A Paraperspective Factorization Method for Shape and Motion Recovery

Conrad J. Poelman and Takeo Kanade

School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213-3890
(Conrad.Poelman@cs.cmu.edu, tk@cs.cmu.edu)

Abstract. The factorization method, first developed by Tomasi and Kanade, recovers both the shape of an object and its motion from a sequence of images, using many images and tracking many feature points to obtain highly redundant feature position information. The method robustly processes the feature trajectory information using singular value decomposition (SVD), taking advantage of the linear algebraic properties of orthographic projection. However, an orthographic formulation limits the range of motions the method can accommodate. Paraperspective projection, first introduced by Ohta, is a projection model that closely approximates perspective projection by modelling several effects not modelled under orthographic projection, while retaining linear algebraic properties. We have developed a paraperspective factorization method that can be applied to a much wider range of motion scenarios, such as image sequences containing significant translational motion toward the camera or across the image. We present the results of several experiments which illustrate the method's performance in a wide range of situations, including an aerial image sequence of terrain taken from a low-altitude airplane.

1 Introduction

Recovering the geometry of a scene and the motion of the camera from a stream of images is an important task in a variety of applications, including navigation, robotic manipulation, and aerial cartography. While this is possible in principle, traditional methods have failed to produce reliable results in many situations [2].

Tomasi and Kanade [9][10] developed a robust and efficient method for accurately recovering the shape and motion of an object from a sequence of images, called the *factorization method.* It achieves its accuracy and robustness by applying a well-understood numerical computation, the singular value decomposition (SVD), to a large number of images and feature points, and by directly computing shape without computing the depth as an intermediate step. The method was tested on a variety of real and synthetic images, and was shown to perform well even for distant objects.

The Tomasi-Kanade factorization method, however, assumed an orthographic projection model, since it can be described by linear equations. The applicability of the method is therefore limited to image sequences created from certain types of camera motions. The orthographic model contains no notion of the distance from the camera to the object. As a result, shape reconstruction from image sequences containing large translations toward or away from the camera often produces deformed object shapes, as the method tries to explain the size differences in the images by creating size differences in the object. The method also supplies no estimation of translation along the camera's optical axis, which limits its usefulness for certain tasks.

There exist several perspective approximations which capture more of the effects of perspective projection while remaining linear. Scaled orthographic projection, sometimes referred to as "weak perspective" [4], accounts for the scaling effect of an object as it moves towards and away from the camera. Paraperspective projection, first introduced by Ohta [5] and named by Aloimonos [1], models the position effect (an object is viewed from different angles as it translates across the field of view) as well as the scaling effect.

In this paper, we present a factorization method based on the paraperspective projection model. The paraperspective factorization method is still fast, and robust with respect to noise. It can be applied to a wider realm of situations than the original factorization method, such as sequences containing significant depth translation or containing objects close to the camera, and can be used in applications where it is important to recover the distance to the object in each image, such as navigation.

We begin by describing our camera and world reference frames and introduce the mathematical notation that we use. We then present our paraperspective factorization method, followed by the results of several experiments using synthetic data which explore the method's performance. We conclude with the results of two experiments using real image sequences, which demonstrate the practicality of our system.

2 Problem Description

In a shape-from-motion problem, we are given a sequence of F images taken from a camera that is moving relative to an object. We locate P prominent feature points in the first image, and track these points from each image to the next, recording the coordinates (u_{fp},v_{fp}) of each point p in each image f. Each feature point p that we track corresponds to a single world point, located at position $\mathbf{s}_p$ in some fixed world coordinate system. Each image f was taken at some camera orientation, which we describe by the orthonormal unit vectors $\mathbf{i}_f$, $\mathbf{j}_f$, and $\mathbf{k}_f$, where $\mathbf{i}_f$ and $\mathbf{j}_f$ correspond to the x and y axes of the camera's image plane, and $\mathbf{k}_f$ points along the camera's line of sight. We describe the position of the camera in each frame f by the vector $\mathbf{t}_f$ indicating the camera's focal point. This formulation is illustrated in Fig. 1.

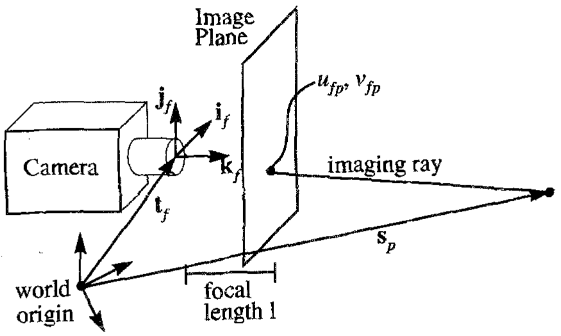


Fig. 1. Coordinate system

The result of the feature tracker is a set of P feature point coordinates (u_{fp},v_{fp}) for each of the F frames of the image sequence. From this information, our goal is to estimate the shape of the object as $\hat{\mathbf{s}}_p$ for each object point, and the motion of the camera as $\hat{\mathbf{i}}_f$, $\hat{\mathbf{j}}_f$, $\hat{\mathbf{k}}_f$ and $\hat{\mathbf{t}}_f$ for each frame in the sequence.

3 The Paraperspective Factorization Method

3.1 Paraperspective Projection

Paraperspective projection was first developed by Ohta [5] in order to solve a shape from texture problem. It closely approximates perspective projection by modelling both the scaling effect (closer objects appear larger than distant ones) and the position effect (objects in the periphery of the image are viewed from a different angle than those near the center of projection [1]), while retaining the linear properties of orthographic projection. The paraperspective projection of an object onto an image, illustrated in Fig. 2, involves two steps.

1. An object point is projected along the direction of the line connecting the focal point of the camera to the object's centroid, onto a hypothetical image plane parallel to the real image plane and passing through the object's centroid.
2. The point is then projected onto the real image plane using perspective projection. Because the hypothetical plane is parallel to the real image plane, this is equivalent to simply scaling the point coordinates by the ratio of the camera focal length and the distance between the two planes.[1]

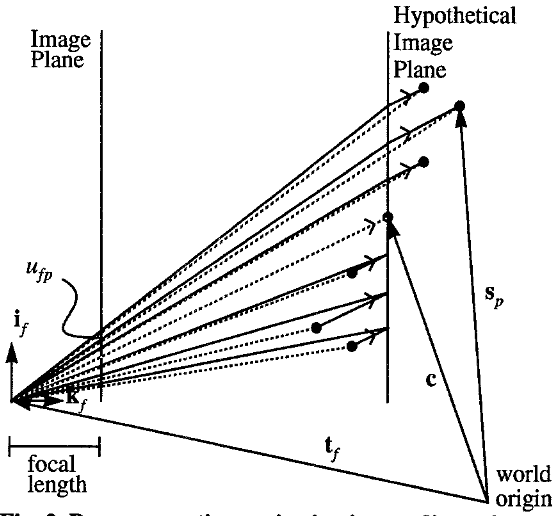


Fig. 2. Paraperspective projection in two dimensions
Dotted lines indicate true perspective projection
>> indicate parallel lines.

In frame f, each object point $\mathbf{s}_p$ is projected along the direction $\mathbf{c} - \mathbf{t}_f$ (which is the direction from the camera's focal point to the object's centroid) onto the plane that passes through the object's centroid $\mathbf{c}$ and has normal $\mathbf{k}_f$. The result of this projection is scaled by the ratio of the camera's focal length l to the depth to the object's centroid, $z_f = (\mathbf{c} - \mathbf{t}_f) \cdot \mathbf{k}_f$. For simplicity, we assume unit focal length, $l = 1$.

Without loss of generality we simplify the mathematics by placing the world origin at the object's centroid $\mathbf{c}$ so that by definition

$$\mathbf{c} = \frac{1}{P}\sum_{p=1}^{P} \mathbf{s}_p = 0. \qquad (1)$$

1. The scaled orthographic projection model (also known as "weak perspective") is similar to paraperspective projection, except that the direction of the initial projection in step 1 is parallel to the camera's optical axis rather than parallel to the line connecting the object's centroid to the camera's focal point. This model captures the scaling effect of perspective projection, but not the position effect. See [6] for details.

The equations for paraperspective projection using this formulation are

$$u_{fp} = \frac{1}{z_f}\left\{\left[\mathbf{i}_f + \frac{\mathbf{i}_f\cdot\mathbf{t}_f}{z_f}\mathbf{k}_f\right]\cdot\mathbf{s}_p - (\mathbf{t}_f\cdot\mathbf{i}_f)\right\} \qquad v_{fp} = \frac{1}{z_f}\left\{\left[\mathbf{j}_f + \frac{\mathbf{j}_f\cdot\mathbf{t}_f}{z_f}\mathbf{k}_f\right]\cdot\mathbf{s}_p - (\mathbf{t}_f\cdot\mathbf{j}_f)\right\} \tag{2}$$

These equations appear much more complicated than the corresponding equations for orthographic projection, which are simply $u_{fp} = \mathbf{i}_f\cdot(\mathbf{s}_p - \mathbf{t}_f)$ and $v_{fp} = \mathbf{j}_f\cdot(\mathbf{s}_p - \mathbf{t}_f)$. However, both can be rewritten in the form

$$u_{fp} = \mathbf{m}_f\cdot\mathbf{s}_p + x_f \qquad v_{fp} = \mathbf{n}_f\cdot\mathbf{s}_p + y_f, \tag{3}$$

although the corresponding definitions of x_f, y_f, $\mathbf{m}_f$, and $\mathbf{n}_f$ differ. In the orthographic case, $\mathbf{m}_f = \mathbf{i}_f$, $\mathbf{n}_f = \mathbf{j}_f$, $x_f = -\mathbf{i}_f\cdot\mathbf{t}_f$, and $y_f = -\mathbf{j}_f\cdot\mathbf{t}_f$. In the paraperspective case, the definitions are

$$z_f = -\mathbf{t}_f\cdot\mathbf{k}_f \tag{4}$$

$$x_f = -\frac{\mathbf{t}_f\cdot\mathbf{i}_f}{z_f} \qquad y_f = -\frac{\mathbf{t}_f\cdot\mathbf{j}_f}{z_f} \tag{5}$$

$$\mathbf{m}_f = \frac{\mathbf{i}_f - x_f\mathbf{k}_f}{z_f} \qquad \mathbf{n}_f = \frac{\mathbf{j}_f - y_f\mathbf{k}_f}{z_f} \tag{6}$$

This similarity in the form of the projection equations enables us to perform the basic decomposition of the matrix in the same manner that Tomasi and Kanade did for the orthographic case, as is described in the following section.

3.2 Paraperspective Decomposition

We can combine (3), for all points p from 1 to P, and all frames f from 1 to F, into the single matrix equation

$$\begin{bmatrix} u_{11} & \dots & u_{1P} \\ \dots & \dots & \dots \\ u_{F1} & \dots & u_{FP} \\ v_{11} & \dots & v_{1P} \\ \dots & \dots & \dots \\ v_{F1} & \dots & v_{FP} \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1 \\ \dots \\ \mathbf{m}_F \\ \mathbf{n}_1 \\ \dots \\ \mathbf{n}_F \end{bmatrix} \begin{bmatrix} \mathbf{s}_1 & \dots & \mathbf{s}_P \end{bmatrix} + \begin{bmatrix} x_1 \\ \dots \\ x_F \\ y_1 \\ \dots \\ y_F \end{bmatrix} \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}, \tag{7}$$

or in short

$$W = MS + T\begin{bmatrix} 1 & \dots & 1 \end{bmatrix}. \tag{8}$$

The $2F \times P$ matrix W, called the *measurement matrix*, collects all of the image measurements (u_{fp}, v_{fp}) such that each column of W contains all the observations for a single point, while each row contains the observed u- or v-coordinates for a single frame. M is the $2F \times 3$ motion matrix whose rows are the $\mathbf{m}_f$ and $\mathbf{n}_f$ vectors, S is the $3 \times P$ shape matrix whose columns are the $\mathbf{s}_p$ vectors, and T is the $2F \times 1$ translation vector.

Using (1) and (3) we can write

$$\begin{aligned} \sum_{p=1}^{P} u_{fp} &= \sum_{p=1}^{P} (\mathbf{m}_f\cdot\mathbf{s}_p + x_f) = \mathbf{m}_f\cdot\sum_{p=1}^{P}\mathbf{s}_p + Px_f = Px_f \\ \sum_{p=1}^{P} v_{fp} &= \sum_{p=1}^{P} (\mathbf{n}_f\cdot\mathbf{s}_p + y_f) = \mathbf{n}_f\cdot\sum_{p=1}^{P}\mathbf{s}_p + Py_f = Py_f \end{aligned} \tag{9}$$

Therefore we can compute x_f and y_f, which are the elements of the translation vector T, immediately from the image data as

$$x_f = \frac{1}{P}\sum_{p=1}^{P} u_{fp} \qquad y_f = \frac{1}{P}\sum_{p=1}^{P} v_{fp}. \tag{10}$$

Once we know the translation vector T, we subtract it from W, giving the *registered measurement matrix*

$$W^* = W - T\begin{bmatrix}1 & \dots & 1\end{bmatrix} = MS. \tag{11}$$

Since W^* is the product of two matrices each of rank at most 3, W^* has rank at most 3, just as it did in the orthographic projection case. When noise is present, the rank of W^* will not be exactly 3, but by computing the SVD of W^* and only retaining the largest 3 singular values, we can factor it into

$$W^* = \hat{M}\hat{S}, \tag{12}$$

where $\hat{M}$ is a $2F \times 3$ matrix and $\hat{S}$ is a $3 \times P$ matrix. Using the SVD to perform this factorization guarantees that the product $\hat{M}\hat{S}$ is the best possible rank 3 approximation to W^*, in the sense that it minimizes the sum of squares difference between corresponding elements of W^* and $\hat{M}\hat{S}$.

3.3 Paraperspective Normalization

The decomposition of W^* into the product of $\hat{M}$ and $\hat{S}$ by (12) is only determined up to a linear transformation. Any non-singular 3×3 matrix A and its inverse could be inserted between $\hat{M}$ and $\hat{S}$, and their product would still equal W^*. Thus the actual motion and shape matrices are given by

$$M = \hat{M}A \qquad S = A^{-1}\hat{S}, \tag{13}$$

with the appropriate 3×3 invertible matrix A selected. The correct A is determined by observing that the rows of the motion matrix M (the $\mathbf{m}_f$ and $\mathbf{n}_f$ vectors) must be of a certain form. Taking advantage of the fact that $\mathbf{i}_f$, $\mathbf{j}_f$, and $\mathbf{k}_f$ are unit vectors, from (6) we observe that

$$|\mathbf{m}_f|^2 = \frac{1 + x_f^2}{z_f^2} \qquad |\mathbf{n}_f|^2 = \frac{1 + y_f^2}{z_f^2}. \tag{14}$$

We know the values of x_f and y_f from our initial registration step, but we do not know the value of the depth z_f. Thus we cannot impose individual constraints on the magnitudes of $\mathbf{m}_f$ and $\mathbf{n}_f$ as was done in the orthographic factorization method where we required that $\mathbf{m}_f$ and $\mathbf{n}_f$ each have unit magnitude. Instead we adopt the following set of constraints on the ratios of the magnitudes of $\mathbf{m}_f$ and $\mathbf{n}_f$.

$$\frac{|\mathbf{m}_f|^2}{1 + x_f^2} = \frac{|\mathbf{n}_f|^2}{1 + y_f^2} \qquad \left(= \frac{1}{z_f^2} \right). \tag{15}$$

There is also a constraint on the angle relationship of $\mathbf{m}_f$ and $\mathbf{n}_f$. From (6), and the knowledge that $\mathbf{i}_f$, $\mathbf{j}_f$, and $\mathbf{k}_f$ are orthogonal unit vectors,

$$\mathbf{m}_f \cdot \mathbf{n}_f = \frac{\mathbf{i}_f - x_f\mathbf{k}_f}{z_f} \cdot \frac{\mathbf{j}_f - y_f\mathbf{k}_f}{z_f} = \frac{x_f y_f}{z_f^2}. \tag{16}$$

The problem with this constraint is that, again, z_f is unknown. We could use either of the two values given in (15) for $1/z_f^2$, but in the presence of noisy input data the two will not be exactly equal, so we use the average of the two quantities. We choose the arithmetic mean over the geometric mean or some other measure in order to keep the solution of these constraints linear. Thus our second set of constraints is

$$\mathbf{m}_f \cdot \mathbf{n}_f = x_f y_f \frac{1}{2}\left(\frac{|\mathbf{m}_f|^2}{1 + x_f^2} + \frac{|\mathbf{n}_f|^2}{1 + y_f^2} \right). \tag{17}$$

This is the paraperspective version of the orthographic constraint that required that the dot product of $\mathbf{m}_f$ and $\mathbf{n}_f$ be zero.

Equations (15) and (17) are homogeneous constraints, which could be trivially satisfied by the solution $\forall f \; \mathbf{m}_f = \mathbf{n}_f = 0$, or $M = 0$. To avoid this solution, we impose the additional constraint

$$|\mathbf{m}_1| = 1. \qquad (18)$$

This does not effect the final solution except by a scaling factor.

Equations (15), (17), and (18) give us $2F+1$ equations, which are the paraperspective version of the *metric constraints*. We compute the 3×3 matrix A such that $M = \hat{M}A$ best satisfies these metric constraints in the least sum-of-squares error sense. This is a simple problem because the constraints are linear in the 6 unique elements of the symmetric 3×3 matrix $Q = A^T A$. Thus we compute Q by solving the overconstrained linear system of $2F+1$ equations in 6 variables defined by the metric constraints, compute its Jacobi Transformation $Q = L\Lambda L^T$ (where Λ is the diagonal eigenvalue matrix), and as long as Q is positive definite, $A = (L\Lambda^{1/2})^T$.

3.4 Paraperspective Motion Recovery

Once the matrix A has been determined, we compute the shape matrix $S = A^{-1}\hat{S}$ and the motion matrix $M = \hat{M}A$. For each frame f, we now need to recover the camera orientation vectors $\hat{\mathbf{i}}_f$, $\hat{\mathbf{j}}_f$, and $\hat{\mathbf{k}}_f$, as well as the depth to the object z_f, from the vectors $\mathbf{m}_f$ and $\mathbf{n}_f$, which are the rows of M. From (6) we see that

$$\hat{\mathbf{i}}_f = z_f \mathbf{m}_f + x_f \hat{\mathbf{k}}_f \qquad \hat{\mathbf{j}}_f = z_f \mathbf{n}_f + y_f \hat{\mathbf{k}}_f. \qquad (19)$$

Since the $\hat{\mathbf{i}}_f$, $\hat{\mathbf{j}}_f$, and $\hat{\mathbf{k}}_f$ produced must be orthonormal, they can be written as functions of only three rotational variables. We can then view the problem as, for each frame f, solving an overconstrained system of 6 equations (the expansion of (19) to each of its vector components) in 4 variables (the three rotational variables and z_f). These small systems of equations can be solved quickly and efficiently using any one of a variety of equation solving techniques. Due to the arbitrary world coordinate orientation, to obtain a unique solution we then rotate the computed shape and motion to align the world axes with the first frame's camera axes, so that $\hat{\mathbf{i}}_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$ and $\hat{\mathbf{j}}_1 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$.

All that remain to be computed are the translations for each frame. We calculate the depth z_f from (15). Once we know x_f, y_f, z_f, $\hat{\mathbf{i}}_f$, $\hat{\mathbf{j}}_f$, and $\hat{\mathbf{k}}_f$, we can calculate $\mathbf{t}_f$ using (4) and (5).

4 Comparison of Methods using Synthetic Data

In this section we compare the performance of our new paraperspective factorization method with the previous orthographic factorization method. The comparison also includes a factorization method based on scaled orthographic projection, which models the scaling effect of perspective projection but not the position effect, in order to demonstrate the importance of modelling the position effect for objects at close range[1]. Our results show that the paraperspective factorization method is a vast improvement over the orthographic method, and underscore the importance of modelling both the scaling and position effects.

4.1 Data Generation

Each synthetic feature point sequence was created by moving a known unit-sized "object" (a set of 60 3D points) through a known motion sequence. The motion con-

1. The scaled orthographic factorization method is equivalent to using the paraperspective factorization method with the focal length of the camera set to infinity. It uses for metric constraints $|\mathbf{m}_f| = |\mathbf{n}_f|$, $\mathbf{m}_f \cdot \mathbf{n}_f = 0$, and $|\mathbf{m}_1| = 1$. See [6] for more details about this method.

sisted of 60 image frames of an object rotating through a total of 30 degrees each of roll, pitch, and yaw. The "object depth" (the distance from the camera's focal point to the front of the object) in the first frame was varied from 3 to 60 times the object size. In each sequence, the object translated across the field of view by a distance of one object size horizontally and vertically, and translated away from the camera by half its initial distance from the camera. Each "image" was created by perspectively projecting the 3D points onto the image plane, for each sequence choosing the largest focal length that would keep the object in the field of view throughout the sequence. The coordinates in the image plane were then perturbed by adding gaussian noise, to model tracking imprecision.

4.2 Error Measurement

We ran each of the three factorization methods on each synthetic sequence and measured the rotation error, shape error, X and Y offset error, and Z offset (depth) error. The errors shown are the root-mean-square (RMS) difference between the known shape or motion parameters and the measured values. Since the shape and translation are only recovered up to a scaling factor, we first scaled these results by the factor that minimized the RMS error. The term "offset" refers to the translational component of the motion as measured in the camera's coordinate frame rather than in world coordinates; the X offset is $\hat{t}_f \cdot \hat{i}_f$, the Y offset is $\hat{t}_f \cdot \hat{j}_f$, and the Z offset is $\hat{t}_f \cdot \hat{k}_f$. Note that the orthographic factorization method supplies no estimation of translation along the camera's optical axis, so the Z offset error could not be computed for that method.

4.3 Discussion of Results

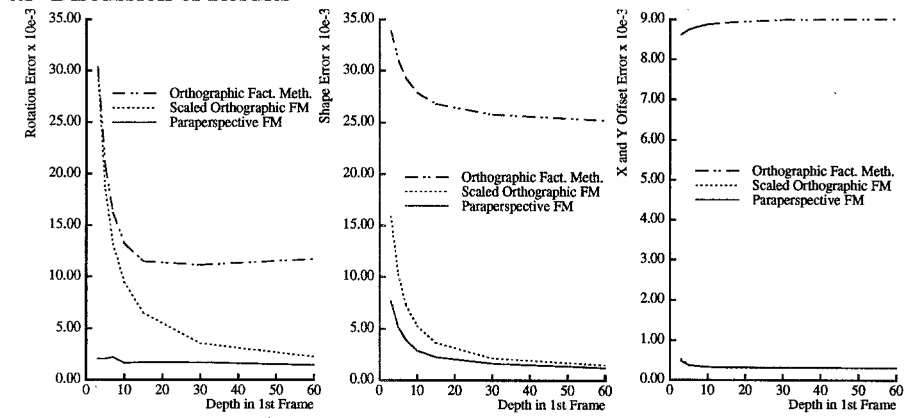


Fig. 3. Methods compared for a typical case
noise standard deviation = 2 pixels

Fig. 3 shows the average errors in the solutions computed by the various methods, as a functions of object depth in the first frame. We see that the paraperspective method performs significantly better than the orthographic factorization method regardless of depth, because orthography cannot model the scaling of the image that occurs due to the motion along the camera's optical axis. The figure also shows that the paraperspective method performs substantially better than the scaled orthographic method at close range, while the errors from the two methods are nearly the same when the object is distant. This confirms the importance of modelling the position effect when objects are near the camera.

In other experiments in which the object was centered in the image and there was no translation across the field of view, the paraperspective method and the scaled orthographic method performed equally well, as we would expect since such image sequences contain no position effects. Similarly, we found that when the object remained centered in the image and there was no depth translation, the orthographic factorization method performed very well, and the paraperspective factorization method provided no significant improvement since such sequences contain neither scaling effects nor position effects.

Our C implementation of the paraperspective factorization method required 20-24 seconds to solve a system of 60 frames and 60 points on a Sun 4/65, with most of this time spent computing the SVD of the measurement matrix.

4.4 Analysis of Paraperspective Method using Synthetic Data

Now that we have shown the advantages of the paraperspective factorization method over the previous method, we further analyze the performance of the paraperspective method to determine its behavior at various depths and its robustness with respect to noise. The synthetic sequences used in these experiments were created in the same manner as in the previous section, except that the standard deviation of the noise was varied from 0 to 4.0 pixels.

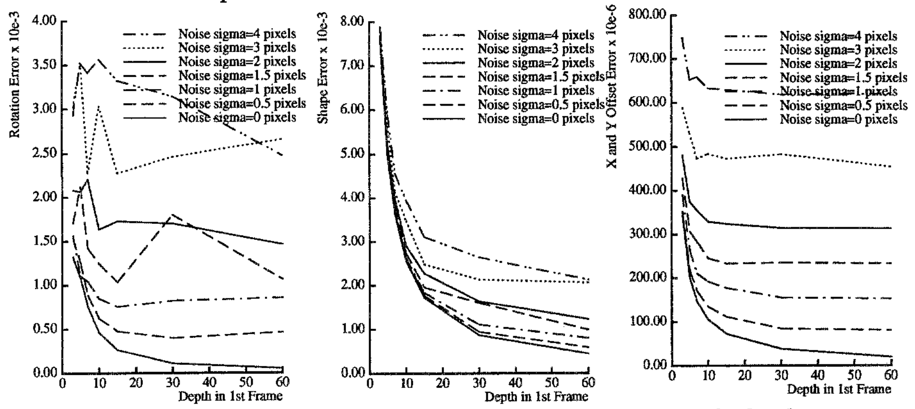


Fig. 4. Paraperspective shape and motion recovery by noise level

In Fig. 4, we see that at high depth values, the error in the solution is roughly proportional to the level of noise in the input, while at low depths the error is inversely related to the depth. This occurs because at low depths, perspective distortion of the object's shape is the primary source of error in the computed results. At higher depths, perspective distortion of the object's shape is negligible, and noise becomes the dominant cause of error in the results. For example, at a noise level of 1 pixel, the rotation and XY-offset errors are nearly invariant to the depth once the object is farther from the camera than 10 times the object size. The shape results, however, appear sensitive to perspective distortion even at depths of 30 or 60 times the object size. We also found that the error in the recovered depth to the object in each frame (Z offset error, not shown), was nearly proportional to both the noise level and the initial depth.

5 Shape and Motion Recovery from Real Image Sequences

We tested the paraperspective factorization method on two real image sequences - a laboratory experiment in which a small model building was imaged, and an aerial sequence taken from a low-altitude plane using a hand-held video camera. Both sequences contain significant perspective effects, due to translations along the optical axis and across the field of view. We implemented a system to automatically identify and track features, based on [10] and [3]. This tracker computes the position of a square feature window which minimizes the sum of the squares of the intensity difference over the feature window from one image to the next.

5.1 Hotel Model Sequence

A hotel model was imaged by a camera mounted on a computer-controlled movable platform. The camera motion included substantial translation away from the camera and across the field of view (see Fig. 5). The feature tracker automatically identified and tracked 197 points throughout the sequence of 181 images.

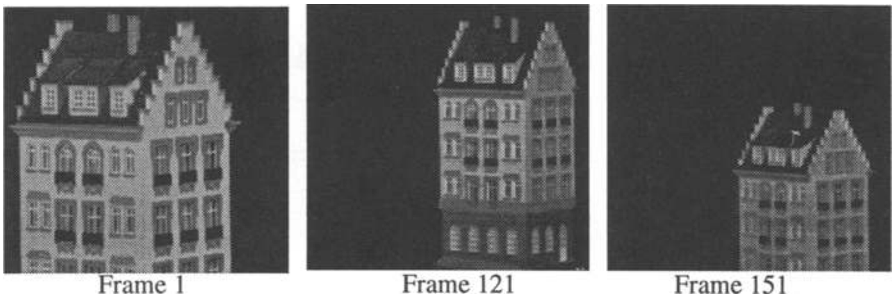


Fig. 5. Hotel Model Image Sequence

We analyzed this sequence using both the paraperspective factorization method and the orthographic factorization method. The shape recovered by the orthographic factorization method was rather deformed (see Fig. 6) and the recovered motion incorrect, because the method could not account for the scaling and position effects which are prominent in the sequence. The paraperspective factorization method, however, models these effects of perspective projection, and therefore produced an accurate shape and accurate motion.

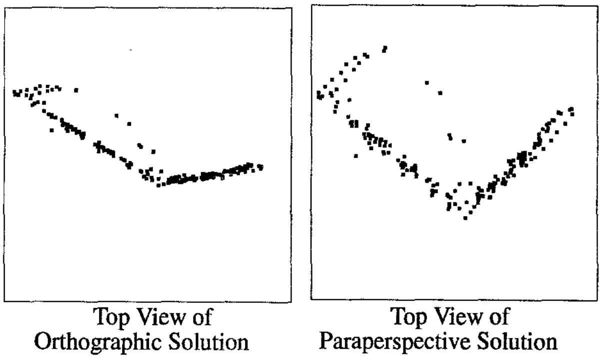


Fig. 6. Comparison of Orthographic and Paraperspective Shape Results

Several features in the sequence were poorly tracked, and as a result their recovered 3D positions were incorrect. While they did not disrupt the overall solution greatly, we found that we could achieve improved results by automatically removing these features in the following manner. Using the recovered shape and motion, we computed the reconstructed measurement matrix W^{recon}, and then eliminated from W those features for which the average error between the elements of W and W^{recon} was more than twice the average such error. We then ran the shape and motion recovery again, using only the remaining 179 features. Eliminating the poorly tracked features decreased errors in the recovered rotation about the camera's x-axis in each frame by an average of 0.5 degrees, while the errors in the other rotation parameters were also slightly improved. The final rotation values are shown in Fig. 7, along with the values we measured using the camera platform. The computed rotation about the camera x-axis, y-axis, and z-axis was always within 0.29 degrees, 1.78 degrees, and 0.45 degrees of the measured rotation, respectively.

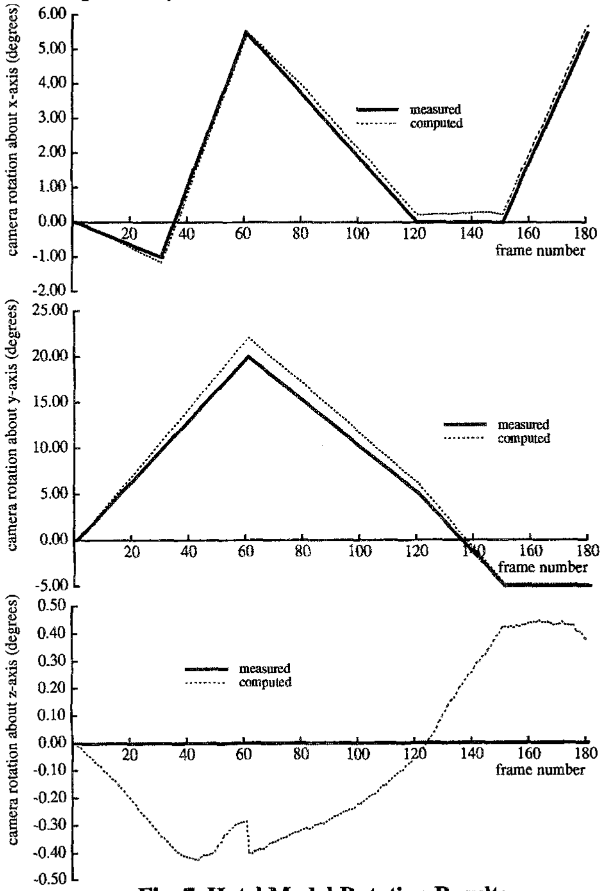
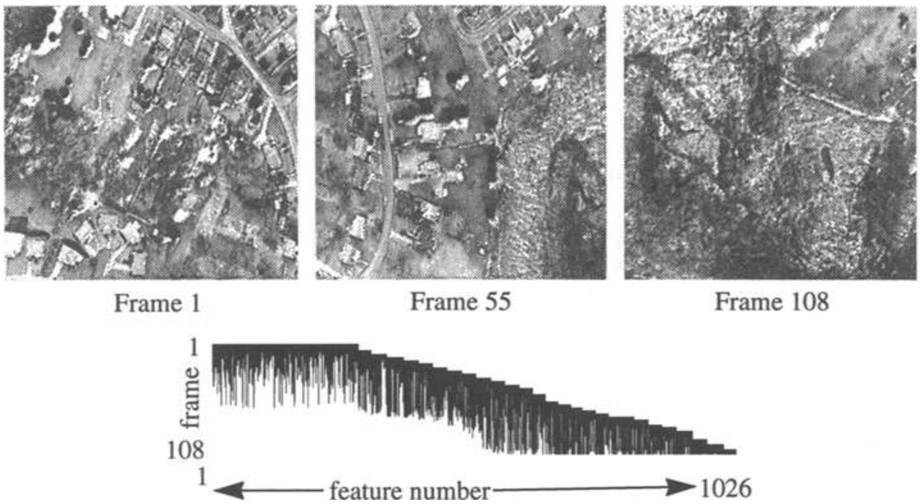


Fig. 7. Hotel Model Rotation Results

5.2 Aerial Image Sequence

An aerial image sequence was taken from a small airplane overflying a suburban Pittsburgh residential area adjacent to a steep, snowy valley, using a small hand-held video camera. The plane altered its altitude during the sequence and also varied its roll, pitch, and yaw slightly. Several images from the sequence are shown in Fig. 8.

Due to the bumpy motion of the plane and the instability of the hand-held camera, features often moved by as much as 30 pixels from one image to the next. The original feature tracker could not track motions of more than approximately 3 pixels, so we



Fill pattern indicating points visible in each frame

Fig. 8. Aerial Image Sequence

implemented a coarse-to-fine tracker. The tracker first estimated the translation using low resolution images, and then refined that value using the same methods as the initial tracker.

The sequence covered a long sweep of terrain, so none of the features were visible throughout the entire sequence. As some features left the field of view, new features were automatically detected and added to the set of features being tracked. A vertical bar in the fill pattern (shown in Fig. 8) indicates the range of frames through which a feature was successfully tracked. A total of 1026 points were tracked in the 108 image sequence, with each point being visible for an average of 30 frames of the sequence. The data still contained more than sufficient redundancy to recover the shape and motion, but we could not compute the SVD of the measurement matrix since for some pairs (f, p), u_{fp} and v_{fp} were unknown. In order to accommodate the missing observations, we developed a new method for decomposing W into $W = \hat{M}\hat{S} + T$ without using the SVD, described in [6]. This slightly modified factorization method was then used to recover the shape of the terrain and the motion of the airplane. Two views of the reconstructed terrain map are shown in Fig. 9. While no ground-truth was available for the shape or the motion, we observed that the terrain was qualitatively correct, captur-

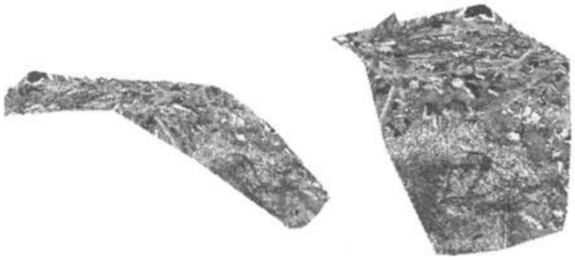


Fig. 9. Two Views of Reconstructed Terrain

ing the flat residential area and the steep hillside as well, and that the recovered positions of features on buildings were elevated from the surrounding terrain.

6 Conclusions

The principle that the measurement matrix has rank 3, as put forth by Tomasi and Kanade in [9], was dependent on the use of an orthographic projection model. We have shown in this paper that this important result also holds for the case of paraperspective projection, which closely approximates perspective projection. We have devised a paraperspective factorization method based on this model, which uses different metric constraints and motion recovery techniques, but retains many of the features of the original factorization method.

In image sequences in which the object being viewed translates significantly toward or away from the camera or across the camera's field of view, the paraperspective factorization method performs significantly better than the orthographic method. The paraperspective method requires that the camera's focal length and center of projection be known in order to accurately model the position effect. In cases in which these are not known, the scaled orthographic factorization method can still be used to model the scaling effect of image projection. Both methods also compute the relative distance from the camera to the object in each image and can accommodate missing or uncertain tracking data, which enables their use in a variety of applications.

7 Acknowledgments

The authors wish to thank Radu Jasinschi for pointing out the existence of the paraperspective projection model and suggesting its applicability to the factorization method. Additional thanks goes to Carlo Tomasi and Toshihiko Morita for their helpful and insightful comments.

8 References

1. John Y. Aloimonos, *Perspective Approximations*, Image and Vision Computing, 8(3):177-192, August 1990.
2. T. Broida, S. Chandrashekhar, and R. Chellappa, *Recursive 3-D Motion Estimation from a Monocular Image Sequence*, IEEE Transactions on Aerospace and Electronic Systems, 26(4):639-656, July 1990.
3. Bruce D. Lucas and Takeo Kanade, *An Iterative Image Registration Technique with an Application to Stereo Vision*, Proceedings of the 7th International Joint Conference on Artificial Intelligence, 1981.
4. Joseph L. Mundy and Andrew Zisserman, *Geometric Invariance in Computer Vision*, The MIT Press, 1992, p. 512.
5. Yu-ichi Ohta, Kiyoshi Maenobu, and Toshiyuki Sakai, *Obtaining Surface Orientation from Texels Under Perspective Projection*, Proceedings of the 7th International Joint Conference on Artificial Intelligence, pp. 746-751, August 1981.
6. Conrad J. Poelman and Takeo Kanade, *A Paraperspective Factorization Method for Shape and Motion Recovery*, Technical Report CMU-CS-93-219, Carnegie Mellon University, Pittsburgh, PA, December 1993.
7. William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, 1988.
8. Camillo Taylor, David Kriegman, and P. Anandan, *Structure and Motion From Multiple Images: A Least Squares Approach*, IEEE Workshop on Visual Motion, pp. 242-248, October 1991.
9. Carlo Tomasi and Takeo Kanade, *Shape and Motion from Image Streams: a Factorization Method - 2. Point Features in 3D Motion*, Technical Report CMU-CS-91-105, Carnegie Mellon University, Pittsburgh, PA, January 1991.
10. Carlo Tomasi and Takeo Kanade, *Shape and Motion from Image Streams: a Factorization Method*, Technical Report CMU-CS-91-172, Carnegie Mellon University, Pittsburgh, PA, September 1991.
11. Roger Tsai and Thomas 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.