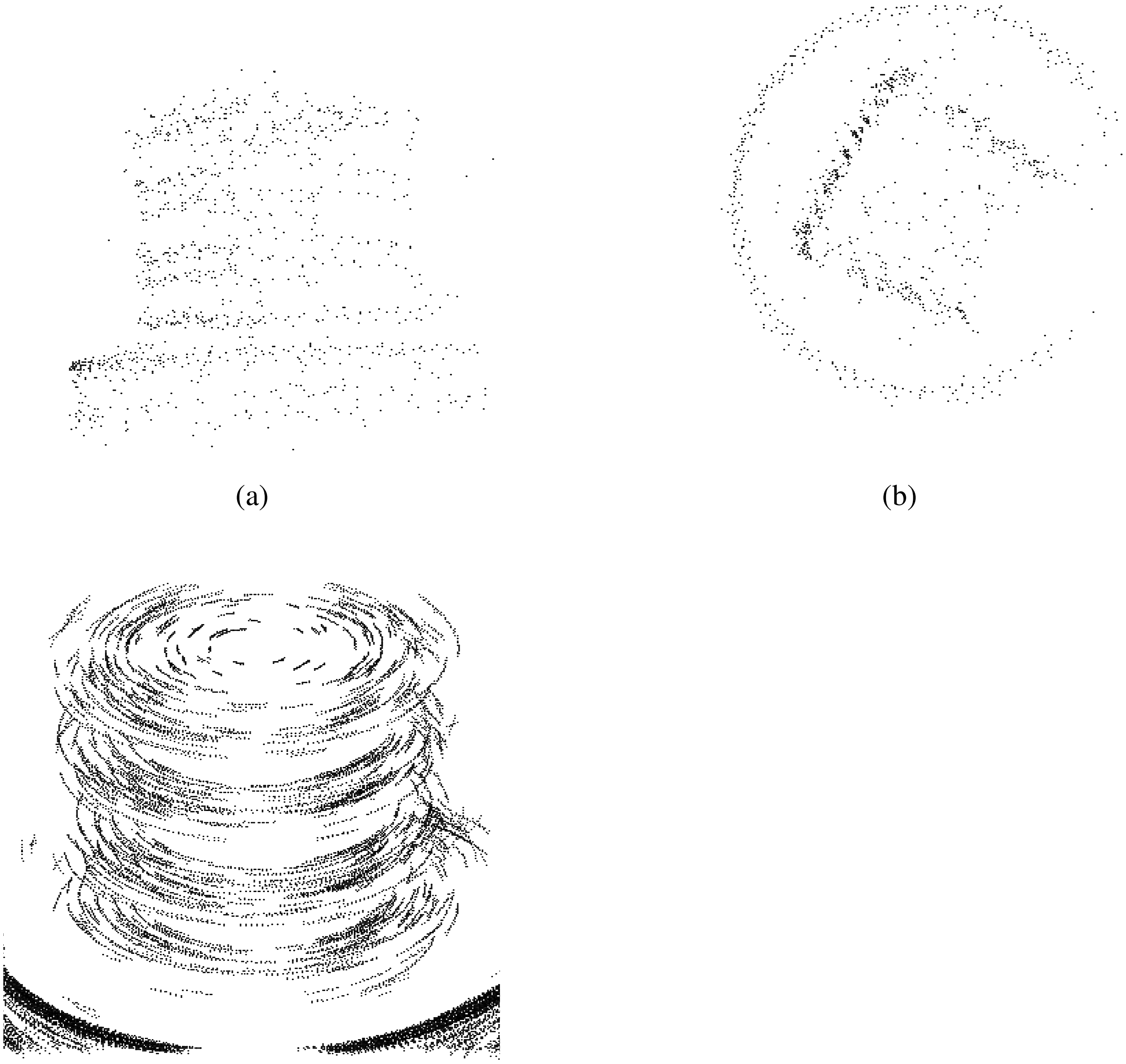
We have applied our algorithm to real image sequences, using the monotonicity-based tracker described in Section 6. Figures 15 and 16 show the results of our shape from motion least squares algorithm applied to these tracks. We can see that the shape of the turntable sides, the cube, and the soda can are all recovered quite well. Two of the cube sides are not quite parallel, which suggests that the projective structure is recovered better than the Euclidean structure.[4] This may be caused by the limited length of the tracks produced by our tracker.

10 Discussion and Conclusions

In this paper, we have demonstrated that shape and motion can be recovered from extended motion sequences by directly applying an iterative non-linear least squares minimization technique, without the need for an initialization stage based on algebraic or linear reconstruction algorithms. Our algorithm is based on a novel re-formulation of the perspective projection equations which encourages the recovery of object-centered shape and motion parameters, and in the limit reduces to the case of orthographic projection. To initialize our algorithm, we project the 2-D points in the middle frame to a constant depth in 3-D. We then simultaneously solve for better structure and motion estimates using the Levenberg-Marquardt algorithm combined with sparse matrix techniques. Our experiments indicate that the algorithm usually converges to its final solution in under a dozen iterations. Because we initialize our algorithm with such a simple (non-informative) estimate of the true shape, the experimental results suggest that the region of convergence for our iterative algorithm is quite broad, and that complicated initialization techniques are not required.

The shape and motion recovery algorithm developed in this paper has several advantages over existing techniques. It can handle perspective (in fact, arbitrary) projection equations, partial

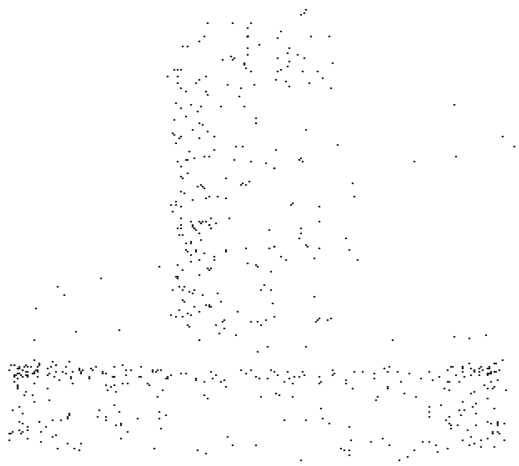
[4] Recall that under artibrary 3-D projective transformations, co-planar points remain co-planar, but parallel planes do not necessarily remain parallel.



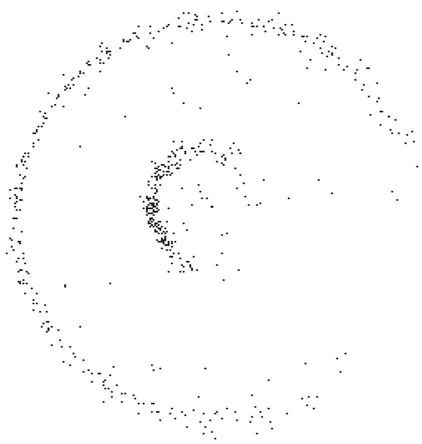
(a) (b)

(c)

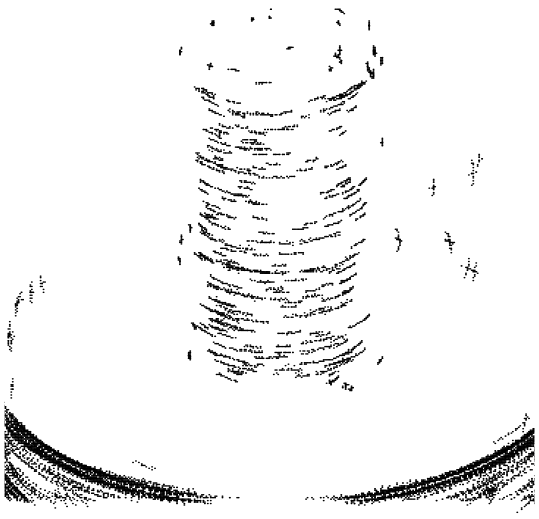
Figure 15: Recovered points and transform for the cube scene using 96 frames
(a) Side view (b) Top view (c) Tracks in 2D image space. The lines in (c) join backprojected estimated points with the actual image points.



(a)



(b)



(c)

Figure 16: Recovered points and transform for the coke can scene using 96 frames
(a) Side view (b) Top view (c) Tracks in 2D image space. The lines in (c) join backprojected estimated points with the actual image points.

and uncertain tracks, and line segment matches in a unified framework. Additional information, such as known calibration points or angular relationships (e.g., co-planarity of certain points, or orthogonality between recovered lines) can easily be added. It makes optimal and robust use of the data, since measurements can be individually weighted and outliers can be rejected. Solving for the unknowns in a batch fashion leads to optimal estimates, while the computational costs are kept reasonable by using sparse matrix techniques. Recovering object-centered shape is more reliable than camera-centered shape, especially for narrow fields of view. Finally, the iterative recovery of shape and motion without a special initialization stage makes this a particularly simple and general technique for shape recovery.

During the development of the algorithm, we did observe occasional occurrences of depth reversals, especially under weak orthography. These are simple to correct, by reflecting the shape about a constant depth plane and checking if the image plane error is reduced. We also observed that solving for the shape and motion parameters simultaneously instead of in alternation (as in [Taylor *et al.*, 1991]) significantly speeds up the convergence. Changing the perspective projection model from a camera-centered projection (5) to a projection about an intermediate frame (6) makes the recovery of camera parameters much quicker. It also speeds up the structure and motion recovery, since the structure description is object-centered rather than camera-centered.

We have begun experiments in recovering projective structure and motion. Our preliminary results indicate that this approach converges much more quickly than Euclidean structure. Many approaches to projective structure recovery [Faugeras, 1992; Demey *et al.*, 1992] use only two images and a small number of points, whereas our approach uses many frames and points and tolerates incomplete correspondences (see also [Mohr *et al.*, 1992]). We also evaluate our structure error by finding the best projective match between the estimated and true structures, whereas previous approaches [Demey *et al.*, 1992] use either invariants or 2D *transfer errors*. In future work, we plan to investigate a recursive formulation which models the correlation between the structure and motion parameters. From the experimental side, we would like to validate our approach on real data using known 3-D ground truth, and apply our techniques to more complicated scenes.

References

[Anandan, 1989] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283–310, January 1989.

[Ayache, 1991] N. Ayache. *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception*. MIT Press, Cambridge, Massachusetts, 1991.

[Bathe and Wilson, 1976] K.-J. Bathe and E. L. Wilson. *Numerical Methods in Finite Element Analysis*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[Bierman, 1977] G. J. Bierman. *Factorization Methods for Discrete Sequential Estimation*. Academic Press, New York, New York, 1977.

[Broida and Chellappa, 1991] T. Broida and R. Chellappa. Estimating the kinematics and structure of a rigid object from a sequence of monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(6):497–513, June 1991.

[Chen and Tsuji, 1992] Q. Chen and S. Tsuji. A hierarchical method that solves the shape and motion from an image sequence problem. In *IEEE/RSJ Int'l Conference on Intelligent Robots and Systems*, pages 2131–2138, July 1992.

[Cui *et al.*, 1990] N. Cui, J. Weng, and P. Cohen. Extended structure and motion analysis from monocular image sequences. In *Third International Conference on Computer Vision (ICCV'90)*, pages 222–229, IEEE Computer Society Press, Osaka, Japan, December 1990.

[Debrunner and Ahuja, 1990] C. H. Debrunner and N. Ahuja. A direct data approximation based motion estimation algorithm. In *10th Int'l Conference on Pattern Recognition*, pages 384–389, 1990.

[Demey *et al.*, 1992] S. Demey, A. Zisserman, and P. Beardsley. Affine and projective structure from motion. In *British Machine Vision Conference (BMVC92)*, pages 49–58, Springer-Verlag, Leeds, England, September 1992.

[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.

[Faugeras *et al.*, 1987] O. D. Faugeras, F. Lustman, and G. Toscani. Motion and structure from motion from point and line matches. In *First International Conference on Computer Vision (ICCV'87)*, pages 25–34, IEEE Computer Society Press, London, England, June 1987.

[Gennery, 1979] D. B. Gennery. Stereo camera calibration. In L. S. Baumann, editor, *Proceedings ARPA IUS Workshop*, pages 101–107, 1979.

[Gennery, 1991] D. B. Gennery. *Camera Calibration Including Lens Distortion*. Technical Report JPL D-8580, National Aeronautics and Space Administration, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 1991.

[Huber, 1981] P. J. Huber. *Robust Statistics*. John Wiley & Sons, New York, New York, 1981.

[Kories and Zimmermann, 1986] R. Kories and G. Zimmermann. A versatile method for the estimation of displacement vector fields from image sequences. In *IEEE Workshop on Motion: Representation and Analysis*, pages 101–106, IEEE Computer Society Press, 1986.

[Kumar *et al.*, 1989] R. V. R. Kumar, A. Tirumalai, and R. C. Jain. A non-linear optimization algorithm for the estimation of structure and motion parameters. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'89)*, pages 136–143, IEEE Computer Society Press, San Diego, California, June 1989.

[Longuet-Higgins, 1981] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[Mohr *et al.*, 1992] R. Mohr, L. Quan, F. Veillon, and B. Boufama. *Relative 3D reconstruction using multiple uncalibrated images*. Technical Report RT 84-IMAG-12, LIFIA — IRIMAG, Grenoble, France, June 1992.

[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.

[Sawhney *et al.*, 1990] H. S. Sawhney, J. Oliensis, and A. R. Hanson. Description and reconstruction from image trajectories of rotational motion. In *Third International Conference on Computer Vision (ICCV'90)*, pages 494–498, IEEE Computer Society Press, Osaka, Japan, December 1990.

[Shabana, 1989] A. A. Shabana. *Dynamics of Multibody Systems*. J. Wiley, New York, 1989.

[Shashua, 1992] A. Shashua. *Projective Structure from two Uncalibrated Images: Structure from Motion and Recognition*. A. I. Memo 1363, Massachusetts Institute of Technology, September 1992.

[Shashua, 1993] A. Shashua. Projective depth: A geometric invariant for 3D reconstruction from two perspective/orthographic views and for visual recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, New York, New York, June 1993.

[Spetsakis and Aloimonos, 1990] M. E. Spetsakis and J. Y. Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–185, June 1990.

[Szeliski, 1990] R. Szeliski. *Real-Time Octree Generation from Rotating Objects*. Technical Report 90/12, Digital Equipment Corporation, Cambridge Research Lab, December 1990.

[Szeliski, 1993] R. Szeliski. *A Least Squares Approach to Affine and Projective Structure and Motion Recovery*. Technical Report, Digital Equipment Corporation, Cambridge Research Lab, (in preparation) 1993.

[Taylor *et al.*, 1991] C. J. Taylor, D. J. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *IEEE Workshop on Visual Motion*, pages 242–248, IEEE Computer Society Press, Princeton, New Jersey, October 1991.

[Tomasi and Kanade, 1990] C. Tomasi and T. Kanade. Shape and motion without depth. In *Third International Conference on Computer Vision (ICCV'90)*, pages 91–95, IEEE Computer Society Press, Osaka, Japan, December 1990.

[Tomasi and Kanade, 1991] C. Tomasi and T. Kanade. Factoring image sequences into shape and motion. In *IEEE Workshop on Visual Motion*, pages 21–28, IEEE Computer Society Press, Princeton, New Jersey, October 1991.

[Tsai, 1987] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, August 1987.

[Tsai and Huang, 1984] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):13–27, January 1984.

[Ullman, 1979] S. Ullman. *The Interpretation of Visual Motion*. MIT Press, Cambridge, Massachusetts, 1979.

[Weng *et al.*, 1989a] J. Weng, N. Ahuja, and T. S Huang. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-11(5):451–476, May 1989.

[Weng *et al.*, 1989b] J. Weng, N. Ahuja, and T. S Huang. Optimal motion and structure information. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'89)*, pages 144–152, IEEE Computer Society Press, San Diego, California, June 1989.

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

A Elemental transform derivatives

The derivatives of the two perspective projection functions (5) and (6) with respect to their 3D arguments and internal parameters are straightforward:

$$\frac{\partial \mathcal{P}_1}{\partial \mathbf{x}^T} = \begin{pmatrix} \frac{f}{z} & 0 & -\frac{fx}{z^2} \\ 0 & \frac{f}{z} & -\frac{fy}{z^2} \end{pmatrix}, \quad \frac{\partial \mathcal{P}_1}{\partial f} = \begin{pmatrix} \frac{x}{z} \\ \frac{y}{z} \end{pmatrix}$$

and

$$\begin{aligned} \frac{\partial \mathcal{P}_2}{\partial \mathbf{x}^T} &= \begin{pmatrix} \frac{s}{1+\eta z} & 0 & -\frac{s\eta x}{(1+\eta z)^2} \\ 0 & \frac{s}{1+\eta z} & -\frac{s\eta y}{(1+\eta z)^2} \end{pmatrix}, \\ \frac{\partial \mathcal{P}_2}{\partial s} &= \begin{pmatrix} \frac{x}{1+\eta z} \\ \frac{y}{1+\eta z} \end{pmatrix}, \quad \frac{\partial \mathcal{P}_2}{\partial \eta} = \begin{pmatrix} -\frac{sxz}{(1+\eta z)^2} \\ -\frac{syz}{(1+\eta z)^2} \end{pmatrix}. \end{aligned}$$

The derivatives of an elemental rigid transformation (2)

$$\mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{t}$$

are

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{R}^T, \quad \frac{\partial \mathbf{y}}{\partial \mathbf{t}} = \mathbf{I}, \quad \text{and} \quad \frac{\partial \mathbf{x}}{\partial \mathbf{q}} = \mathbf{G}(\mathbf{q})\mathbf{C}(\mathbf{x})\mathbf{R}^T,$$

where

$$\mathbf{C}(\mathbf{x}) = \begin{pmatrix} 0 & x_2 & -x_1 \\ -x_2 & 0 & x_0 \\ x_1 & -x_0 & 0 \end{pmatrix} \quad \text{and} \quad \mathbf{G}(\mathbf{q}) = -2 \begin{pmatrix} -q_0 & -q_1 & -q_2 \\ w & -q_2 & q_1 \\ q_2 & w & -q_0 \\ -q_1 & q_0 & w \end{pmatrix}$$

(see [Shabana, 1989]). The derivatives of a screen coordinate with respect to any motion or structure parameter can be computed by applying the chain rule and the above set of equations.

B Matrix layout and LDU decomposition

We arrange the parameter vector $\mathbf{a}$ with all of the structure terms $\mathbf{p}_i$ followed by the time varying motion terms $\mathbf{m}_j$ followed by the global transform (camera) parameters $\mathbf{m}_{\mathrm{g}}$. The Hessian matrix then has the form

$$A = \begin{pmatrix} K & N & O \\ N^T & L & P \\ O^T & P^T & M \end{pmatrix} \tag{15}$$

where

$$K = \sum_j c_{ij} \frac{\partial \mathbf{f}^T(\mathbf{a}_{ij})}{\partial \mathbf{p}_i} \frac{\partial \mathbf{f}(\mathbf{a}_{ij})}{\partial \mathbf{p}_i^T}$$

is the matrix of the point derivative terms,

$$L = \sum_i c_{ij} \frac{\partial \mathbf{f}^T(\mathbf{a}_{ij})}{\partial \mathbf{m}_j} \frac{\partial \mathbf{f}(\mathbf{a}_{ij})}{\partial \mathbf{m}_j^T}$$

is the local (time-varying) transform derivative term matrix,

$$M = \sum_i \sum_j c_{ij} \frac{\partial \mathbf{f}^T(\mathbf{a}_{ij})}{\partial \mathbf{m}_{\mathrm{g}}} \frac{\partial \mathbf{f}(\mathbf{a}_{ij})}{\partial \mathbf{m}_{\mathrm{g}}^T}$$

is the global transform derivative term matrix, and N, O, and P are the cross-derivative term matrices. Note that K and L are block diagonal in our case as we assume both spatial and temporal independence. They would be in general not be block diagonal if spatial or temporal regularization was being used. Matrix N will normally be banded since points do not appear in all frames. The weighted error gradient vector $\mathbf{b}$ has a similar form, with terms involving 3D points followed by terms involving time-varying transforms followed by the global terms.

The matrix $\mathbf{A}$ is stored in skyline form [Bathe and Wilson, 1976], where the skyline for each column in $\mathbf{A}$ is the set of contiguous elements from the diagonal up to the "highest" non-zero element (i.e., the non-zero element with the lowest row index). To solve the set of equations $\mathbf{A}\delta\mathbf{a} = \mathbf{b}$, we use LDU decomposition [Press *et al.*, 1992]. It is here that the skyline form of the matrix shows its real advantage, since its form minimizes the amount of *fill-in* (and hence computation performed) during LDU decomposition and backsubstitution [Bathe and Wilson, 1976]. For example, if the matrix N were zero and there were no global parameters ($|\mathbf{m}_{\mathrm{g}}| = 0$), the system solution becomes equivalent to solving for each structure and motion parameter independently [Taylor *et al.*, 1991]. For non-zero N and $\mathbf{m}_{\mathrm{g}}$, we can still have significant computational savings if the number of frames is significantly less than the number of points (in the converse case, we can interchange the order of $\mathbf{p}_i$ and $\mathbf{m}_j$ in $\mathbf{A}$).