A Factorization Method for Affine Structure from Line Correspondences

Long Quan* Takeo Kanade
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract

A family of structure from motion algorithms called the factorization method has been recently developed from the orthographic projection model to the affine camera model [23, 16, 18]. All these algorithms are limited to handling only point features of the image stream. We propose in this paper an algorithm for the recovery of shape and motion from line correspondences by the factorization method with the affine camera. Instead of one step factorization for points, a multi-step factorization method is developed for lines based on the decomposition of the whole shape and motion into three separate substructures. Each of these substructures can then be linearly solved by factorizing the appropriate measurement matrices. It is also established that affine shape and motion with uncalibrated affine cameras can be achieved with at least seven lines over three views, which extends the previous results of Koenderink and Van Doorn [9] for points to lines.

1 Introduction

Points and line segments are generally considered as two most fundamental image features in vision. Line segments, compared with points are more global features, therefore more accurate, reliable and stable regarding to segmentation process. Pose estimation, stereo and structure from motion using line features have all been explored by a number of researchers [3, 1, 12, 11, 5, 27, 15]. However, most structure from motion algorithms using lines are limited to the minimal view case (three views), and have no closed form solutions. Their high non-linearity makes the algorithms sensitive to noise [12, 5]. The linear algorithms proposed in [11, 21, 8] were based on a heavy over-parametrization which still lead to unstable solutions.

In the last few years, a family of linear algorithms for structure from motion using highly redundant image sequences called the factorization method has been extensively studied [23, 26, 19, 16, 18] (the works [25, 9, 17, 10, 13, 25] are also closely related) for point features from orthographic projections to affine cameras. This kind of algorithm decomposes directly the feature points of the image stream into object shape and camera motion. Using simplified camera models from orthographic to affine, the principal gain is that the numerical computation is extremely well conditioned owing to the robust singular value decomposition algorithm. In this paper, we propose to extend these factorization algorithms for point features to line segment features.

The line factorization method in this paper will be developed with the affine camera model [14, 19, 18], so it will be naturally valid for all orthographic, weak perspective and para-perspective projection models.

Based on the previous results on the recovery of shape and motion from line correspondences using perspective cameras [12, 5], a four-step factorization algorithm instead of one step factorization for points will be developed taking advantage of the linearity of affine models. First, the whole structure to be recovered is decomposed into three components: (1) the rotations of the camera motion and the directions of 3D lines; (2) the translations of the camera motion; and (3) the other two d.o.f. of the 3D lines. As we are at first working with the uncalibrated affine camera, all these quantities are *uncalibrated*, this means that these quantities are primarily rather affine than Euclidean. Then each of these components can be linearly solved by factorizing the appropriate measurement matrices. It is interesting to observe that the very first step, as it will be clear later, is equivalent to a two dimensional projective reconstruction from one dimensional projective spaces. The second step will factorize the measurement matrix consisting of all the rescaled directions of image lines into the affine camera rotations and the affine directions of 3D lines. The third step turns out the affine translations by factorizing the measurement matrix obtained from the constraints on the camera motion. The fourth step factorizes the measurement matrix of interpretation planes into the space lines. All factorization can be nicely handled by SVD, hence can automatically deal with the singular or near to singular cases that may appear. We also establish that the minimal data required for the recovery of affine structure from line correspondences with the affine camera is seven lines over three views, which extends the previous results of Koenderink and Van Doorn [9] for points to lines.

2 The affine camera model: review

Throughout the paper, without explicit mention, capital letters in bold are generally used to denote matrices, and small case letters in bold denote vectors; small case letters and greek letters denote scalars.

As far as perspective cameras (pin-hole cameras) are concerned [4], the projection between a point $\mathbf{x} = (x, y, z, t)^T$ in $\mathcal{P}^3$ and a point $\mathbf{u} = (u, v, w)^T$ in $\mathcal{P}^2$ can

*On leave from GRAVIR-CNRS-INRIA, Grenoble, France

1063-6919/96 $5.00 © 1996 IEEE

be described by a 3×4 projection matrix $\mathbf{P}$ as

$$\lambda \mathbf{u} = \mathbf{P}\mathbf{x}, \tag{1}$$

which is a linear mapping in homogeneous coordinates.

For a restricted class of camera models, by seting the third row of the perspective camera $\mathbf{P}$ to $(0, 0, 0, \lambda)$, we obtain the affine camera initially introduced by Mundy and Zisserman in [14]

$$\mathbf{P} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & p_{34} \end{pmatrix} = \begin{pmatrix} \mathbf{M}_{2\times 3} & \mathbf{t}_{3\times 1} \\ \mathbf{0}_{1\times 3} & \end{pmatrix}.$$

For points not at infinity within affine spaces, $\mathbf{u} = (\overline{\mathbf{u}}, 1)^T = (\overline{u}, \overline{v}, 1)^T$, $\mathbf{x} = (\overline{\mathbf{x}}, 1)^T = (\overline{x}, \overline{y}, \overline{z}, 1)^T$ and $\mathbf{t} = (\overline{\mathbf{t}}, 1)^T = (a_{14}/a_{34}, a_{24}/a_{34}, 1)^T$, we have

$$\overline{\mathbf{u}} = \mathbf{M}\overline{\mathbf{x}} + \overline{\mathbf{t}}.$$

If we further use relative coordinates of the points with respect to a given reference point (for instance, the centroid of a set of points), the vector $\overline{\mathbf{t}}$ is canceled, therefore we have the following linear mapping between space points and image points:

$$\Delta\overline{\mathbf{u}} = \mathbf{M}\Delta\overline{\mathbf{x}}. \tag{2}$$

The affine camera generalizes the orthographic projection, weak perspective and para-perspective projections and preserves the affine properties. Several investigators [19, 18] have been interested in this model and achieved interesting results based on this concept.

3 Geometry of lines under affine camera

Now consider a line in $\mathcal{R}^3$ going through a point $\overline{\mathbf{x}}_0$ and of direction $\mathbf{d}_x$: $\overline{\mathbf{x}} = \overline{\mathbf{x}}_0 + \lambda\mathbf{d}_x$, which will be projected by $\mathbf{P}$ into an image line:

$$\mathbf{P}\begin{pmatrix} \overline{\mathbf{x}} \\ 1 \end{pmatrix} = (\mathbf{M}\overline{\mathbf{x}}_0 + \overline{\mathbf{t}}) + \lambda\mathbf{M}\mathbf{d}_x = \overline{\mathbf{u}}_0 + \lambda\mathbf{M}\mathbf{d}.$$

This line in image goes through the point

$$\overline{\mathbf{u}}_0 = \mathbf{M}\overline{\mathbf{x}}_0 + \overline{\mathbf{t}}$$

and has the direction

$$\mathbf{d}_u = \lambda\mathbf{M}\mathbf{d}_x. \tag{3}$$

This equation reflects nothing but the key property of the affine camera: lines parallel in 3D remain parallel in the image.

Now, let us consider how lines constrain the camera motion. It is well known that line correspondences from two views do not impose any constraint on camera motion, the minimum number of views required is three. If the interpretation plane of an image line for a given view is defined as the plane going through the line and the projection center, the well-known geometric interpretation of the constraint available for each line correspondence across three views (*cf.* [12, 5]) is that the interpretation planes from different views must intersect in a common line in space.

If the equation of a line in image is given by $\mathbf{l}^T\mathbf{u} = 0$, then substituting $\mathbf{u} = \lambda\mathbf{P}\mathbf{x}$ into it turns out the equation of the interpretation plane of $\mathbf{l}$ in space: $\mathbf{l}^T\mathbf{P}\mathbf{x} = 0$.

The plane is therefore given by the 4-vector $\mathbf{p}^T = \mathbf{l}^T\mathbf{P}$, which can also be expressed as $\mathbf{p}^T = (\mathbf{d}_x, d_x)^T$ where $\mathbf{d}_x$ is the normal vector of the plane.

For an image line of direction $\mathbf{d}_u$, it can be written as $\mathbf{l} = (\mathbf{d}_u, d_u)^T$, its interpretation plane is

$$\mathbf{p}^T = \mathbf{l}^T\mathbf{P} = (\mathbf{M}^T\mathbf{d}_u, \mathbf{l}^T\mathbf{t})^T. \tag{4}$$

Once the equation of the interpretation planes of lines are made explicit in terms of the image line and the projection matrix, the geometric constraint of line correspondence on the camera motion implies that 3×4 matrix whose rows are the three interpretation planes $(\mathbf{p}^T, \mathbf{p}'^T, \mathbf{p}''^T)^T$ has rank at most two. Hence, all of its 3×3 minors vanish. There are at total four 3×3 minors for a 3×4 matrix, it is well-known that these minors are not algebraically independent, are connected by the quadratic identities. There are only two of them independent.

The vanishing of any two such minors provide the two constraints on camera motion for a given line correspondence of three views. These constraints will be used to do the second step of factorization in Section 6.

4 Rescaling–step 0

Equation (3) relating image direction and space direction is the key equation for line factorization method. Note that Equation (3)—compared with Equation (1) describing a projection from $\mathcal{P}^3$ to $\mathcal{P}^2$—describes nothing but a projective projection from $\mathcal{P}^2$ to $\mathcal{P}^1$. This means that the affine reconstruction of lines with a two-dimensional affine camera is equivalent, partly, to the projective reconstruction of points with a one-dimensional perspective camera! The preliminary step for line factorization will be a two-dimensional projective reconstruction from one-dimensional projective spaces. This projective reconstruction will allows us to rescale properly the image directions for further submitting them to factorization.

This part is largely inspired by many recent works [24, 22, 6, 7, 20, 21] on the geometry of multi-views of two dimensional perspective camera, especially the approaches taken by Triggs and Sturm [24, 22]. We extend these ideas to one-dimensional camera. It turns out some interesting properties which were absent for 2 dimensional camera.

4.1 Matching constraints of one-dimensional perspective views

First, let's rewrite Equation (3) into Equation (5) using $\mathbf{u}$ and $\mathbf{x}$ instead of $\mathbf{d}_u$ and $\mathbf{d}_x$ to stress that we are dealing with points in projective space of lower dimensions $\mathcal{P}^2$ and $\mathcal{P}^1$ rather than line directions in vector spaces of higher dimensions $\mathcal{R}^3$ and $\mathcal{R}^2$:

$$\lambda \mathbf{u} = \mathbf{M}_{2\times3}\mathbf{x}. \tag{5}$$

This describes exactly a one-dimensional perspective camera which projects a point $\mathbf{x}$ in $\mathcal{P}^2$ onto a point $\mathbf{u}$ in $\mathcal{P}^1$.

We can now examine the matching constraints of multiple views of the same point. It is quite evident that there is no any constraints for two views. The minimum number of view that may have any geometric constraints is three.

Let the three views of the same point $\mathbf{x}$ be given as follows:

$$\left\{\begin{array}{rcl} \lambda\mathbf{u} & = & \mathbf{M}\mathbf{x} \\ \lambda'\mathbf{u}' & = & \mathbf{M}'\mathbf{x} \\ \lambda''\mathbf{u}'' & = & \mathbf{M}''\mathbf{x}. \end{array}\right.$$

This can be rewritten together in matrix form as

$$\begin{pmatrix} \mathbf{M} & \mathbf{u} & 0 & 0 \\ \mathbf{M}' & 0 & \mathbf{u}' & 0 \\ \mathbf{M}'' & 0 & 0 & \mathbf{u}'' \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ -\lambda \\ -\lambda' \\ -\lambda'' \end{pmatrix} = 0. \tag{6}$$

The vector $(\mathbf{x}, -\lambda, -\lambda', -\lambda'')^T$ can not be a zero vector, so that

$$\begin{vmatrix} \mathbf{M} & \mathbf{u} & 0 & 0 \\ \mathbf{M}' & 0 & \mathbf{u}' & 0 \\ \mathbf{M}'' & 0 & 0 & \mathbf{u}'' \end{vmatrix} = 0. \tag{7}$$

The expansion of this determinant turns out a trilinear constraints of three views

$$\sum_{i,j,k=1}^{2} T_{ijk}u_iu'_ju''_k = 0,$$

where T_{ijk} is a $2\times2\times2$ homogeneous tensor.

It can also be easily seen that there is no non-trivial quadrilinear constraints by adding more views as all of them reduce to the trilinearity. This proves the uniqueness of the trilinear constraint. Moreover, the $2\times2\times2$ homogeneous tensor T_{ijk} has $7 = 2\times2\times2-1$ d.o.f., so the tensor T_{ijk} is a minimal parametrization of three views since three views have exactly 7 d.o.f., up to a projective transformation in $\mathcal{P}^2$.

Each correspondence across three views gives one linear equation on the tensor T_{ijk}, with at least 7 points in $\mathcal{P}^1$, the tensor T_{ijk} can be linearly estimated.

4.2 Retrieving the projection matrices from the trilinearity

The geometry of the three views is more conveniently, also the most completely represented by the projection matrices associated with each view. In the previous section, the trilinear tensor was expressed in terms of the projection matrices. We are now seeking a map which goes back to projection matrix representation from the unique trilinear tensor of the three views.

Without loss of generality, we can always take the following normal forms of projection matrices for the set of three views

$$\begin{array}{rcl} \mathbf{M} & = & (\mathbf{I}_{2\times2} \;\; \mathbf{0})\,, \\ \mathbf{M}' & = & (\mathbf{A}_{2\times2} \;\; \mathbf{c}) \;\; = \;\; (\mathbf{a} \;\; \mathbf{b} \;\; \mathbf{c})\,, \\ \mathbf{M}'' & = & (\mathbf{D}_{2\times2} \;\; \mathbf{f}) \;\; = \;\; (\mathbf{d} \;\; \mathbf{e} \;\; \mathbf{f})\,. \end{array}$$

With such projection matrices, the trilinear tensor T_{ijk} is given by using (7) as

$$\lambda T_{ijk} = \pm(d_{\bar{i}j}c_{\bar{k}} - a_{\bar{i}j}f_{\bar{k}}), \tag{8}$$

where $i, j, k = 1, 2$; $\bar{i} = (i+1) \bmod 2$, the same for $\bar{j}$ and $\bar{k}$.

If we consider the tensor T_{ijk} as a 8-vector $(s_1, \ldots, s_8)^T$, the eight homogeneous equations of (8) can be rearranged by cancelling the common scalar λ into

$$\mathbf{G}\begin{pmatrix} \mathbf{d} \\ \mathbf{e} \\ \mathbf{f} \end{pmatrix} = 0, \tag{9}$$

where $\mathbf{G}$ is given by

$$\begin{pmatrix} 0 & s_8c_2 & -s_1c_1 & 0 & s_1b1 & -s_8a2 \\ 0 & 0 & -s_2c_1 & s_8c_2 & s_2b1 & -s_8b2 \\ 0 & -s_8c_1 & -s_3c_1 & 0 & s_3b1 & s_8a1 \\ 0 & 0 & -s_4c_1 & -s_8c_1 & s_4b1 & s_8b1 \\ -s_8c_2 & 0 & -s_5c_1 & 0 & s_8a2 + s_5b1 & 0 \\ 0 & 0 & -s_6c_1 - s_8c_2 & 0 & s_8b2 + s_6b1 & 0 \\ s_8c_1 & 0 & -s_7c_1 & 0 & -s_8a1 + s_7b1 & 0 \end{pmatrix}.$$

Since $(\mathbf{d}, \mathbf{e}, \mathbf{f})^T$ can not be zero vector, so all its 6×6 minors must vanish. There are 2 algebraically independent such minors, each of them gives a quadratic homogeneous polynomial in $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$. At this point, we are still unable to uniquely solve for $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ without further constraints. We can notice that $\mathbf{A}$ is only determined up to adding a matrix of form $\mathbf{c}\mathbf{v}^T$ for any 2-vector $\mathbf{v}$. Thus we can further constrain $\mathbf{A}$ such that $\mathbf{A}^T\mathbf{c} = 0$. This is equivalent to saying that the rank of $\mathbf{A}$ is one, i.e. we have $\mathbf{b} = k\mathbf{a}$ for a non-zero scalar k. This produces two scalar constraints on $\mathbf{a}$, $\mathbf{b}$. Together with the previous two quadratic constraints on $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$, we obtain a homogeneous quadratic equation in a_1 and a_2:

$$\alpha a_1^2 + \beta a_1a_2 + \gamma a_2^2 = 0, \tag{10}$$

where $\alpha = s_3s_8 - s_4s_7$, $\beta = s_7s_2 + s_5s_4 - s_3s_6 - s_8s_1$ and $\gamma = s_6s_1 - s_2s_5$, and this quadratic equation may be easily solved for a_1/a_2.

Then k is given by the following linear equation in terms of a_1/a_2

$$(-a_2s_1 + s_3a_1)k + (s_2a_2 - s_4a_1) = 0.$$

After that, the 2-vector $\mathbf{c}$ is obtained by solving $\mathbf{A}^T\mathbf{c} = 0$. Thus, the projection matrix $\mathbf{M}'$ is fully recovered up to two solutions.

Finally, the 6-vector $(\mathbf{d}, \mathbf{e}, \mathbf{f})^T$ for the projection matrix $\mathbf{M}''$ is linearly solved by Equation (9) in terms of $\mathbf{M}'$.

4.3 Rescaling of one-dimensional image points

According to Triggs and Sturm [24, 22], projective reconstruction is equivalent to the rescaling of the image points. For each image point through three views $\lambda^{(j)}\mathbf{u}^{(j)} = \mathbf{M}^{(j)}\mathbf{x}$, the scaling factors $\lambda^{(j)}$, taken individually, are arbitrary; however, taken as a whole, they encode all the projective structure of all views and points.

One way to recover the rescaling factors, up to a scaling factor, is directly to use the basic reconstruction equation (6) or alternatively to observe the following matrix identity:

$$\begin{pmatrix} \mathbf{M} & \lambda\mathbf{u} \\ \mathbf{M}' & \lambda'\mathbf{u}' \\ \mathbf{M}'' & \lambda''\mathbf{u}'' \end{pmatrix} = \begin{pmatrix} \mathbf{M} \\ \mathbf{M}' \\ \mathbf{M}'' \end{pmatrix} (\mathbf{I}_{3\times 3} \quad \mathbf{x}).$$

The rank of the left matrix is therefore at most 3. All 4×4 minors vanish, three of them are algebraically independent. Each can be expanded by cofactors in the last column to obtain a linear homogeneous equation in $\lambda, \lambda', \lambda''$. Therefore $(\lambda, \lambda', \lambda'')^T$ can be linearly solved up to a scalar by

$$\begin{pmatrix} * & * & * \\ * & * & * \\ * & * & * \end{pmatrix} \begin{pmatrix} \lambda \\ \lambda' \\ \lambda'' \end{pmatrix} = 0,$$

where $*$ designate a known constant entry in the matrix.

For each triplet of views, the image points can be consistently rescaled according to the previous paragraphe. For general $n > 3$ view case, we can take appropriate triplets among n views such that any two triplets has at least a common view. Then, the rescaling factors for any given point of all triplets of views can be chained together over n views.

5 Direction factorization–step 1

Suppose we are given m line correspondences in n views. The view is indexed by a superscript number and the feature by a subscript one. We can now create the $2n \times m$ measurement matrix $\mathbf{W}_D$ of all lines of all views by stacking the direction vectors $\mathbf{d}_u^{(j)}{}_i$ properly rescaled by $\lambda_i^{(j)}$ as follows:

$$\mathbf{W}_D = \begin{pmatrix} \lambda_1\mathbf{d}_{u_1} & \lambda_2\mathbf{d}_{u_2} & \cdots & \lambda_m\mathbf{d}_{u_m} \\ \lambda_1'\mathbf{d}_{u_1}' & \lambda_2'\mathbf{d}_{u_2}' & \cdots & \lambda_m'\mathbf{d}_{u_m}' \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1^{(n)}\mathbf{d}_{u_1}^{(n)} & \lambda_2^{(n)}\mathbf{d}_{u_2}^{(n)} & \cdots & \lambda_n^{(n)}\mathbf{d}_{u_m}^{(n)} \end{pmatrix}.$$

As the following matrix equation holds for the measurement matrix $\mathbf{W}_D$:

$$\mathbf{W}_D = \mathbf{M}_D\mathbf{D}_D = \begin{pmatrix} \mathbf{M} \\ \mathbf{M}' \\ \vdots \\ \mathbf{M}^{(n)} \end{pmatrix} (\mathbf{d}_{x_1} \quad \mathbf{d}_{x_2} \quad \cdots \quad \mathbf{d}_{x_m}),$$

the rank of $\mathbf{W}_D$ is therefore at most of three. The factorization method can then be applied to $\mathbf{W}_D$.

Let

$$\mathbf{W}_D = \mathbf{U}_D\Sigma_D\mathbf{V}_D^T$$

be the SVD factorization of $\mathbf{W}_D$, the 3×3 diagonal matrix Σ_{D3} be obtained by keeping the first three singular values (assuming that singular values are ordered) of Σ and $\mathbf{U}_{D3}$ ($\mathbf{V}_{D3}$) be the first 3 columns (rows) of $\mathbf{U}$ ($\mathbf{V}$).

Then, the product $\mathbf{U}_{D3}\Sigma_{D3}\mathbf{V}_{D3}^T$ gives the best rank 3 approximation to $\mathbf{W}_D$.

One possible solution for $\hat{\mathbf{M}}$ and $\hat{\mathbf{D}}$ may be taken to be

$$\hat{\mathbf{M}} = \mathbf{U}_{D3}\Sigma_{D3}^{1/2} \quad \text{and} \quad \hat{\mathbf{D}} = \Sigma_{D3}^{1/2}\mathbf{V}_{D3}.$$

For any nonsingular 3×3 matrix $\mathbf{A}_{3\times 3}$, $\hat{\mathbf{M}}' = \hat{\mathbf{M}}\mathbf{A}_{3\times 3}$ and $\hat{\mathbf{D}}' = \mathbf{A}_{3\times 3}^{-1}\hat{\mathbf{D}}$ are also a valid solution, as we have

$$\hat{\mathbf{M}}\mathbf{A}\mathbf{A}^{-1}\hat{\mathbf{D}} = \hat{\mathbf{M}}'\hat{\mathbf{D}}' = \hat{\mathbf{M}}\hat{\mathbf{D}}.$$

This means that the recovered direction matrix $\hat{\mathbf{D}}$ and the rotation matrix $\hat{\mathbf{M}}$ are defined only up to affine transformations.

6 Translation factorization–step 2

Once we obtained the affine line directions and affine rotations of the camera motion from the first factorization step, we proceed to the second step to obtain scaled affine translations from factorization.

For each interpretation plane $\mathbf{l}^T\mathbf{P} = (\mathbf{d}_x, d_x)^T$ of each image line, its direction component is completely determined with the recovered $\hat{\mathbf{M}}$ from equation (4) as

$$\mathbf{d}_x = \hat{\mathbf{M}}^T\mathbf{d}_u.$$

Actually only its fourth component $d_x = \mathbf{l}^T\mathbf{t}$ remains undermined, depending linearly on $\mathbf{t}$. The interpretation plane can be properly written as

$$\mathbf{p}^T = (\lambda\mathbf{M}^T\mathbf{d}_u, \mu\mathbf{l}^T\mathbf{t})^T.$$

We can then stack all the planes of different views for a given line as the following $n \times 4$ measurement matrix of planes:

$$\mathbf{W}_P = \begin{pmatrix} * & * & * & \mathbf{l}^T\mathbf{t} \\ * & * & * & \mathbf{l}'^T\mathbf{t}' \\ \vdots & \vdots & \vdots & \vdots \\ * & * & * & \mathbf{l}^{(n)^T}\mathbf{t}^{(n)} \end{pmatrix}.$$

This matrix $\mathbf{W}_P$ geometrically represents the pencil of planes, so it has at most rank 2. For any three rows, we can have two independent minors, as one by taking the first three columns is always a constant, there remains only one independent minor involving the $\mathbf{t}^{(i)}$.

Expanding each minor, formed by any three rows i, j and k of $\mathbf{W}_P$, by cofactors in the last column gives an homogeneous linear equation in $\mathbf{t}^{(i)}$, $\mathbf{t}^{(j)}$ and $\mathbf{t}^{(k)}$:

$$(\times \quad \times \quad \times) \begin{pmatrix} \mathbf{t}^{(i)} \\ \mathbf{t}^{(j)} \\ \mathbf{t}^{(k)} \end{pmatrix} = 0,$$

where the "$\times$" designate 3 constants in a row.

For all those vanishing minors, collecting them together, we obtain

$$\begin{pmatrix} \times & \times & \times & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & \times & \times & \times \end{pmatrix} \begin{pmatrix} \mathbf{t} \\ \mathbf{t}' \\ \vdots \\ \mathbf{t}^{(n)} \end{pmatrix} = 0.$$

At this stage, since the origin of the coordinate frame in space is not yet fixed, we may take $\mathbf{t} = (0,0,1)^T$ up to a scaling factor, say t_0, so the final homogeneous linear equations to solve for $(t_0, \mathbf{t}', \ldots, \mathbf{t}^{(n)})^T$ is

$$\mathbf{W}_T \begin{pmatrix} t_0 \\ \mathbf{t}' \\ \vdots \\ \mathbf{t}^{(n)} \end{pmatrix} = \begin{pmatrix} * & \times & \times & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & \times & \times & \times \end{pmatrix} \begin{pmatrix} t_0 \\ \mathbf{t}' \\ \vdots \\ \mathbf{t}^{(n)} \end{pmatrix} = 0.$$

Once again, this system of equations can be nicely solved by SVD factorizing $\mathbf{W}_T$.

At this stage, apart from the undermined overall scaling for the computed $(t_0, \mathbf{t}', \mathbf{t}'', \cdots)^T$, it is still ambiguous up to a sign, as $-(t_0, \mathbf{t}', \mathbf{t}'', \cdots)^T$ is also a valid solution. This sign-inversed solution geometrically reflects the shape in space.

7 Shape factorization–step 3

Once $(t_0, \mathbf{t}', \mathbf{t}'', \cdots)^T$ are recovered by step 2, together with the results of step 1, the projection matrices of all views are completely determined up to one common scaling factor. The matrix $\mathbf{W}_P$ containing all interpretation planes is also completely determined.

Two methods to obtain the shape are possible, one based on the projective representation of lines and another on the minimal representation of lines, inspired by [5]. Due to space limitation, only the first method will be described here.

A projective line in space can be defined either by a pencil of planes (two projective planes define a pencil of planes); or by any of its two points.

If the rank of the matrix $\mathbf{W}_P$ is 2, its nullity is also 2. The range of $\mathbf{W}_P$ define the pencil of planes and the null space spans the projective line in space.

Once again, using SVD to factorize $\mathbf{W}_P$ gives everything we want. Let

$$\mathbf{W}_P = \mathbf{U}_P \Sigma_P \mathbf{V}_P^T$$

be the SVD of $\mathbf{W}_P$ with ordered singular values. Two points of the line might be taken to be $\mathbf{v}_3$ and $\mathbf{v}_4$, so the line is given by

$$\lambda \mathbf{v}_3 + \mu \mathbf{v}_4.$$

One advantage of this method is that using subset selection, near singular views can be detected and discarded.

8 Calibrated affine camera

Up to this point, we have worked with uncalibrated affine camera, the recovered motion and the shape are defined up to an affine transformation. If the affine camera is calibrated, then it is possible to directly obtain Euclidean shape and motion.

Following the intrinsic $\mathbf{K}$ and extrinsic $\mathbf{R}$ decomposition of $\mathbf{M} = \mathbf{KR}$ introduced in [18], the whole metric information from the calibrated affine camera is contained in the affine intrinsic parameters $\mathbf{KK}^T$. The affine motion matrix $\hat{\mathbf{M}} = \begin{pmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \end{pmatrix}$ is constraint by

$$\begin{pmatrix} \mathbf{m}_1 \mathbf{A}\mathbf{A}^T \mathbf{m}_1 & \mathbf{m}_1 \mathbf{A}\mathbf{A}^T \mathbf{m}_2 \\ \mathbf{m}_1 \mathbf{A}\mathbf{A}^T \mathbf{m}_2 & \mathbf{m}_2 \mathbf{A}\mathbf{A}^T \mathbf{m}_2 \end{pmatrix} = \mathbf{KK}^T.$$

The linear solutions may be expected if we solve for the entries of $\mathbf{AA}^T$, however it may happen the linear estimation of $\mathbf{AA}^T$ be not positive-definite due to noise. The alternative non-linear solution using Cholesky parametrization that ensures the positive-definiteness can be found in [18].

Once we obtain the appropriate $\hat{\mathbf{A}}$, then $\hat{\mathbf{M}}\hat{\mathbf{A}}$ and $\hat{\mathbf{A}}^{-1}\hat{\mathbf{D}}$ carry the rotations of the camera and the directions of lines.

The other steps remain the same as for uncalibrated affine camera.

9 Minimum data case

Although the general context of the development of the line factorization method is focused on using the heavily redundant image features and views, the minimal data required for such achievement is equally important and interesting.

The minimum numbers of views and lines required at each step are summarized in Table 9.

Step	views #	lines #
0 (rescaling)	3	7
1 (direction)	2	5
2 (translation)	3	6

Table 1: The minimum numbers of view and lines required at each step of the factorization.

In view of this, we can establish the following.

For the recovery of affine shape and affine motion from line correspondences with an uncalibrated affine camera, the minimum number of views needed is three and the minimum number of lines required is seven for a linear solution.

Note the difference with the the perspective camera case. A minimum of 6 lines is required to have non-linear algorithms with the perspective camera and to have a linear solution, a minimum of 13 lines is required across three views, as have been reported in

[12, 11, 5]. It is important to note that with the affine camera and the method presented in this paper, the number of line correspondences for achieving a linear solution is reduced from thirteen to seven, which presents an important practical significance.

10 Experimental results

We first used the simulated image to validate the theoretical development of the algorithm. The simulation is set up as follows. We first use the perspective projection matrices obtained by calibration. Then these projection matrices are approximated to the affine projection ones. A set of 3D line segments lying on two visible faces of a cube is finally projected into sets of 2D line segments. Each 2D line segment is perturbed by adding different level of noise to create the final simulated images. The algorithm turns out very good reconstruction results up to two pixel noise, then the results degrade with increasing noises. Figures 1 show the reconstruction results with one pixel noise. Each 3D line segment is displayed by its two endpoints. The endpoints are obtained by backprojecting the endpoints of the line segment in one of the views onto the 3D line.

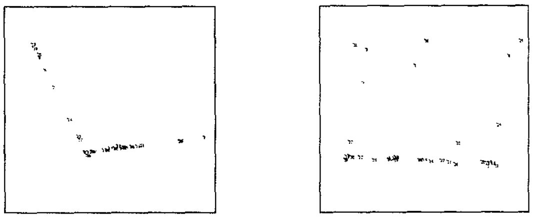

Figure 1: The top and side views of the reconstructed affine shape of the simulation with one pixel noise.

11 Discussions

We have presented a four step factorization algorithm for the recovery of shape and motion from line correspondences with an affine camera. This algorithm extends the previous algorithms for points to line features which can be more accurately extracted from images. The method is based on the decomposition of the whole structure into three separate parameter sets, each one is then solved by factorizing the different measurement matrices. Although the four steps are sequential, the first two steps are based on very stable and accurate measures, so the accuracy for the following steps is hardly influenced.

As the line factorization algorithm presented in this paper is developed within the same framework as suggested in [18] for points, it is therefore straightforward to integrate both points and lines into the same framework.

Acknowledgement

This work is partly supported by CNRS and French Ministère de l'Education which are gratefully acknowledged.

References

[1] N. Ayache. Stereovision and sensor fusion. MIT-Press, 1990.

[2] M. Spetsakis ans J. Aloimonos. Structure from motion using line correspondences. IJCV, 4:171–183, 1990.

[3] M. Dhome, M. Richetin, J.T. Lapresté, and G. Rives. Determination of the attitude of 3D objects from sigle perspective view. Ieee T-PAMI, 11(12):1265–1278, 1989.

[4] O. Faugeras. Three-Dimensional Computer Vision - A Geometric Viewpoint. Artificial intelligence. M.I.T. Press, Cambridge, MA, 1993.

[5] O.D. Faugeras, F. Lustman, and G. Toscani. Motion and structure from point and line matches. In Proc. the 1st ICCV, London, England, June 1987.

[6] O. Faugeras and B. Mourrain. About the correspondence of points between N images. In Proceedings of IEEE Workshop on Representations of Visual Scenes, Cambridge, Massachusetts, USA, June 1995.

[7] R. Hartley. Lines and Points in Three Views - An Integrated Approach. Technical report, G.E. CRD, 1994.

[8] R.I. Hartley. Projective reconstruction from line correspondences. In Proc. CVPR, Seattle, Washington, USA, 1994.

[9] J.J. Koenderink and A. J. Van Doorn. Affine structure from motion. Technical report, Utrecht University, Utrecht, The Netherlands, October 1989.

[10] C.H. Lee and T. Huang. Finding point correspondences and determining motion of a rigid object from two weak perspective views. CVGIP, 52:309–327, 1990.

[11] Y. Liu and T.S. Huang. A linear algorithm for motion estimation using straight line correspondences. CVGIP, 44(1):35–57, October 1988.

[12] Y. Liu and T.S. Huang. Estimation of rigid body motion using straight line correspondences. CVGIP, 43(1):37–52, July 1988.

[13] Ph.F. McLauchlan, I.D. Reid and D.W. Murray. Recursive affine structure and motion from image sequences. In J.O. Eklundh, editor, Proc. the 3rd ECCV, Stockholm, Sweden, pages 217–224. Springer-Verlag, May 1994.

[14] J.L. Mundy and A. Zisserman, editors. Geometric Invariance in Computer Vision. MIT Press, Cambridge, Massachusetts, USA, 1992.

[15] T.Q. Phong, R. Horaud, A. Yassine, and D. T. Pham. Optimal Estimation of Object Pose from a Single Perspective View. In Proc. the 4th ICCV, Berlin, Germany, May 1993.

[16] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In J.O. Eklundh, editor, Proc. the 3rd ECCV, Stockholm, Sweden, pages 97–108, May 1994.

[17] L. Quan and R. Mohr. Affine shape representation from motion through reference points. JMIV, 1:145–151, 1992. also in IEEE Workshop on Visual Motion, New Jersey, pages 249–254, 1991.

[18] L. Quan. Self-calibration of an affine camera from multiple views. IJCV, 1995. to appear.

[19] L.S. Shapiro, A. Zisserman, and M. Brady. Motion from point matches using affine epipolar geometry. IJCV, 1994.

[20] A. Shashua. Algebraic functions for recognition. Ieee Transactions on PAMI, 1994. in press.

[21] M. Spetsakis and J. Aloimonos. A Unified theory of structure from motion. In Proceedings DARPA IU Workshop, 1990.

[22] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In Proc. the 4th ECCV, Cambridge, England, 1996.

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

[24] B. Triggs. The geometry of projective reconstruction I: Matching constraints and the joint image. In Proc. the 5th ICCV, Cambridge, Massachusetts, USA, 1995.

[25] S. Ullman. The Interpretation of Visual Motion. The MIT Press, 1979.

[26] D. Weinshall and C. Tomasi. Linear and incremental acquisition of invariant shape models from image sequences. In Proceedings of the 4th International Conference on Computer Vision, Berlin, Germany. Ieee, 1993.

[27] Z. Zhang and O. Faugeras. Three-dimensional motion computation and object segmentation in a long sequence of stereo frames. IJCV, 7(3):211–241, 1992.