Reconstruction of a Scene with Multiple Linearly Moving Objects

Mei Han Takeo Kanade
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213
meihan, tk@cs.cmu.edu

Abstract

We describe an algorithm for reconstructing a scene containing multiple moving objects. Given a monocular image sequence, we recover the scene structure, the trajectories of the moving objects and the camera motion simultaneously. The number of the moving objects is automatically detected without prior motion segmentation. Assuming that the objects are moving linearly with constant speeds, we propose a unified geometrical representation of the static scene and the moving objects. This representation enables the embedding of the motion constraints into the scene structure, which leads to a factorization-based algorithm. Experimental results on synthetic and real images are presented.

1. Introduction

We are interested in video sequences of scenes rich with moving objects taken from a moving airborne platform. Many interesting problems have been discussed on such sequences including scene reconstruction [1, 8], motion segmentation [11, 9], reconstruction of moving trajectories [2] and camera motion recovery [7, 4]. Most of these methods deal with the above problems separately. However, the information integrated over the sequence provides constraints on the scene structure, the trajectories of the moving objects and the camera motion. We are therefore motivated to seek a one step reconstruction algorithm.

In aerial video sequences, the moving objects are often far from the camera. It is therefore difficult to get multiple feature points from one moving object. It is a good approximation to abstract the moving objects as points. As pointed out in [2], recovering the locations of the moving point from a monocular image sequence is impossible without assumptions about its trajectory. We assume that the objects are moving linearly with constant speeds. This assumption is reasonable for most moving objects, such as cars, planes and people, especially for short time intervals.

We propose a unified representation of the static scene and the moving objects in which each point has an initial position and a constant velocity. Points on the static scene are defined to have zero velocity. This representation embeds the motion constraints within the scene structure, and naturally leads to a factorization-based algorithm which reconstructs the scene structure, the trajectories of the moving objects and the camera motion simultaneously. The number of the moving objects is automatically detected without prior motion segmentation. We also discuss solutions to the degenerate cases. Experiments on synthetic and real images are presented.

1.1. Related work

Zelnik-Manor and Irani [12, 5] propose using subspace constraints on multi-frame information to compute homography and optical flows. Their work demonstrates that the use of geometric constraints provides a good way to integrate information over image sequences. The multibody factorization method proposed by Costeira and Kanade [4] reconstructs the motions and shapes of independently moving objects, but requires that each object have multiple feature points. Avidan and Shashua [2] recover the linear trajectory of a 3D point by line fitting. They assume that the object is moving along a line, but they do not require the object to move with constant speed. They assume the camera motion is given as well as the prior motion segmentation, and do not recover the scene structure.

2. Representation

We propose a unified representation of the static scene and the moving objects. Assuming that m feature points are tracked over n images, some of them static and the others moving linearly with constant speeds, we regard every point as a moving point with constant velocity: the static points simply have zero velocity. Any point $\mathbf{p}_j$ is represented by,

$$\mathbf{p}_j = \mathbf{s}_j + i\mathbf{v}_j \tag{1}$$

in a world coordinate system, where $i = 1 \cdots n$ and $j = 1 \cdots m$. n is the number of frames and m is the number of feature points. $\mathbf{s}_j$ is the point position at frame 0 (i.e., when the 0th frame is taken) and $\mathbf{v}_j$ is its motion velocity.

In this paper we use the orthographic camera model. It is straightforward to extend the derivation to weak and para perspective projections which are used in our experiments. If a point $\mathbf{p}_j$ is observed in frame i at image coordinates (u_{ij}, v_{ij}), then,

$$\begin{aligned} u_{ij} &= \mathbf{i}_i \cdot \mathbf{p}_j + t_{xi} \\ v_{ij} &= \mathbf{j}_i \cdot \mathbf{p}_j + t_{yi} \end{aligned} \tag{2}$$

$\mathbf{i}_i$ and $\mathbf{j}_i$ are the rotation axes of the ith camera. $\mathbf{t}_{xi}$ and $\mathbf{t}_{yi}$ are the translations. Therefore,

$$\begin{aligned} u_{ij} &= \mathbf{i}_i \cdot \mathbf{s}_j + i\, \mathbf{i}_i \cdot \mathbf{v}_j + t_{xi} \\ v_{ij} &= \mathbf{j}_i \cdot \mathbf{s}_j + i\, \mathbf{j}_i \cdot \mathbf{v}_j + t_{yi} \end{aligned} \tag{3}$$

We put all the feature points coordinates (u_{ij}, v_{ij}) in a $2n \times m$ matrix W called the *measurement matrix*:

$$W = \begin{bmatrix} u_{11} & \cdots & u_{1m} \\ \cdots & \cdots & \cdots \\ u_{n1} & \cdots & u_{nm} \\ v_{11} & \cdots & v_{1m} \\ \cdots & \cdots & \cdots \\ v_{n1} & \cdots & v_{nm} \end{bmatrix} \tag{4}$$

Each column of W contains the observations for a single point, and each row contains the observed u-coordinates or v-coordinates for a single frame. We have,

$$W = MS + T\,[1\ 1\ \cdots\ 1] \tag{5}$$

with the rotation matrix:

$$M = \begin{bmatrix} \mathbf{m}_1 & \mathbf{m}_2 & \cdots & \mathbf{m}_n & \mathbf{n}_1 & \mathbf{n}_2 & \cdots & \mathbf{n}_n \\ \mathbf{m}'_1 & \mathbf{m}'_2 & \cdots & \mathbf{m}'_n & \mathbf{n}'_1 & \mathbf{n}'_2 & \cdots & \mathbf{n}'_n \end{bmatrix}^{\mathrm{T}} \tag{6}$$

where

$$\begin{aligned} \mathbf{m}_i &= \mathbf{i}_i & \mathbf{m}'_i &= i\,\mathbf{i}_i \\ \mathbf{n}_i &= \mathbf{j}_i & \mathbf{n}'_i &= i\,\mathbf{j}_i \end{aligned} \tag{7}$$

and the shape matrix:

$$S = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_m \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \end{bmatrix} \tag{8}$$

The translation matrix T is:

$$T = [t_{x1}\ t_{x2}\ \cdots\ t_{xn}\ t_{y1}\ t_{y2}\ \cdots\ t_{yn}]^{\mathrm{T}} \tag{9}$$

The constraints of the objects moving linearly with constant speeds enables the unified representation of the motion matrix M, composed of the rotation axes ($\mathbf{m}_i$ and $\mathbf{n}_i$) and the scaled rotation axes ($\mathbf{m}'_i$ and $\mathbf{n}'_i$), and of the shape matrix, composed of the scene structure ($\mathbf{s}_j$) and the motion velocities ($\mathbf{v}_j$).

3. Reconstruction

In this section we describe our factorization-based algorithm. Given tracked feature points, the algorithm decomposes the measurement matrix to recover the scene structure, the trajectories of the moving objects and the camera motion in a single step. The number of the moving objects is automatically detected without prior motion segmentation.

3.1. Moving coordinate system

As the points are either static or moving linearly with constant speeds, the center of gravity of all the points is moving linearly with constant speed as well. The velocity of the center of gravity is equal to the average of all the velocities ($\mathbf{v}_j$). We transform the 3D representations to a moving world coordinate system with fixed orientation (such as being aligned with the first camera) and the origin at the center of gravity of all the points. Therefore,

$$\sum_{j=1}^{m} \mathbf{p}_j = 0 \tag{10}$$

From Equation (2), we have,

$$\begin{aligned} \sum_{j=1}^{m} u_{ij} &= \sum_{j=1}^{m} (\mathbf{i}_i \cdot \mathbf{p}_j + t_{xi}) = \mathbf{i}_i \sum_{j=1}^{m} \mathbf{p}_j + m t_{xi} \\ \sum_{j=1}^{m} v_{ij} &= \sum_{j=1}^{m} (\mathbf{j}_i \cdot \mathbf{p}_j + t_{yi}) = \mathbf{j}_i \sum_{j=1}^{m} \mathbf{p}_j + m t_{yi} \end{aligned} \tag{11}$$

We can compute the translation vector directly from equation (11):

$$t_{xi} = \frac{1}{m} \sum_{j=1}^{m} u_{ij} \quad t_{yi} = \frac{1}{m} \sum_{j=1}^{m} v_{ij} \tag{12}$$

3.2. Decomposition

Once the translation vector T is known, we subtract it from W in Equation (5):

$$\begin{aligned} \hat{W} &= W - T \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} && (13) \\ &= \hat{M}\hat{S} = \hat{M}AA^{-1}\hat{S} = MS && (14) \end{aligned}$$

where $M = \hat{M}A$ and $S = A^{-1}\hat{S}$. According to the representations of M and S in Equations (6) and (8), we know that the rank of the matrix $\hat{W}$ is at most 6 no matter how many moving objects are there. We perform a SVD on $\hat{W}$ and get the best possible rank 6 approximation of $\hat{W}$ as $\hat{M}\hat{S}$, where $\hat{M}$ is a $2n \times 6$ matrix and $\hat{S}$ is a $6 \times m$ matrix. Any non-singular 6×6 matrix A could be inserted between $\hat{M}$ and $\hat{S}$ to get another motion and shape pair.

3.3. Normalization

Metric constraints are imposed to translate the current pair of motion ($\hat{M}$) and shape ($\hat{S}$) to the Euclidean solutions through recovering the linear transformation A. This process is called *normalization*. We recover this 6×6 matrix A by observing that the rows of the motion matrix M consist of the rotation axes and the scaled ones (Equation (6)):

$$|\mathbf{m}_i|^2 = 1 \quad |\mathbf{n}_i|^2 = 1 \quad \mathbf{m}_i \cdot \mathbf{n}_i = 0 \tag{15}$$

$$|\mathbf{m'}_i|^2 = i^2 \quad |\mathbf{n'}_i|^2 = i^2 \quad \mathbf{m'}_i \cdot \mathbf{n'}_i = 0 \tag{16}$$

$$\mathbf{m}_i \cdot \mathbf{n'}_i = 0 \quad \mathbf{m'}_i \cdot \mathbf{n}_i = 0 \tag{17}$$

The above equations impose linear constraints on the elements of MM^{T}. Since

$$MM^{\mathrm{T}} = \hat{M}AA^{\mathrm{T}}\hat{M}^{\mathrm{T}} \tag{18}$$

these constraints are linear on the elements of the symmetric matrix $Q = AA^{\mathrm{T}}$. Define

$$A = \left[\begin{array}{cc} B_1 & B_2 \end{array}\right] \tag{19}$$

where A is 6×6 matrix and B_1, B_2 are both 6×3 matrices. Since $M = \hat{M}A$,

$$\begin{aligned} \hat{M}B_1 &= [\mathbf{m}_1 \cdots \mathbf{m}_n \, \mathbf{n}_1 \cdots \mathbf{n}_n]^{\mathrm{T}} \\ \hat{M}B_2 &= [\mathbf{m'}_1 \cdots \mathbf{m'}_n \, \mathbf{n'}_1 \cdots \mathbf{n'}_n]^{\mathrm{T}} \\ &= N[\mathbf{m}_1 \cdots \mathbf{m}_n \, \mathbf{n}_1 \cdots \mathbf{n}_n]^{\mathrm{T}} \end{aligned} \tag{20}$$

where

$$N = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 2 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & n & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & n \end{bmatrix} \tag{21}$$

according to Equation (7). Therefore,

$$\hat{M}B_2 = N\hat{M}B_1 \tag{22}$$

B_2 is over constrained given B_1 and $\hat{M}$:

$$B_2 = KB_1 \tag{23}$$

where

$$K = \hat{M}^{-1}N\hat{M} \tag{24}$$

and $\hat{M}^{-1}$ is the generalized inverse matrix which is $6 \times 2n$ and uniquely defined when $n \geq 3$.

From Equation (20), we see that Equation (15) imposes constraints on the 21 unknown elements of the 6×6 symmetric matrix $Q_1 = B_1 B_1^{\mathrm{T}}$, while Equation (16) imposes constraints on the 21 unknown elements of $Q_2 = B_2 B_2^{\mathrm{T}}$. From Equation (23) we have,

$$Q_2 = B_2 B_2^{\mathrm{T}} = KB_1 B_1^{\mathrm{T}} K^{\mathrm{T}} = KQ_1 K^{\mathrm{T}} \tag{25}$$

which translates the constraints on Q_2 to constraints on Q_1. Equation (17) imposes constraints on $Q_3 = B_2 B_1^{\mathrm{T}}$ which can also be translated into constraints on Q_1:

$$Q_3 = B_2 B_1^{\mathrm{T}} = KB_1 B_1^{\mathrm{T}} = KQ_1 \tag{26}$$

Therefore, each frame contributes 8 constraints (Equations (15) to (17)) on Q_1. In total, we get $8n$ equations on the 21 unknown elements of the symmetric matrix Q_1. Linear least squares solutions are computed. We then compute the matrix B_1 from Q_1 by matrix decomposition and B_2 by Equation (23), so we recover the linear transformation A.

3.4. Shape and motion reconstruction

Once the matrix A has been found, the shape matrix is computed using $S = A^{-1}\hat{S}$ and the motion matrix is $M = \hat{M}A$. We compute the camera rotation axes as

$$\mathbf{i}_i = \mathbf{m}_i \quad \mathbf{j}_i = \mathbf{n}_i \quad \mathbf{k}_i = \mathbf{m}_i \times \mathbf{n}_i \tag{27}$$

The shape matrix consists of the scene structure and the velocities represented in the moving world coordinate system. We need to transform the representation back to a fixed coordinate system with the origin at the center of gravity of all the points at frame 1.

First the velocity of the moving coordinate system is computed. Since the system is moving at the average velocity of all the moving points, the static points share the same velocity which is the negative value of the average velocity. It is often the case that there are more static points than the points from any moving object, so we let every point vote for a "common" velocity (denoted as $\mathbf{v}_c$). The velocity with the most votes is taken as the negative velocity of the moving coordinate system. The points with the "common" velocity are automatically classified as static and the scene structure is computed as:

$$\mathbf{sc}_j = \mathbf{s}_j + \mathbf{v}_c \tag{28}$$

where $\mathbf{sc}_j$ denotes the scene point position represented in the fixed coordinate system. According to Equation (1), $\mathbf{s}_j$ is the point position at frame 0.

The points which do not have the "common" velocity are the moving points. The number of the moving objects is therefore detected. Their starting positions represented in the fixed coordinate system are:

$$\mathbf{sm}_j = \mathbf{s}_j + \mathbf{v}_c \tag{29}$$

and their velocities are:

$$\mathbf{vm}_j = \mathbf{v}_j - \mathbf{v}_c \tag{30}$$

3.5. Algorithm outline

We summarize the algorithm as follows:

1. Compute the camera translations T from the measurement matrix W according to Equation (12);
2. Subtract T from W to generate $\hat{W}$ according to Equation (13);
3. Perform SVD on $\hat{W}$ and get $\hat{M}$ and $\hat{S}$;
4. Set up linear equations of the 21 unknown elements of the symmetric matrix Q_1 by imposing constraints in Equations (15) to (17);
5. Factorize Q_1 to get B_1 from $Q_1 = B_1 B_1^{\mathrm{T}}$;
6. Compute B_2 from $B_2 = KB_1$;
7. Combine B_1 and B_2 to generate the linear transformation matrix $A = [B_1 \; B_2]$;
8. Recover the shape matrix using $S = A^{-1}\hat{S}$ and motion matrix using $M = \hat{M}A$;
9. Recover the camera rotations as in Equation (27);
10. Reconstruct the scene structure and the trajectories of the moving objects according to Equations (28) to (30).

4. Degenerate cases

The algorithm described in Section 3 solves the full rank case where the static structure and the motion space of the objects are both rank 3. This is the case when the scene is three dimensional and the velocities of the moving objects span a three dimensional space. In this section we discuss the solutions for degenerate cases.

If the scene has a degenerate shape, such as all the points lie in a plane, the plane plus parallax method [6] detects the case and solves for the scene structure (plane position), the camera motion and the motion segmentation [1, 7]. The motion trajectories can be recovered using the method proposed by Avidan and Shashua [2] given the reconstruction of the camera motion. In this section we discuss the solutions to the degenerate motion space of the objects.

We classify the degenerate situations as three classes:

1. Rank-3 case: The matrix $\hat{W}$ has rank 3 when there is no moving object in the scene. The one-object factorization method [10, 8] is used to recover the scene structure and the camera motion.
2. Rank-4 case: The matrix $\hat{W}$ has rank 4 when there is one moving object or multiple objects moving in the same and/or the opposite direction (not necessarily the same 3D line). Section 4.2 describes a linear algorithm for this case.
3. Rank-5 case: The matrix $\hat{W}$ has rank 5 when the velocities of the objects lie in a two dimensional space (not necessarily the same 3D plane). Section 4.3 gives a nonlinear solution to this case.

4.1. Rank approximation

Given tracked feature points, we first need to decide which case (full rank or the three degenerate cases) is the best approximation. The rank of the matrix $\hat{W}$ is one important clue. However, finding the rank of $\hat{W}$ is not straightforward. Both inaccuracies in feature locations and approximation of perspective projection using orthographic (weak perspective or para perspective) camera models induce noises in the rank computation.

We use an algorithm similar to [3, 5] to detect the rank of $\hat{W}$. We first estimate the noise level of the input images and approximate the rank using the singular values of $\hat{W}$ and the noise level. The rank of $\hat{W}$ can only be any value in $\{3, 4, 5, 6\}$, which is determined by the motion space of the objects and is not dependent on the number of moving objects. Compared with Costeira and Kanade's method [4], in which the rank value is used to detect the number of moving objects and is affected by degenerate objects, our rank computation is more reliable.

4.2. Rank-4 case

When only one moving object is in the scene, or when all moving objects travel in the same or the opposite direction, the motion space is one dimensional. We align the $\mathbf{x}$ direction of the world coordinate system with the motion direction. The system is still moving with the constant velocity. Therefore, the motion and shape matrices are (compare with Equations (6) and (8)):

$$\begin{aligned} M &= \begin{bmatrix} \mathbf{m}_1 & \mathbf{m}_2 & \cdots & \mathbf{n}_1 & \mathbf{n}_2 & \cdots & \mathbf{n}_n \\ i_{x1} & 2i_{x2} & \cdots & j_{x1} & 2j_{x2} & \cdots & nj_{xn} \end{bmatrix}^{\mathrm{T}} \\ S &= \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_m \\ v_{x1} & v_{x2} & \cdots & v_{xm} \end{bmatrix} \end{aligned} \tag{31}$$

Similar derivations apply to the computation of T (Equation (12)) and the decomposition of $\hat{W}$ (Equation(14)). In this case the rank of $\hat{W}$ is 4 and the linear transformation matrix A is 4×4. Similarly, we define

$$A = \begin{bmatrix} B_1 & B_2 \end{bmatrix} \tag{32}$$

where B_1 is 4×3, B_2 is 4×1 and we have,

$$B_2 = KB_{11} \tag{33}$$

where B_{11} is the first column of B_1 and K is defined in Equation(24). Since the matrix M consists of the rotation axes and only the $\mathbf{x}$ elements of the scaled rotation axes, the constraints in Equations (16) and (17) cannot be represented as linear constraints on the elements of MM^{T}. The constraints in Equation (15) still hold and provide full rank linear equations on the 10 unknown elements of the symmetric 4×4 matrix Q_1. Least squares solutions are computed. We then compute B_1 by matrix decomposition of Q_1 and B_2 by Equation (33).

We apply a derivation similar to the one in Section 3 to recover the motion and shape. The alignment matrix R of the world coordinate system is determined by:

$$B_2 R = K(B_1 R)_1 \tag{34}$$

which means that the alignment is constrained to make the $\mathbf{x}$ direction of the world coordinate system as the moving direction. $()_1$ denotes the first column of the matrix.

4.3. Rank-5 case

When the velocities of all moving objects lie in a two dimensional space, we assume that the $\mathbf{x}$ direction and $\mathbf{y}$ direction of the world coordinate system are aligned with the two dimensional motion space. The system is still moving with the constant velocity. Therefore, the motion and shape matrices are:

$$\begin{aligned} M &= \begin{bmatrix} \mathbf{m}_1 & \mathbf{m}_2 & \cdots & \mathbf{n}_1 & \mathbf{n}_2 & \cdots & \mathbf{n}_n \\ i_{x1} & 2i_{x2} & \cdots & j_{x1} & 2j_{x2} & \cdots & nj_{xn} \\ i_{y1} & 2i_{y2} & \cdots & j_{y1} & 2j_{y2} & \cdots & nj_{yn} \end{bmatrix}^{\mathrm{T}} \\ S &= \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_m \\ v_{x1} & v_{x2} & \cdots & v_{xm} \\ v_{y1} & v_{y2} & \cdots & v_{ym} \end{bmatrix} \end{aligned} \tag{35}$$

So the rank of $\hat{W}$ is 5 and the linear transformation matrix A is 5×5. Similarly, we define

$$A = \begin{bmatrix} B_1 & B_2 \end{bmatrix} \tag{36}$$

where B_1 is 5×3 and B_2 is 5×2:

$$B_2 = K \begin{bmatrix} B_{11} & B_{12} \end{bmatrix} \tag{37}$$

where B_{11} and B_{12} are the first two columns of B_1 and K is defined in Equation (24). Here only the constraints in Equation (15) can be represented as linear constraints on the elements of Q_1. In this case the constraints are not sufficient to solve for the 15 unknown elements of the symmetric 5×5 matrix Q_1 linearly.

The constraints in Equations (16) and (17) can be represented on the elements of Q_1 and the five elements of the third column of B_1, which is a 5×1 vector denoted by $\mathbf{c}$, such as:

$$|\mathbf{m}'_i|^2 = \hat{\mathbf{m}}_i B_2 B_2^{\mathrm{T}} \hat{\mathbf{m}}_i^{\mathrm{T}} + i^2 \hat{\mathbf{m}}_i \mathbf{c}\mathbf{c}^{\mathrm{T}} \hat{\mathbf{m}}_i^{\mathrm{T}} = i^2 \tag{38}$$

where $\hat{\mathbf{m}}_i$ is the ith row of matrix $\hat{M}$.

Therefore, we get linear equations of the 15 unknown elements of Q_1 and the 15 unknown elements of $\mathbf{c}\mathbf{c}^{\mathrm{T}}$. Since these equations cannot provide full rank constraints on the 30 unknowns, there is no linear solutions of Q_1 and $\mathbf{c}\mathbf{c}^{\mathrm{T}}$ directly. However, the constraints are full rank on the elements of Q_1 if $\mathbf{c}\mathbf{c}^{\mathrm{T}}$ is given. That is, if $\mathbf{c}$ can be computed, we can get a linear solution of Q_1. In this way we change the problem to a small scale nonlinear optimization on the 5 elements of $\mathbf{c}$. Once the vector $\mathbf{c}$ is computed, the matrix Q_1 is computed by least squares solutions. B_1 is then calculated from Q_1 and B_2 by Equation (37).

The alignment matrix R of the world coordinate system is determined by:

$$B_2 R = K \begin{bmatrix} (B_1 R)_1 & (B_1 R)_2 \end{bmatrix} \tag{39}$$

This constrains the alignment of the world coordinate system by putting the $\mathbf{x} - \mathbf{y}$ plane of the system on the motion plane. The above equation is solved by the least eigenvalue method. $()_1$ and $()_2$ denote the first and second column of the matrix respectively.

5. Experiments

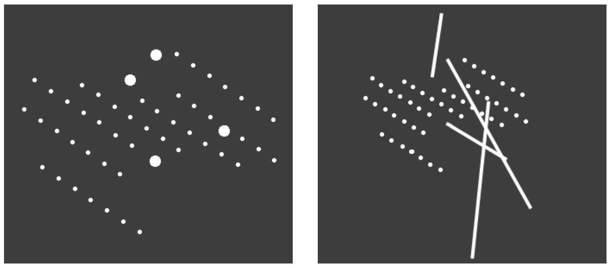
In this section a number of experiments are described. First some synthetic images are used to evaluate the quality of the algorithm. Then two experiments are conducted on real image sequences. The first sequence was taken by a hand-held camera of an indoor scene, and the reconstruction results are compared with the ground truth. The second sequence was taken by a small plane flying over the buildings.

5.1. Synthetic sequences

We generate sequences of 100 frames with 49 feature points from the static scene and 0 to 9 objects moving in random directions. The shape of the static scene is a sweep of the sin curve in the space. We add 2% noises to the feature locations.

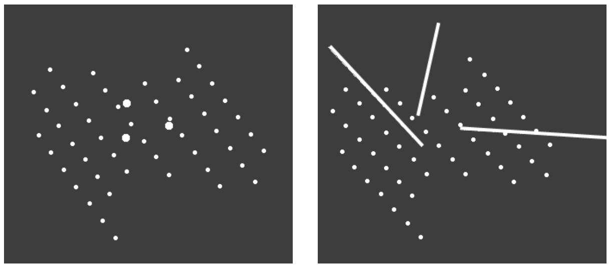
Figure 1 illustrates the case where 4 objects are moving randomly in 3D space. The algorithm automatically detects the number of the moving objects as 4, reconstructs the static scene and the initial positions of the 4 moving objects, as shown in Figure 1(a). Figure 1(b) shows the trajectories of the moving objects as well as the static scene.

We perform experiments on the case that there are two moving objects whose directions are on a plane. The algorithm detects that the rank as 5 and recovers the scene structure and the two motion trajectories correctly. We also try the case that there are three moving objects but their motion directions lie in a two dimensional space. The algorithm gets the right rank approximation (5) and the accurate reconstructions (shown in Figure 2).



(a) (b)

Figure 1. Full rank case: a scene with a three dimensional motion space. (a) The reconstructed scene structure and the initial positions of the moving objects. (b) The reconstructed scene and the motion trajectories.



(a) (b)

Figure 2. Rank-5 case: a scene with three motion trajectories which lie in a two dimensional space. (a) The reconstructed scene structure and the initial positions of the moving objects. (b) The reconstructed scene and the motion trajectories.

We also conduct experiments on rank-4 cases that there is only one moving object, and that there are multiple moving objects which are moving in the same or the opposite direction. The algorithm detects the rank as 4 in both cases. For the case that there is no moving object, the algorithm correctly detects the rank as 3 and recovers the scene structure.

In all cases, we measure the reconstruction error by comparison with the ground truth. Since the reconstruction from monocular image sequences is up to scale, we assume that the size of the static shape is 1. With 2% image noises, the maximum distance between the recovered static points and their known positions is 1.0%, the maximum error of the reconstructed initial positions of the moving objects is 1.2% and the velocity error is less than 1.1%. We also assess the quality of the camera motion reconstruction. The maximum distance between the recovered camera locations and the ground truth values is 1.4% and the maximum angle between the recovered camera orientations and the known values is $0.1°$.

5.2. Real sequence 1

This sequence was taken of an indoor scene by a handheld camera. Three objects, a car, a plane and a toy person, are moving linearly with constant speeds. The car and the person are moving on the floor, and the speed of the car is three times of the speed of the person. Their motion directions are perpendicular with each other. The plane is taking off on a slope and moves two times as fast as the car. The boxes represent the static scene. 24 images were taken. Three of them are shown in Figure 3(a). 23 feature points were manually selected and tracked. We use the first 18 frames to perform the reconstruction. The shapes of the boxes, the starting positions of the moving objects and the motion velocities are recovered and demonstrated in Figure 3(b) (with texture mapping) and (c) (with wireframe), the motion trajectories are overlaid in the images. Figure 3 (d) show the recovered camera locations and orientations.

We assess the quality of the reconstruction by comparison with the ground truth. The angle between the motion direction of the car and that of the person is $90.15°$, the ratio between the speeds is 3.05 which is close to the expected value 3.0. The ratio of the speed of the plane to that of the car is 1.97. The maximum distance between the positions of the recovered static points and the ground truth positions is 2mm. The recovered motion direction of the plane is $20°$ tilted upward from the floor, which is close to the expected value.

We project the motion trajectories back to the images and measure the discrepancies of the tracked objects and the back projections in the last seven frames. The maximum discrepancy is 2 pixels.

5.3. Real sequence 2

This sequence was taken by a small airplane flying over a scene with multiple moving cars. The first 80 frames of a 90 frame sequence are used, three of these frames are shown in Figure 4(a). 35 feature points were manually selected in the first frame corresponding to the buildings and the two moving cars. These points were automatically tracked in the remaining frames. The algorithm estimates the rank of $\hat{W}$ as 4 because the two cars are moving in opposite directions. Figures 4(b) and (c) show the recovered buildings as well as the motion trajectories. Since the resolution of the input images is very low, the texture mapping is not very clear. Similar to the experiment in Section 5.2, we measure the discrepancies of the back projection cars and the tracked cars for the final 10 frames. The maximum discrepancies are 4 pixels for the white car and 5 pixels for the black car.

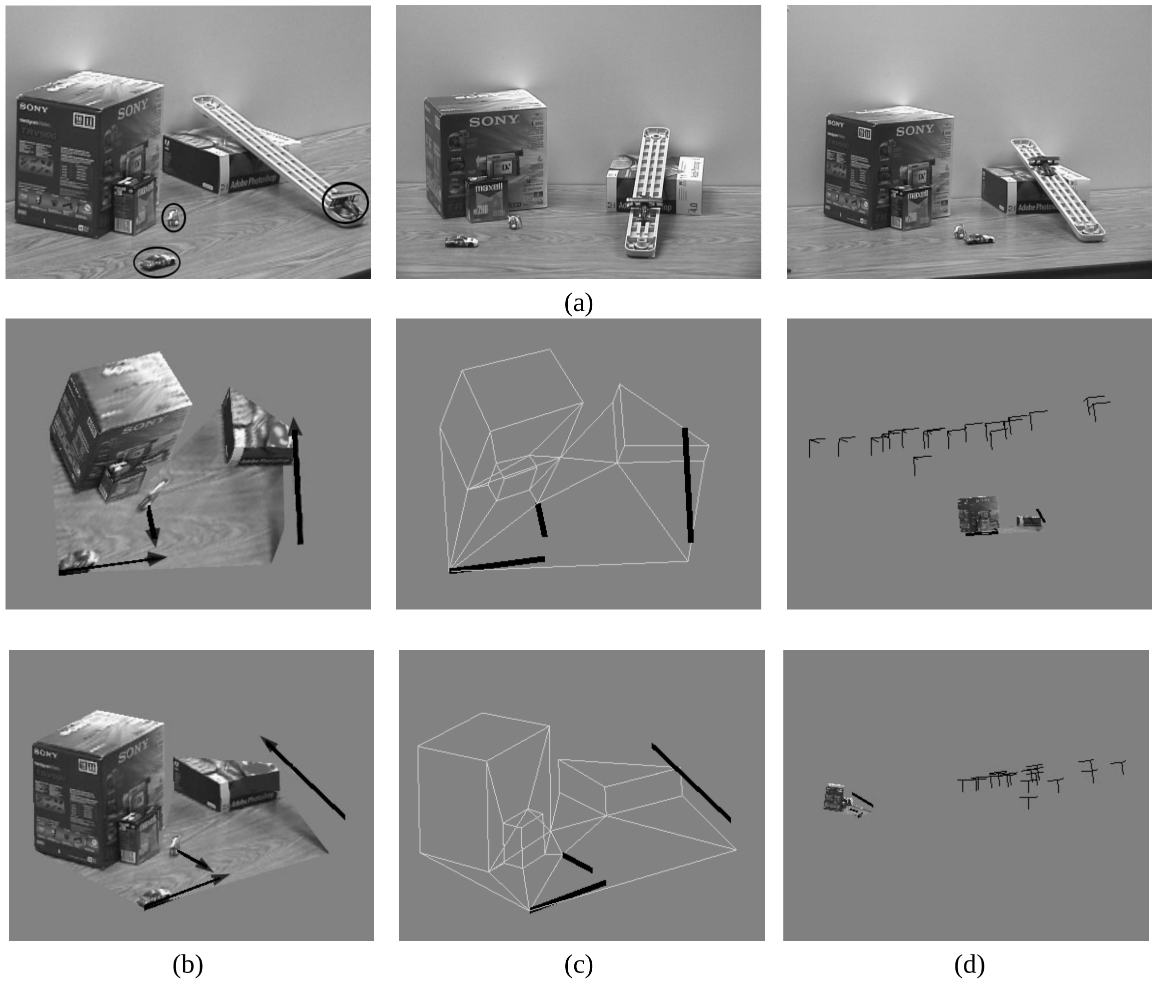


Figure 3. (a) 1st, 7th and 18th images of the indoor sequence, the moving objects are circled in the 1st image. (b) Two views of the reconstruction with texture mapping, the black lines denote the recovered motion trajectories, the arrows show the motion directions. (c) Two views of the reconstruction with wireframe, the black lines denote the recovered motion trajectories. (d) Two views of the reconstruction, the 3-axis figures are the recovered cameras.

6. Discussion

Assuming that the objects are moving linearly with constant speeds, we propose a unified geometrical representation incorporating the static scene and the moving objects. This representation enables the embedding of the motion constraints into the scene structure, that is, the shape matrix is composed of two spaces: one is the scene structure space and another is the motion space. The algorithm makes use of the constraints between the camera motion and the shape matrix to perform the reconstruction. Experiments show that the reconstruction is reliable in the presence of noises. However, analysis is necessary to the sensitivity to noises of the two spaces (the scene space and the motion space) because every point, either static or moving, contributes to the scene space and only the moving points contribute to the motion space.

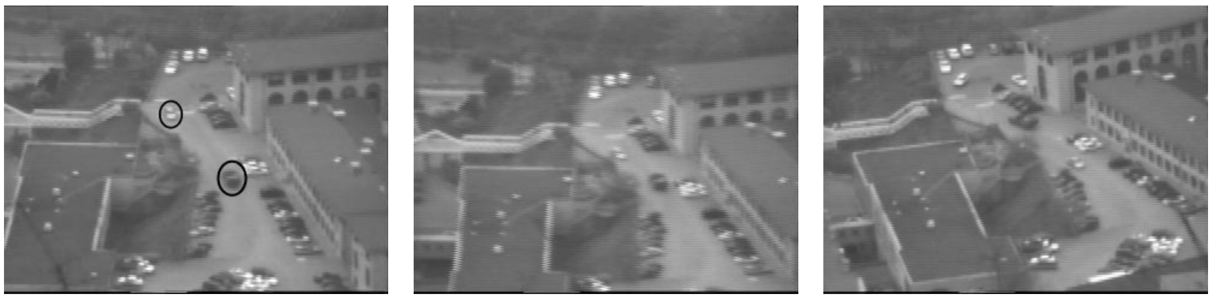
Acknowledgments

We would like to thank Simon Baker for his insightful suggestions on this work and thoughtful comments on this paper, Jianbo Shi, Martial Hebert, David Larose and Teck Khim Ng for fruitful discussions.

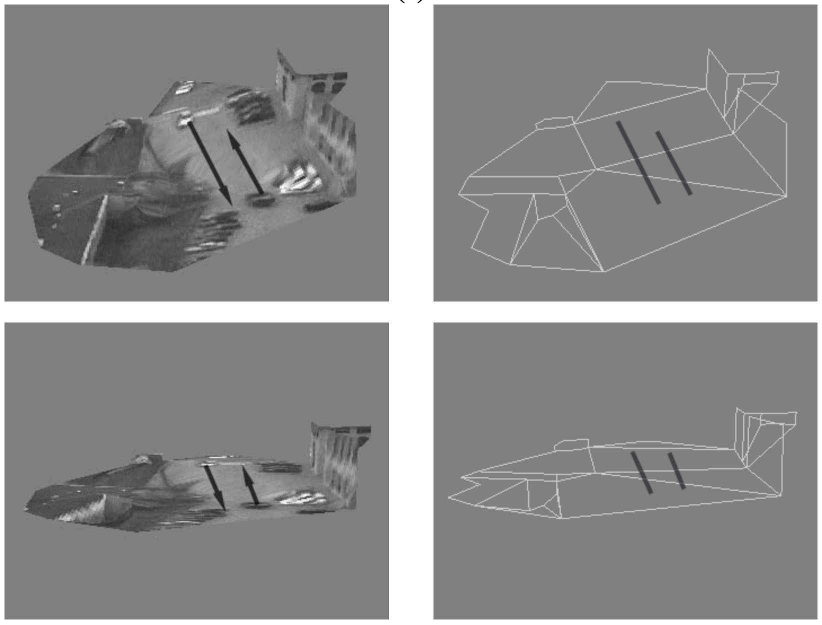
References

[1] P. Anandan, K. Hanna, and R. Kumar. Shape recovery from multiple views: A parallax based approach. In *ARPA94*, pages II:947–955, 1994.

[2] S. Avidan and A. Shashua. Trajectory triangulation of lines: Reconstruction of a 3d point moving along a line from a



(a)



(b) (c)

Figure 4. (a) 1st, 33th and 80th images from the outdoor sequence, the moving objects are circled in the 1st image. (b) Two views of the reconstruction with texture mapping, the black lines denote the recovered motion trajectories, the arrows show the motion directions. (c) Two views of the reconstruction with wireframe, the black lines denote the recovered motion trajectories.

monocular image sequence. In *CVPR99*, pages II:62–66, 1999.

[3] T. Boult and L. G. Brown. Factorization-based segmentation of motions. In *Proceedings of the 1991 Visual Motion Workshop*, pages 179–186, 1991.

[4] J. Costeira and T. Kanade. A multibody factorization method for independently moving-objects. *IJCV*, 29(3):159–179, 1998.

[5] M. Irani. Multi-frame optical flow estimation using subspace constraints. In *ICCV99*, pages 626–633, 1999.

[6] M. Irani, P. Anandan, and D. Weinshall. From reference frames to reference planes: Multi-view parallax geometry and applications. In *ECCV98*, 1998.

[7] M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using region alignment. *PAMI*, 19(3):268–272, March 1997.

[8] C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *PAMI*, 19(3):206–218, 1997.

[9] H. Sawhney, Y. Guo, J. Asmuth, and R. Kumar. Independent motion detection in 3d scenes. In *ICCV99*, pages 612–619, 1999.

[10] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137–154, 1992.

[11] P. Torr and D. Murray. Outlier detection and motion segmentation. *SPIE*, 2059:432–443, 1993.

[12] L. Zelnik-Manor and M. Irani. Multi-view subspace constraints on homographies. In *ICCV99*, pages 710–715, 1999.