Memoirs of the Faculty of Engineering, Okayama University, Vol. 40, pp. 53–63, January 2006

# *Uncalibrated Factorization Using a Variable Symmetric Affine Camera*

Kenichi KANATANI*, Yasuyuki SUGAYA and Hanno ACKERMANN
Department of Computer Science, Okayama University
Okayama 700-8530 Japan

(Received December 7, 2005)

In order to reconstruct 3-D Euclidean shape by the Tomasi-Kanade factorization, one needs to specify an affine camera model such as orthographic, weak perspective, and paraperspective. We present a new method that does not require any such specific models. We show that a minimal requirement for an affine camera to mimic perspective projection leads to a unique camera model, which we call a *symmetric affine camera*, which has two free functions. We determine their values from input images by linear computation and demonstrate by experiments that an appropriate camera model is automatically selected.

## 1. Introduction

One of the best known techniques for 3-D reconstruction from feature point tracking through a video stream is the Tomasi-Kanade *factorization* [20], which computes the 3-D shape of the scene by approximating the camera imaging by an affine transformation. The computation consists of linear calculus alone without involving iterations (see [10] for the computational details). The solution is sufficiently accurate for many practical purposes and is used as an initial solution for more sophisticated iterative reconstruction based on perspective projection [3].

If the camera model is not specified, other than being affine, the 3-D shape is computed only up to an affine transformation, known as *affine reconstruction*. For computing the correct shape (*Euclid reconstruction*[1]), we need to specify the camera model. For this, *orthographic*, *weak perspective*, and *paraperspective* projections have been used [12]. However, the reconstruction accuracy does not necessarily follow that order [2]. To find the best camera models in a particular circumstance, one needs to choose the best one *a posteriori*. Is there any method for automatically selecting an appropriate camera model? This is the motivation of this paper.

Basri [1] pointed out that any affine camera can be regarded as paraperspective projection if the scale and the reference point are appropriately adjusted, and Sugimoto [19] exploited this fact for object recognition from a single image. Shapiro et al. [14] described the epipolar geometry for affine cameras and 3-D reconstruction methods based on it. Quan [13] showed that a generic affine camera has three intrinsic parameters and that they can be determined by self-calibration if the same camera is moved (i.e., the three intrinsic parameters are unchanged).

This paper extends Quan's result to variable intrinsic parameters. However, these three parameters cannot be determined if they vary freely, i.e., if the camera is completely arbitrary from frame to frame. The situation is similar to the *dual absolute quadric constraint* [3] for upgrading projective reconstruction to Euclidean, which cannot be imposed unless minimal constraints are imposed on the internal parameters (e.g., zero skew).

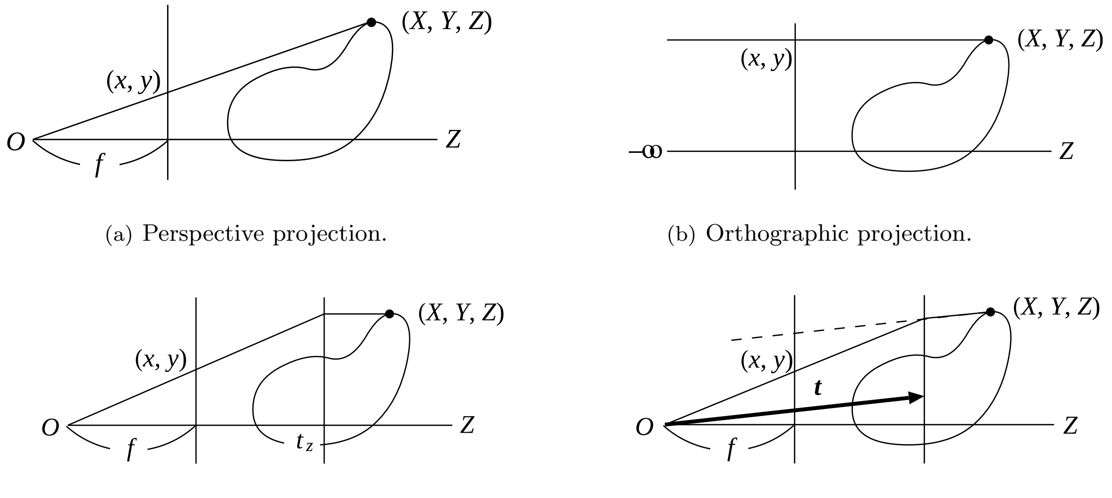
In this paper, we show that minimal requirements for the general affine camera to mimic perspective projection leads to a unique camera model, which we call a *symmetric affine camera*, having two free functions of motion parameters; specific choices of their function forms result in the orthographic, weak perspective, and paraperspective models.

Here, however, we do not specify such function forms. We determine their values directly from input images, taking advantage of the fact that at most two time varying quantities can be eliminated from the generic metric constraint. As a result, all the computation is linear just as in the case of the traditional factorization method, and an appropriate model is automatically selected.

Sec. 2 summarizes fundamentals of affine cameras, and Sec. 3 summarizes the metric constraint. In Sec. 4, we derive our symmetric affine camera model.

*E-mail kanatani@suri.it.okayama-u.ac.jp

[1]Since the absolute scale is indeterminate, this should strictly be called *similarity reconstruction*, but the term "Euclidean" is widely used now.




(a) Perspective projection. (b) Orthographic projection.

(c) Weak perspective projection. (d) Paraperspective projection.

Figure 1: Camera models.

Sec. 5 describes the procedure for 3-D reconstruction using our model. Sec. 6 shows experiments, and Sec. 7 concludes this paper. The full computational details of our procedure are given in Appendix.

## 2. Affine Cameras

We first summarize fundamentals about affine cameras.

Consider a camera-based $XYZ$ coordinate system with the origin $O$ at the projection center and the $Z$ axis along the optical axis. *Perspective projection* maps a point $(X, Y, Z)$ in the scene onto a point in the image with coordinates $(x, y)$ such that

$$x = f\frac{X}{Z}, \qquad y = f\frac{Y}{Z}, \tag{1}$$

where $f$ is a constant called the *focal length* (Fig. 1(a)).

Consider a world coordinate system fixed to the scene, and let $\boldsymbol{t}$ and $\{\boldsymbol{i},\ \boldsymbol{j},\ \boldsymbol{k}\}$ be its origin and the orthonormal basis vectors described with respect to the camera coordinate system. For convenience (with some risk of confusion), we call $\boldsymbol{t}$ the *translation*, the matrix $\boldsymbol{R} = \begin{pmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{pmatrix}$ having $\{\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}\}$ as columns the *rotation*, and $\{\boldsymbol{t},\ \boldsymbol{R}\}$ the *motion parameters*. However, we should always keep in mind that all the descriptions in this paper are *with respect to the camera coordinate system*, not the world coordinate system[2].

If

(i) the object of our interest is localized around the world coordinate origin $\boldsymbol{t}$, and

(ii) the size of the object is small as compared with $\|\boldsymbol{t}\|$,

the imaging can be approximated by an *affine camera* [14] in the form

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \Pi_{11} & \Pi_{12} & \Pi_{13} \\ \Pi_{21} & \Pi_{22} & \Pi_{23} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} \pi_1 \\ \pi_2 \end{pmatrix}. \tag{2}$$

We call the $2\times3$ matrix $\boldsymbol{\Pi} = (\Pi_{ij})$ and the 2-D vector $\boldsymbol{\pi} = (\pi_i)$ the *projection matrix* and the *projection vector*, respectively; their elements are "functions" of the motion parameters $\{\boldsymbol{t},\ \boldsymbol{R}\}$. Unlike Quan [13], we do not separate "intrinsic" parameters from the motion parameters (or "extrinsic" parameters); the intrinsic parameters are *implicitly* defined via the *functional forms* of $\{\boldsymbol{\Pi},\ \boldsymbol{\pi}\}$ on $\{\boldsymbol{t},\ \boldsymbol{R}\}$, i.e., as "coefficients" in them. Typical affine cameras are

**Orthographic projection** (Fig. 1(b))

$$\boldsymbol{\Pi} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \qquad \boldsymbol{\pi} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{3}$$

**Weak perspective projection** (Fig. 1(c))

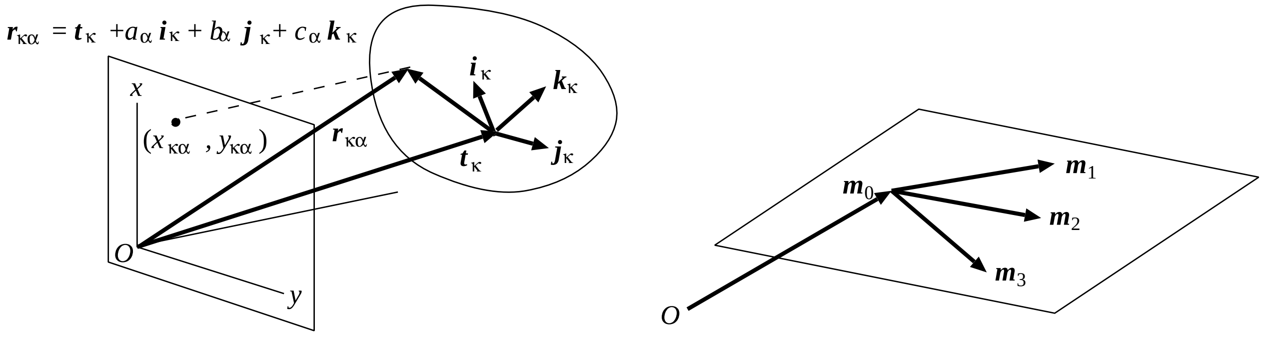
$$\boldsymbol{\Pi} = \begin{pmatrix} f/t_z & 0 & 0 \\ 0 & f/t_z & 0 \end{pmatrix}, \quad \boldsymbol{\pi} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{4}$$

**Paraperspective projection** (Fig. 1(d))

$$\boldsymbol{\Pi} = \begin{pmatrix} f/t_z & 0 & -ft_x/t_z^2 \\ 0 & f/t_z & -ft_x/t_z^2 \end{pmatrix}, \quad \boldsymbol{\pi} = \begin{pmatrix} ft_x/t_z \\ ft_y/t_z \end{pmatrix}. \tag{5}$$

Suppose we track $N$ feature points over $M$ frames. Identifying the frame number $\kappa$ with "time", let $\boldsymbol{t}_\kappa$ and $\{\boldsymbol{i}_\kappa,\ \boldsymbol{j}_\kappa,\ \boldsymbol{k}_\kappa\}$ be the origin and the basis vectors of the world coordinate system at time $\kappa$ (Fig. 2(a)).

[2] The Tomasi-Kanade factorization adopts the standpoint that the camera is moving relative to a stationary object [11, 12, 20, 21]. Although both are mathematically equivalent, our formulation is better suited for the subsequent analysis.




(a) (b)

Figure 2: (a) Camera-based description of the world coordinate system. (b) Affine space constraint.

The 3-D position of the $\alpha$th point at time $\kappa$ has the form

$$\boldsymbol{r}_{\kappa\alpha} = \boldsymbol{t}_\kappa + a_\alpha \boldsymbol{i}_\kappa + b_\alpha \boldsymbol{j}_\kappa + c_\alpha \boldsymbol{k}_\kappa. \tag{6}$$

Under the affine camera of eq. (2), its image coordinates $(x_{\kappa\alpha}, y_{\kappa\alpha})$ are given by

$$\begin{pmatrix} x_{\kappa\alpha} \\ y_{\kappa\alpha} \end{pmatrix} = \tilde{\boldsymbol{t}}_\kappa + a_\alpha \tilde{\boldsymbol{i}}_\kappa + b_\alpha \tilde{\boldsymbol{j}}_\kappa + c_\alpha \tilde{\boldsymbol{k}}_\kappa, \tag{7}$$

where $\tilde{\boldsymbol{t}}_\kappa$, $\tilde{\boldsymbol{i}}_\kappa$, $\tilde{\boldsymbol{j}}_\kappa$, and $\tilde{\boldsymbol{k}}_\kappa$ are 2-D vectors defined by

$$\tilde{\boldsymbol{t}}_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{t}_\kappa + \boldsymbol{\pi}_\kappa, \tag{8}$$

$$\tilde{\boldsymbol{i}}_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{i}_\kappa, \quad \tilde{\boldsymbol{j}}_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{j}_\kappa, \quad \tilde{\boldsymbol{k}}_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{k}_\kappa. \tag{9}$$

Here, $\boldsymbol{\Pi}_\kappa$ and $\boldsymbol{\pi}_\kappa$ are the projection matrix and the projective vector, respectively, at time $\kappa$. The motion history of the $\alpha$th point is represented by a vector

$$\boldsymbol{p}_\alpha = \begin{pmatrix} x_{1\alpha} & y_{1\alpha} & x_{2\alpha} & y_{2\alpha} & \dots & x_{M\alpha} & y_{M\alpha} \end{pmatrix}^\top, \tag{10}$$

which we simply call the *trajectory* of that point. Using eq. (7), we can write

$$\boldsymbol{p}_\alpha = \boldsymbol{m}_0 + a_\alpha \boldsymbol{m}_1 + b_\alpha \boldsymbol{m}_2 + c_\alpha \boldsymbol{m}_3, \tag{11}$$

where $\boldsymbol{m}_0$, $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, and $\boldsymbol{m}_3$ are the $2M$-dimensional vectors defined, respectively, by

$$\begin{pmatrix} \tilde{\boldsymbol{t}}_1 \\ \tilde{\boldsymbol{t}}_2 \\ \vdots \\ \tilde{\boldsymbol{t}}_M \end{pmatrix}, \quad \begin{pmatrix} \tilde{\boldsymbol{i}}_1 \\ \tilde{\boldsymbol{i}}_2 \\ \vdots \\ \tilde{\boldsymbol{i}}_M \end{pmatrix}, \quad \begin{pmatrix} \tilde{\boldsymbol{j}}_1 \\ \tilde{\boldsymbol{j}}_2 \\ \vdots \\ \tilde{\boldsymbol{j}}_M \end{pmatrix}, \quad \begin{pmatrix} \tilde{\boldsymbol{k}}_1 \\ \tilde{\boldsymbol{k}}_2 \\ \vdots \\ \tilde{\boldsymbol{k}}_M \end{pmatrix}. \tag{12}$$

Thus, all the trajectories $\{\boldsymbol{p}_\alpha\}$ are constrained to be in the 3-D affine space $\mathcal{A}$ in $\mathcal{R}^{2M}$ passing through $\boldsymbol{m}_0$ and spanned by $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, and $\boldsymbol{m}_3$ (Fig. 2(b)). This fact is known as the *affine space constraint*, which is also the basis for multi-body motion segmentation [7, 8, 9, 15, 16, 17, 18].

## 3. Metric Constraint

Next, we summarize the metric constraint on affine cameras.

Since the world coordinate system can be placed arbitrarily, we let its origin coincide with the centroid of the $N$ feature points. This implies $\sum_{\alpha=1}^N a_\alpha = \sum_{\alpha=1}^N b_\alpha = \sum_{\alpha=1}^N c_\alpha = 0$, so we have from eq. (11)

$$\frac{1}{N}\sum_{\alpha=1}^N \boldsymbol{p}_\alpha = \boldsymbol{m}_0, \tag{13}$$

i.e., $\boldsymbol{m}_0$ is the centroid of the trajectories $\{\boldsymbol{p}_\alpha\}$ in $\mathcal{R}^{2M}$. It follows that the deviation $\boldsymbol{p}'_\alpha$ of $\boldsymbol{p}_\alpha$ from the centroid $\boldsymbol{m}_0$ is written as[3]

$$\boldsymbol{p}'_\alpha = \boldsymbol{p}_\alpha - \boldsymbol{m}_0 = a_\alpha \boldsymbol{m}_1 + b_\alpha \boldsymbol{m}_2 + c_\alpha \boldsymbol{m}_3, \tag{14}$$

which means that $\{\boldsymbol{p}'_\alpha\}$ are constrained to be in the 3-D subspace $\mathcal{L}$ in $\mathcal{R}^{2M}$. Hence, the (second-order) *moment matrix*[4]

$$\boldsymbol{C} = \sum_{\alpha=1}^N \boldsymbol{p}'_\alpha \boldsymbol{p}'^{\top}_\alpha \tag{15}$$

is of rank 3, having three nonzero eigenvalues. The corresponding unit eigenvectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$ constitute an orthonormal basis of the subspace $\mathcal{L}$, and $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, and $\boldsymbol{m}_3$ are expressed as a linear combination of them in the form

$$\boldsymbol{m}_i = \sum_{j=1}^3 A_{ji} \boldsymbol{u}_j. \tag{16}$$

[3] In the traditional formulation [11, 12, 20, 21], vectors $\{\boldsymbol{p}'_\alpha\}$ are combined into the *observation* (or *measurement*) *matrix*, $\boldsymbol{W} = \begin{pmatrix} \boldsymbol{p}'_1 & \dots & \boldsymbol{p}'_N \end{pmatrix}$, and the object coordinates $\{(a_\alpha, b_\alpha, c_\alpha)\}$ are combined into the *shape matrix*, $\boldsymbol{S} = \begin{pmatrix} a_1 & \dots & a_N \\ b_1 & \dots & b_N \\ c_1 & \dots & c_N \end{pmatrix}$. Then, eq. (14) is written as $\boldsymbol{W} = \boldsymbol{M}\boldsymbol{S}$, where $\boldsymbol{M}$, the *motion matrix*, is defined by the first of eqs. (17). However, our formulation is better suited for the subsequent analysis.

[4] This matrix is called by many different names such as the "covariance matrix" and the "scatter matrix".

Let $\boldsymbol{M}$ and $\boldsymbol{U}$ be the $2M \times 3$ matrices consisting of $\{\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3\}$ and $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$ as columns:

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{m}_1 & \boldsymbol{m}_2 & \boldsymbol{m}_3 \end{pmatrix}, \quad \boldsymbol{U} = \begin{pmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \boldsymbol{u}_3 \end{pmatrix}. \tag{17}$$

From eq. (16), $\boldsymbol{M}$ and $\boldsymbol{U}$ are related by the matrix $\boldsymbol{A} = (A_{ij})$ in the form[5]:

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{A}. \tag{18}$$

The rectifying matrix $\boldsymbol{A} = (A_{ij})$ should be determined so that $\boldsymbol{m}_1$, $\boldsymbol{m}_2$ and $\boldsymbol{m}_3$ in eq. (12) are projections of the orthonormal basis vectors $\{\boldsymbol{i}_\kappa, \boldsymbol{j}_\kappa, \boldsymbol{k}_\kappa\}$ in the form of eqs. (9). From eq. (9), we obtain

$$\begin{pmatrix} \tilde{\boldsymbol{i}}_\kappa & \tilde{\boldsymbol{j}}_\kappa & \tilde{\boldsymbol{k}}_\kappa \end{pmatrix} = \boldsymbol{\Pi}_\kappa \begin{pmatrix} \boldsymbol{i}_\kappa & \boldsymbol{j}_\kappa & \boldsymbol{k}_\kappa \end{pmatrix} = \boldsymbol{\Pi}_\kappa \boldsymbol{R}_\kappa, \tag{19}$$

where $\boldsymbol{R}_\kappa$ is the rotation at time $\kappa$. If we let $\boldsymbol{m}^\dagger_{\kappa(a)}$ be the $(2(\kappa-1)+a)$th column of the transpose $\boldsymbol{M}^\top$ of the matrix $\boldsymbol{M}$ in eqs. (17), $\kappa = 1, ..., M$, $a = 1, 2$. The transpose of both sides of eq. (19) is

$$\boldsymbol{R}_\kappa^\top \boldsymbol{\Pi}_\kappa^\top = \begin{pmatrix} \boldsymbol{m}^\dagger_{\kappa(1)} & \boldsymbol{m}^\dagger_{\kappa(2)} \end{pmatrix}. \tag{20}$$

Eq. (18) implies $\boldsymbol{M}^\top = \boldsymbol{A}^\top \boldsymbol{U}^\top$, so if we let $\boldsymbol{u}^\dagger_{\kappa(a)}$ be the $(2(\kappa-1)+a)$th column of the transpose $\boldsymbol{U}^\top$ of the matrix $\boldsymbol{U}$ in eqs. (17), we obtain

$$\boldsymbol{m}^\dagger_{\kappa(a)} = \boldsymbol{A}^\top \boldsymbol{u}^\dagger_{\kappa(a)}. \tag{21}$$

Substituting this, we can rewrite eq. (20) as

$$\boldsymbol{R}_\kappa^\top \boldsymbol{\Pi}_\kappa^\top = \boldsymbol{A}^\top \begin{pmatrix} \boldsymbol{u}^\dagger_{\kappa(1)} & \boldsymbol{u}^\dagger_{\kappa(2)} \end{pmatrix}. \tag{22}$$

Let $\boldsymbol{U}^\dagger_\kappa$ the $3 \times 2$ matrix having $\boldsymbol{u}^\dagger_{\kappa(1)}$ and $\boldsymbol{u}^\dagger_{\kappa(2)}$ as columns:

$$\boldsymbol{U}^\dagger_\kappa = \begin{pmatrix} \boldsymbol{u}^\dagger_{\kappa(1)} & \boldsymbol{u}^\dagger_{\kappa(2)} \end{pmatrix}. \tag{23}$$

Then, eq. (22) can be rewritten as $\boldsymbol{U}^{\dagger\top}_\kappa \boldsymbol{A}\boldsymbol{A}^\top \boldsymbol{U}^\dagger_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{R}_\kappa \boldsymbol{R}_\kappa^\top \boldsymbol{\Pi}_\kappa^\top$. Since $\boldsymbol{R}_\kappa$ is a rotation matrix, we have the generic *metric constraint*

$$\boldsymbol{U}^{\dagger\top}_\kappa \boldsymbol{T} \boldsymbol{U}^\dagger_\kappa = \boldsymbol{\Pi}_\kappa \boldsymbol{\Pi}_\kappa^\top, \tag{24}$$

where we define the *metric matrix* $\boldsymbol{T}$ by

$$\boldsymbol{T} = \boldsymbol{A}\boldsymbol{A}^\top. \tag{25}$$

Eq. (24) is the generic metric constraint given by Quan [13]. If we take out the elements on both sides, we have the following three expressions:

$$(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) = \sum_{i=1}^3 \Pi^2_{1i\kappa},$$
$$(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = \sum_{i=1}^3 \Pi^2_{2i\kappa},$$
$$(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = \sum_{i=1}^3 \Pi_{1i\kappa}\Pi_{2i\kappa}. \tag{26}$$

If we let, instead of eq. (16), simply $\boldsymbol{m}_i = \boldsymbol{u}_i$, $i = 1, 2, 3$, we can still reconstruct the 3-D shape, but it is a deformation of the true shape by some affine transformation, known as *affine reconstruction*[6]. In order to restore the true shape (*Euclidean reconstruction*), one needs to rectify the basis $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$ of the subspace $\mathcal{L}$ by some linear transformation $\boldsymbol{A}$, and eq. (24) gives the constraint on it. In this sense, eq. (24) corresponds to the *dual absolute quadric constraint* [3] on the homography that rectifies the projective basis of *projective reconstruction* to Euclidean.

Assuming that the three intrinsic parameters are the same throughout the input sequence, Quan [13] eliminated them from eqs. (26) and obtained nonlinear constraints on the metric matrix $\boldsymbol{T}$, which he solved by nonlinear optimization. Here, we focus on the fact that *at most two* time varying unknowns of the camera model can be eliminated from eqs. (26). We now show that (i) we can restrict the camera model without much impairing its descriptive capability so that it has *two* free functions and (ii) we can redefine them in such a way that the resulting $2M$ unknowns are *linearly* estimated.

## 4. Symmetric Affine Cameras

We now seek a concrete form of the affine camera by imposing minimal requirements that eq. (2) mimic perspective projection.

**Requirement 1.** The frontal parallel plane passing through the world coordinate origin is projected as if by perspective projection.

This corresponds to our assumption that the object of our interest is small and localized around the world coordinate origin $(t_x, t_y, t_z)$. A point on the plane $Z = t_z$ is written as $(X, Y, t_z)$, so Requirement 1 requires

$$\begin{pmatrix} fX/t_z \\ fY/t_z \end{pmatrix} = \begin{pmatrix} \Pi_{11} & \Pi_{12} \\ \Pi_{21} & \Pi_{22} \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} + t_z \begin{pmatrix} \Pi_{13} \\ \Pi_{23} \end{pmatrix} + \begin{pmatrix} \pi_1 \\ \pi_2 \end{pmatrix}. \tag{27}$$

[5]In the traditional formulation [11, 12, 20, 21], the observation matrix $\boldsymbol{W}$ is decomposed by SVD (singular value decomposition) into $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top$, and the motion and the shape matrices $\boldsymbol{M}$ and $\boldsymbol{S}$ are set to $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{A}$ an $\boldsymbol{S} = \boldsymbol{A}^{-1}\boldsymbol{\Lambda}\boldsymbol{V}^\top$ via a nonsingular matrix $\boldsymbol{A}$. However, our formulation is better suited for the subsequent analysis.

[6]We are assuming an affine camera model. If we use perspective images, the resulting shape may not be affine reconstruction, of course.

Since this should hold for arbitrary $X$ and $Y$, we obtain

$$\Pi_{11} = \Pi_{22} = \frac{f}{t_z}, \qquad \Pi_{12} = \Pi_{21} = 0,$$

$$t_z\Pi_{13} + \pi_1 = 0, \qquad t_z\Pi_{23} + \pi_2 = 0, \tag{28}$$

which reduces eq. (2) to

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{t_z}\begin{pmatrix} X \\ Y \end{pmatrix} - (t_z - Z)\begin{pmatrix} \Pi_{13} \\ \Pi_{23} \end{pmatrix}, \tag{29}$$

where $f$, $\Pi_{13}$ and $\Pi_{23}$ are arbitrary functions of $\{\boldsymbol{t}, \boldsymbol{R}\}$. In order to obtain a more specific form, we impose the following requirements:

**Requirement 2.** The camera imaging is symmetric around the $Z$-axis.

**Requirement 3.** The camera imaging does not depend on $\boldsymbol{R}$.

Requirement 2 states that if the scene is rotated around the optical axis by an angle $\theta$, the resulting image should also rotate around the image origin by the same angle $\theta$, a very natural requirement. Requirement 3 is also natural, since the orientation of the world coordinate system can be defined arbitrarily, and such indeterminate parameterization should not affect the actual observation.

Let $\mathcal{R}(\theta)$ be the 2-D rotation matrix by angle $\theta$:

$$\mathcal{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{30}$$

Requirement 2 is written as

$$\mathcal{R}(\theta)\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{t_z}\mathcal{R}(\theta)\begin{pmatrix} X \\ Y \end{pmatrix} - (t_z - Z)\begin{pmatrix} \Pi'_{13} \\ \Pi'_{23} \end{pmatrix}, \tag{31}$$

where $\Pi'_{13}$ and $\Pi'_{23}$ are the values of the functions $\Pi_{13}$ and $\Pi_{23}$, respectively, obtained by replacing $t_x$ and $t_y$ in their arguments by $t_x\cos\theta - t_y\sin\theta$ and $t_x\sin\theta + t_y\cos\theta$, respectively; by Requirement 3, the arguments of $\Pi_{13}$ and $\Pi_{23}$ do not contain $\boldsymbol{R}$. Multiplying both sides of eq. (29) by $\mathcal{R}(\theta)$, we obtain

$$\mathcal{R}(\theta)\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{t_z}\mathcal{R}(\theta)\begin{pmatrix} X \\ Y \end{pmatrix} - (t_z - Z)\mathcal{R}(\theta)\begin{pmatrix} \Pi_{13} \\ \Pi_{23} \end{pmatrix}. \tag{32}$$

Comparing eqs. (31) and (32), we conclude that the equality

$$\begin{pmatrix} \Pi'_{13} \\ \Pi'_{23} \end{pmatrix} = \mathcal{R}(\theta)\begin{pmatrix} \Pi_{13} \\ \Pi_{23} \end{pmatrix} \tag{33}$$

should hold identically for an arbitrary $\theta$. According to the theory of invariants [4], this implies

$$\begin{pmatrix} \Pi_{13} \\ \Pi_{23} \end{pmatrix} = c\begin{pmatrix} t_x \\ t_y \end{pmatrix}, \tag{34}$$

where $c$ is an arbitrary function of $t_x^2 + t_y^2$ and $t_z$. Thus, if we define

$$\zeta = \frac{t_z}{f}, \qquad \beta = -\frac{ct_z}{f}, \tag{35}$$

eq. (29) is written as

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{\zeta}\left(\begin{pmatrix} X \\ Y \end{pmatrix} + \beta(t_z - Z)\begin{pmatrix} t_x \\ t_y \end{pmatrix}\right). \tag{36}$$

The corresponding projection matrix $\boldsymbol{\Pi}$ and the projection vector $\boldsymbol{\pi}$ are

$$\boldsymbol{\Pi} = \begin{pmatrix} 1/\zeta & 0 & -\beta t_x/\zeta \\ 0 & 1/\zeta & -\beta t_y/\zeta \end{pmatrix}, \quad \boldsymbol{\pi} = \begin{pmatrix} \beta t_x t_z/\zeta \\ \beta t_y t_z/\zeta \end{pmatrix}, \tag{37}$$

where $\zeta$ and $\beta$ are arbitrary *functions* of $t_x^2 + t_y^2$ and $t_z$. We observe:

- Eq. (36) reduces to the paraperspective projection of eq. (5) if we choose
$$\zeta = \frac{t_z}{f}, \qquad \beta = \frac{1}{t_z}. \tag{38}$$
- Eq. (36) reduces to the weak perspective projection of eq. (4) if we choose
$$\zeta = \frac{t_z}{f}, \qquad \beta = 0. \tag{39}$$
- Eq. (36) reduces to the orthographic projection of eq. (3) if we choose
$$\zeta = 1, \qquad \beta = 0. \tag{40}$$

Thus, eq. (36) includes the traditional affine camera models as special instances and is the *only possible form* that satisfies Requirements 1, 2, and 3.

However, we need *not* define the functions $\zeta$ and $\beta$ in any particular form; we can regard them as *time varying unknowns* and determine their values by *self-calibration*. This is made possible by the fact that *at most two* time varying unknowns can be eliminated from the metric constraint of eqs. (26).

## 5. Procedure for 3-D Reconstruction

3-D Euclidean reconstruction using eq. (36) goes just as when using the traditional camera models [10]. Here is the outline (the full computational details are given in Appendix):

1. We fit a 3-D affine space $\mathcal{A}$ to the trajectories $\{\boldsymbol{p}_\alpha\}$ by least squares. Namely, we compute the centroid $\boldsymbol{m}_0$ by eq. (13) and compute the unit eigenvectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$ of the moment matrix $\boldsymbol{C}$ in eq. (15) for the largest three eigenvalues[7].

[7] This corresponds to the SVD $\boldsymbol{W} = \boldsymbol{U\Lambda V}^\top$ of the observation matrix $\boldsymbol{W}$ in the traditional formulation [12, 20].

2. We eliminate time varying unknowns from the the metric constraint of eqs. (26) and solve for the metric matrix $\boldsymbol{T}$ by least squares. To be specific, substituting eqs. (37) into eqs. (26), we have

$$
\begin{aligned}
(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) &= \frac{1}{\zeta_\kappa^2} + \beta_\kappa^2 \tilde{t}_{x\kappa}^2, \\
(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) &= \frac{1}{\zeta_\kappa^2} + \beta_\kappa^2 \tilde{t}_{y\kappa}^2, \\
(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) &= \beta_\kappa^2 \tilde{t}_{x\kappa}\tilde{t}_{y\kappa},
\end{aligned} \tag{41}
$$

where $\tilde{t}_{x\kappa}$ and $\tilde{t}_{y\kappa}$ are, respectively, the $(2(\kappa - 1) + 1)$th and the $(2(\kappa - 1) + 2)$th components of the centroid $\boldsymbol{m}_0$. Eliminating $\zeta_\kappa$ and $\beta_\kappa$, we obtain

$$
\begin{aligned}
A_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) - C_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \\
- A_\kappa(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = 0,
\end{aligned} \tag{42}
$$

where $A_\kappa = \tilde{t}_{x\kappa}\tilde{t}_{y\kappa}$ and $C_\kappa = \tilde{t}_{x\kappa}^2 - \tilde{t}_{y\kappa}^2$. This is a linear constraint on $\boldsymbol{T}$, so we can determine $\boldsymbol{T}$ by solving the $M$ equations for $\kappa = 1, ..., M$ by least squares. Once we have determined $\boldsymbol{T}$, we can determine $\zeta_\kappa$ and $\beta_\kappa$ from eqs. (41) by least squares.

3. We decompose the metric matrix $\boldsymbol{T}$ into the rectifying matrix $\boldsymbol{A}$ in the form of eq. (25), and compute the vectors $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, and $\boldsymbol{m}_3$ from eq. (16).

4. We compute the translation $\boldsymbol{t}_\kappa$ and the rotation $\boldsymbol{R}_\kappa$ at each time. The translation components $t_{x\kappa}$ and $t_{y\kappa}$ are given by eq. (8) in the form of $t_{x\kappa} = \zeta_\kappa \tilde{t}_{x\kappa}$ and $t_{y\kappa} = \zeta_\kappa \tilde{t}_{y\kappa}$. The three rows $\boldsymbol{r}_{\kappa(1)}$, $\boldsymbol{r}_{\kappa(2)}$, and $\boldsymbol{r}_{\kappa(3)}$ of the rotation $\boldsymbol{R}_\kappa$ are given by solving the linear equations

$$
\begin{aligned}
\boldsymbol{r}_{\kappa(1)} - \beta_\kappa t_{x\kappa}\boldsymbol{r}_{\kappa(3)} &= \zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(1)}, \\
\boldsymbol{r}_{\kappa(2)} - \beta_\kappa t_{y\kappa}\boldsymbol{r}_{\kappa(3)} &= \zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(2)}, \\
\beta_\kappa t_{x\kappa}\boldsymbol{r}_{\kappa(1)} + \beta_\kappa t_{y\kappa}\boldsymbol{r}_{\kappa(2)} + \boldsymbol{r}_{\kappa(3)} &= \zeta_\kappa^2 \boldsymbol{m}^\dagger_{\kappa(1)} \times \boldsymbol{m}^\dagger_{\kappa(2)}.
\end{aligned} \tag{43}
$$

The resulting matrix $\boldsymbol{R}_\kappa$ may not be strictly orthogonal, so we compute its SVD (singular value decomposition) $\boldsymbol{V}_\kappa \boldsymbol{\Lambda}_\kappa \boldsymbol{U}_\kappa^\top$ and redefine $\boldsymbol{V}_\kappa \boldsymbol{U}_\kappa^\top$ to be $\boldsymbol{R}_\kappa$ [5].

5. We recompute the vectors $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, and $\boldsymbol{m}_3$ in the form of eqs. (12) using the computed rotations $\boldsymbol{R}_\kappa = \begin{pmatrix} \boldsymbol{i}_\kappa & \boldsymbol{j}_\kappa & \boldsymbol{k}_\kappa \end{pmatrix}$.

6. We compute the *shape vector* $\boldsymbol{s}_\alpha = (a_\alpha, b_\beta, c_\beta)^\top$ of each point by least-squares expansion of $\boldsymbol{p}'_\alpha$ in the form of eq. (14), minimizing

$$
\|\boldsymbol{p}'_\alpha - a_\alpha \boldsymbol{m}_1 - b_\alpha \boldsymbol{m}_2 - c_\alpha \boldsymbol{m}_3\|^2 = \|\boldsymbol{p}'_\alpha - \boldsymbol{M}\boldsymbol{s}_\alpha\|^2. \tag{44}
$$

The solution is given by $\boldsymbol{s}_\alpha = \boldsymbol{M}^- \boldsymbol{p}_\alpha$, using the pseudoinverse $\boldsymbol{M}^-$ of $\boldsymbol{M}$.

However, the following indeterminacy remains:

1. Another solution is obtained by multiplying all $\{\boldsymbol{t}_\kappa\}$ and $\{\boldsymbol{s}_\alpha\}$ by a common constant.

2. Another solution is obtained by multiplying the all $\{\boldsymbol{R}_\kappa\}$ by a common rotation. The shape vectors $\{\boldsymbol{s}_\alpha\}$ are rotated accordingly.

3. Each solution has its mirror image solution. The mirror image rotation $\boldsymbol{R}'_\kappa$ is obtained by the rotation $\boldsymbol{R}_\kappa$ followed by a rotation around axis $(\beta_\kappa t_{x\kappa}, \beta_\kappa t_{y\kappa}, 1)$ by angle $2\pi$. Then, the shape vectors $\{\boldsymbol{s}_\alpha\}$ change their signs.

4. *The absolute depth $t_z$ of the world coordinate origin is indeterminate.*

Item 1 is the fundamental ambiguity of 3-D reconstruction from images, meaning that a large motion of a large object in the distance is indistinguishable from a small motion of a small object nearby. Item 2 reflects the fact that the orientation of the world coordinate system can be arbitrarily chosen. Item 3 is due to eq. (25), which can be written as $\boldsymbol{T} = (\pm\boldsymbol{AQ})(\pm\boldsymbol{AQ})^\top$ for an arbitrary rotation $\boldsymbol{Q}$, and is inherent of all affine cameras [13, 14].

Item 4 is due to the fact that eq. (36) involves only the *relative depth* of individual point from the world coordinate origin $\boldsymbol{t}_\kappa$. The absolute depth $t_z$ is determined only if $\zeta$ and $\beta$ are given as *specific functions of* $t_z$, as in the case of the traditional camera models. Here, however, we do not specify their functional forms, directly determining their values by self-calibration and leaving $t_z$ unspecified.

## 6. Experiments

Fig. 3 shows four simulated image sequences of $600 \times 600$ pixels perspectively projected with focal length $f = 600$ pixels. Each consists of 11 frames; six decimated frames are shown here. We added Gaussian random noise of mean 0 and standard deviation 1 pixel independently to the $x$ and $y$ coordinates of the feature points and reconstructed their 3-D shape (the frames in Fig. 3(a),(b) are merely for visual ease).

From the resulting two mirror image shapes, we chose the correct one by comparing the depths of two points that are known be close to and away from the camera. Since the absolute depth and scale are indeterminate, we translated the true and the reconstructed shapes so that their centroids are at the coordinate origin and scaled their sizes so that the root-mean-square distance of the feature points from the origin is 1. Then, we rotated the reconstructed shape so that root-mean-square distances between the corresponding points of the two shapes is minimized. We adopted the resulting residual as the measure of reconstruction accuracy.

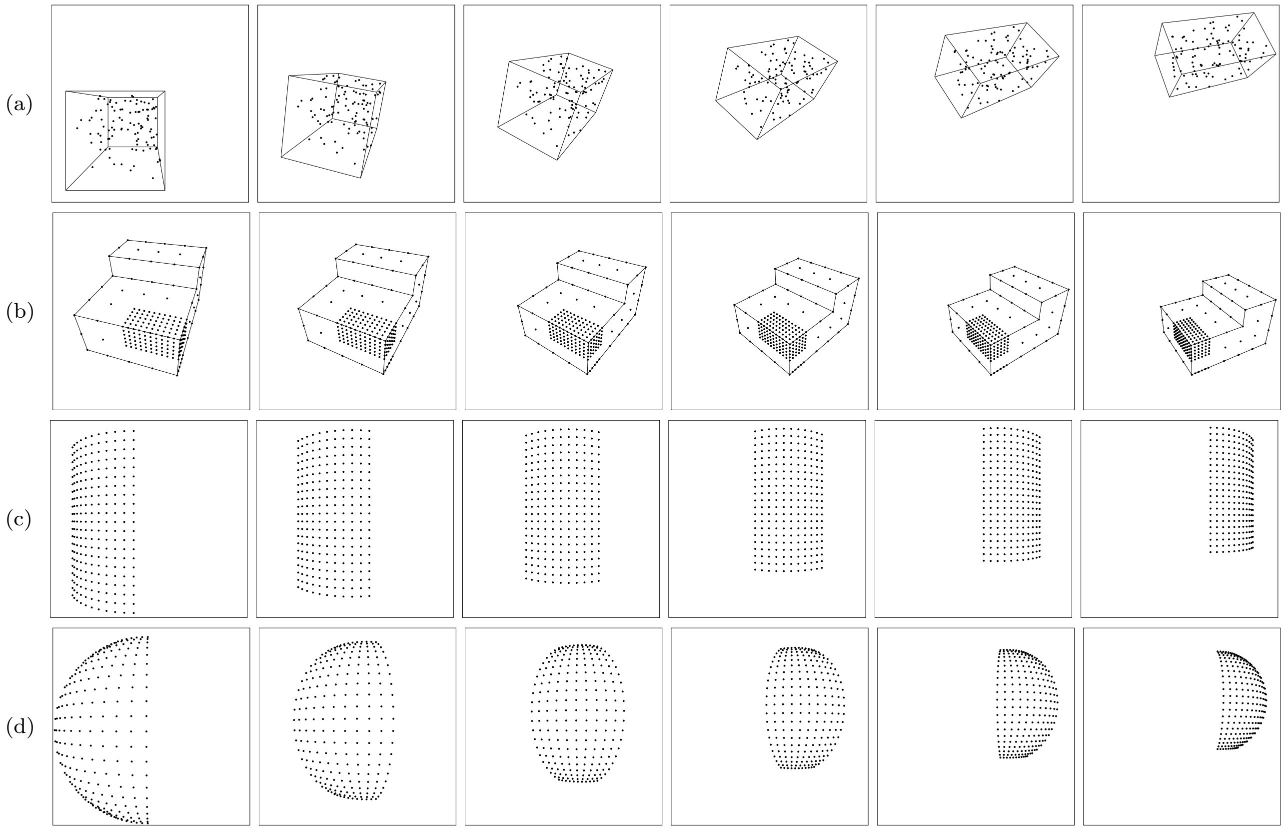
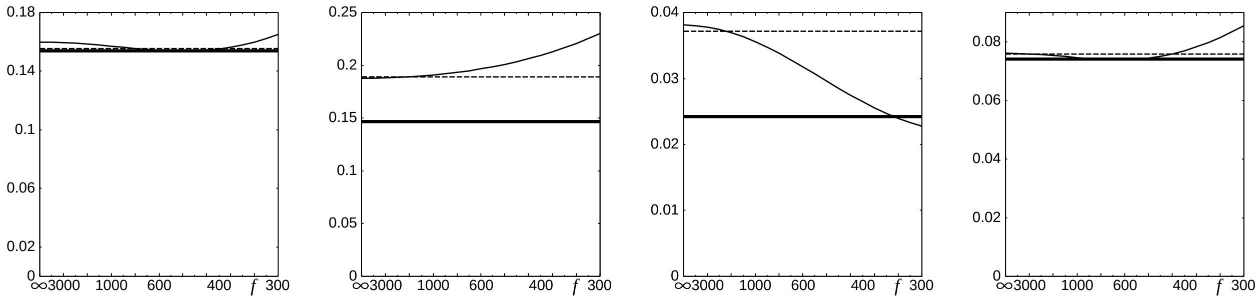



Figure 3: Simulated image sequences (six decimated frames for each).




(a) (b) (c) (d)

Figure 4: 3-D reconstruction accuracy for the image sequences of Fig. 3(a)∼(d). The horizontal axis is scaled in proportion to $1/f$. Three models are compared: The dashed line: weak perspective (dashed lines), paraperspective (thin solid lines), and our generic model (thick solid lines).

We compared three camera models: the weak perspective, the paraperspective, and our symmetric affine camera models. The orthographic model is omitted, since evidently good results cannot be obtained when the object moves in the depth direction. For using the weak perspective and paraperspective models, we need to specify the focal length $f$ (see eqs. (4) and (5)). If the size of the reconstructed shape is normalized as described earlier, the choice of $f$ is irrelevant for the weak perspective model, because it only affects the object size as a whole. However, the paraperspective model depends on the value of $f$ we use.

Fig. 4 plots the reconstruction accuracy vs. the input focal length $f$; the horizontal axis is scaled in proportion to $1/f$. The dashed line is for weak perspective, the thin solid line is for paraperspective, and the thick solid line is for our model. We observe that the paraperspective model does not necessarily give the highest accuracy when $f$ coincides with the focal length (600 pixels) of the perspective images. The error is indeed minimum around $f = 600$ for Fig. 4(b),(c), but the error decreases as $f$ increases for Fig. 4(a) and as $f$ decreases for Fig. 4(d).

We conclude that our model achieves the accuracy comparable to paraperspective projection given an appropriate value of $f$, which is unknown in advance. This means that our model automatically chooses appropriate parameter values without any knowledge about $f$.

We conducted many other experiments (not shown here) and observed similar results. We have found that *degeneracy* can occur in special circumstances. By "degeneracy", we mean that the matrix $\boldsymbol{A}$ is rank deficient so that the resulting vectors $\{\boldsymbol{m}_i\}$ are linearly dependent (see eq. (16)). As a result, the reconstructed shape is "flat" (see eq. (14)). This occurs when the smallest eigenvalue of $\boldsymbol{T}$ computed by least squares is negative, while eq. (25) requires $\boldsymbol{T}$ to be positive semidefinite. In the computation, we replace the negative eigenvalue by zero, resulting in degeneracy.

This type of degeneracy occurs for the traditional camera models, too. In principle, we could avoid it by parameterizing $\boldsymbol{T}$ so that it is guaranteed to be positive definite [13]. However, this would require nonlinear optimization, and the merit of the factorization approach (i.e., linear computation only) would be lost. Moreover, if we look at the images that cause degeneracy, they really look as if a planar object is moving. Since the information is insufficient in the first place, any methods, including self-calibration using the dual absolute quadric constraint, which is very susceptible to noise, may not be able to solve such degeneracies.

## 7. Conclusions

We showed that minimal requirements for an affine camera to mimic perspective projection leads to a unique camera model, which we call "symmetric affine camera", having two free functions, whose specific choices would result in the traditional camera models. We regarded them as time varying parameters and determined their values by self-calibration, using linear computation alone, so that an appropriate model is automatically selected. Our method can be viewed as an extension of Quan's method [13] to varying intrinsic parameters. We have demonstrated by simulation that the reconstruction accuracy is comparable to the paraperspective model given an appropriate focal length estimate.

The full computational details are given in Appendix.

**Acknowledgments:** The authors thank Seitaro Asahara of Recruit Staffing, Co., Ltd., Japan, for participating in this project. This work was supported in part by the Ministry of Education, Culture, Sports, Science and Technology, Japan, under a Grant in Aid for Scientific Research C (No. 17500112).

## References

[1] R. Basri, Paraperspective ≡ affine, *Int. J. Comput. Vision*, **19**-2 (1996-8), 169–179.

[2] K. Deguchi, T. Sasano, H. Arai, and H. Yoshikawa, 3-D shape reconstruction from endoscope image sequences by the factorization method, *IEICE Trans. Inf. & Syst.*, *E79-D*-9 (1996-9), 1329–1336.

[3] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, U.K., 2000.

[4] K. Kanatani, *Group-Theoretical Methods in Image Understanding*, Springer-Verlag, Berlin, Germany, 1990.

[5] K. Kanatani, *Geometric Computation for Machine Vision*, Oxford University Press, Oxford, U.K., 1993.

[6] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier Science, Amsterdam, The Netherlands, 1996,

[7] K. Kanatani, Motion segmentation by subspace separation and model selection, *Proc. 8th Int. Conf. Comput. Vision* July 2001, Vancouver, Canada, Vol. 2, pp. 301–306.

[8] K. Kanatani, Motion segmentation by subspace separation: Model selection and reliability evaluation, *Int. J. Image Graphics*, *2*-2 (2002-4), 179–197.

[9] K. Kanatani, Evaluation and selection of models for motion segmentation, *Proc. 7th Euro. Conf. Comput. Vision*, May 2002, Copenhagen, Denmark, Vol. 3, pp. 335–349.

[10] K. Kanatani and Y. Sugaya Factorization without factorization: Complete Recipe, *Mem. Fac. Eng. Okayama Univ.*, **38**-1/2 (2004-3), 61–72.

[11] T. Morita and T. Kanade, A sequential factorization method for recovering shape and motion from image sequence, *IEEE Trans. Patt. Anal. Mach. Intell.*, *19*-8 (1997-8), 858–867.

[12] C. J. Poelman and T. Kanade, A paraperspective factorization method for shape and motion recovery, *IEEE Trans. Patt. Anal. Mach. Intell.*, *19*-3 (1997-3), 206–218.

[13] L. Quan, Self-calibration of an affine camera from multiple views, *Int. J. Comput. Vision*, **19**-1 (1996-7), 93–105.

[14] L. S. Shapiro, A. Zisserman, and M. Brady, 3D motion recovery via affine epipolar geometry, *Int. J. Comput. Vision*, **16**-2 (1995-10), 147–182.

[15] Y. Sugaya and K. Kanatani, Outlier removal for motion tracking by subspace separation, *IEICE Trans. Inf. & Syst.*, *E86-D*-6 (2003-6), 1095–1102.

[16] Y. Sugaya and K. Kanatani, Extending interrupted feature point tracking for 3-D affine reconstruction, *IEICE Trans. Inf. & Syst.*, *E87-D*-4 (2004-4), 1031–1038.

[17] Y. Sugaya and K. Kanatani, Multi-stage optimization for multi-body motion segmentation, *IEICE Trans. Inf. & Syst.*, *E87-D*-7 (2004-7), 1935–1942.

[18] Y. Sugaya and K. Kanatani, Extracting moving objects from a moving camera video sequence, *Mem. Fac. Eng. Okayama Univ. 39* (2005-1), 56–62.

[19] A. Sugimoto, Object recognition by combining paraperspective images, *Int. J. Comput. Vision*, **19**-2 (1996-8), 181–201.

[20] C. Tomasi and T. Kanade, Shape and motion from image streams under orthography—A factorization method, *Int. J. Comput. Vision*, *9*-2 (1992-10), 137–154.

[21] D. Weinshall and C. Tomasi, Linear and incremental acquisition of invariant shape models from image sequences, *IEEE Trans. Patt. Anal. Mach. Intell.*, *17*-5 (1995-5), 512–517.

# Appendix

## A. Procedure for 3-D Reconstruction

**Input:** Image coordinates $(x_{\kappa\alpha}, y_{\kappa\alpha})$ of the $\alpha$th feature point in the $\kappa$th frame, $\kappa = 1, ..., M$, $\alpha = 1, ..., N$. The origin of the image coordinate system is assumed to be at the center of the frame with the $x$-axis upward and the $y$-axis rightward.

**Output:** Two sets of reconstructed 3-D positions $\{\boldsymbol{r}_\alpha\}$ and $\{\boldsymbol{r}'_\alpha\}$, each a mirror image of the other.

**Procedure:**

1. Compute the centroid $\boldsymbol{m}_0$ of the $2M$-dimensional trajectory vectors $\{\boldsymbol{p}_\alpha\}$ (eq. (12)) and the $2M \times 2M$ moment matrix $\boldsymbol{C}$ (eq. (14)).
2. Let $\tilde{t}_{x\kappa}$ and $\tilde{t}_{y\kappa}$ be the $(2(\kappa-1)+1)$th and the $(2(\kappa-1)+2)$th components of the centroid $\boldsymbol{m}_0$, respectively.
3. Let $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, and $\boldsymbol{u}_3$ be the $2M$-dimensional unit eigenvectors of the moment matrix $\boldsymbol{C}$ for the largest three eigenvalues, and define the following $2M \times 3$ matrix $\boldsymbol{U}$:
$$\boldsymbol{U} = \begin{pmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \boldsymbol{u}_3 \end{pmatrix}. \tag{45}$$
4. Let $\boldsymbol{u}^\dagger_{\kappa(a)}$ be the $(2(\kappa-1)+a)$th column of the transpose $\boldsymbol{U}^\top$, $\kappa = 1, ..., M$, $a = 1, 2$.
5. Compute the $3 \times 3$ metric matrix $\boldsymbol{T}$ (see Appendix B).
6. Compute the parameters $\{\zeta_\kappa\}$ and $\{\beta_\kappa\}$ and the translation components $\{(t_{x\kappa}, t_{y\kappa})\}$ (see Appendix C).
7. Compute the rotations $\{\boldsymbol{R}_\kappa\}$ (see Appendix D).
8. Compute the following $2M \times 3$ matrix $\boldsymbol{M}$:
$$\boldsymbol{M} = \sum_{\kappa=1}^{M} \tilde{\boldsymbol{\Pi}}_\kappa^\top \boldsymbol{R}_\kappa. \tag{46}$$
Here, $\tilde{\boldsymbol{\Pi}}_\kappa = (\tilde{\Pi}_{\kappa(ij)})$ is a $3 \times 2M$ matrix with elements
$$\begin{array}{cl} 1/\zeta_\kappa & \text{if } (i,j) = (1, 2\kappa-1), (2, 2\kappa), \\ -\beta_\kappa t_{x\kappa}/\zeta_\kappa & \text{if } (i,j) = (3, 2\kappa-1), \\ -\beta_\kappa t_{y\kappa}/\zeta_\kappa & \text{if } (i,j) = (3, 2\kappa), \\ 0 & \text{otherwise.} \end{array} \tag{47}$$
9. If the rank of $\boldsymbol{M}$ is 2 or less, exit and switch to the usual factorization procedure based on the weak perspective projection model.
10. Else, compute the shape vectors $\{\boldsymbol{s}_\alpha\}$, $\alpha = 1, ..., N$, as follows:
$$\boldsymbol{s}_\alpha = (\boldsymbol{M}^\top \boldsymbol{M})^{-1} \boldsymbol{M}^\top \boldsymbol{p}'_\alpha, \tag{48}$$
11. Compute the mirror image rotations $\{\boldsymbol{R}'_\kappa\}$, $\kappa = 1, ..., M$, as follows (see Appendix E):
$$\boldsymbol{n}_\kappa = N[\begin{pmatrix} \beta_\kappa t_{x\kappa} \\ \beta_\kappa t_{y\kappa} \\ 1 \end{pmatrix}], \qquad \boldsymbol{\Omega}_\kappa = 2\boldsymbol{n}_\kappa \boldsymbol{n}_\kappa^\top - \boldsymbol{I},$$
$$\boldsymbol{R}'_\kappa = \boldsymbol{\Omega}_\kappa \boldsymbol{R}_\kappa. \tag{49}$$
12. Output the two sets of 3-D positions $\{\boldsymbol{r}_{\kappa\alpha}\}$ and $\{\boldsymbol{r}'_{\kappa\alpha}\}$, $\alpha = 1, ..., N$, $\kappa = 1, ..., M$, given by
$$\boldsymbol{r}_{\kappa\alpha} = \begin{pmatrix} t_{x\kappa} \\ t_{y\kappa} \\ t_{z\kappa} \end{pmatrix} + \boldsymbol{R}_\kappa \boldsymbol{s}_\alpha,$$
$$\boldsymbol{r}'_{\kappa\alpha} = \begin{pmatrix} t_{x\kappa} \\ t_{y\kappa} \\ t_{z\kappa} \end{pmatrix} - \boldsymbol{R}'_\kappa \boldsymbol{s}_\alpha, \tag{50}$$
where $t_{z\kappa}$ is arbitrarily set (e.g., $t_{z\kappa} = 0$).

## B. Computation of the Metric Constraint

Letting the projection matrix elements $\Pi_{ij\kappa}$ in the metric constraint of eqs. (25) be in the form of the first of eqs. (36), we obtain

$$(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) = \frac{1}{\zeta_\kappa^2} + \beta_\kappa^2 \tilde{t}_{x\kappa}^2,$$
$$(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = \frac{1}{\zeta_\kappa^2} + \beta_\kappa^2 \tilde{t}_{y\kappa}^2,$$
$$(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = \beta_\kappa^2 \tilde{t}_{x\kappa}\tilde{t}_{y\kappa}. \tag{51}$$

The third equation can be solved for $\beta_\kappa$ in the form

$$\beta_\kappa = \sqrt{\frac{(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)})}{\tilde{t}_{x\kappa}\tilde{t}_{y\kappa}}}. \tag{52}$$

Substituting this into the first and the second of eqs. (51), we obtain

$$(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) = \frac{1}{\zeta_\kappa^2} + \frac{\tilde{t}_{x\kappa}}{\tilde{t}_{y\kappa}}(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}),$$
$$(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = \frac{1}{\zeta_\kappa^2} + \frac{\tilde{t}_{y\kappa}}{\tilde{t}_{x\kappa}}(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}). \tag{53}$$

Eliminating $1/\zeta_\kappa^2$ by subtraction on both sides, we obtain after rearrangement

$$A_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) - C_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)})$$
$$-A_\kappa(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) = 0, \tag{54}$$

where we put

$$A_\kappa = \tilde{t}_{x\kappa}\tilde{t}_{y\kappa}, \qquad C_\kappa = \tilde{t}_{x\kappa}^2 - \tilde{t}_{y\kappa}^2. \tag{55}$$

We determine the metric matrix $\boldsymbol{T}$ by least squares, minimizing

$$K = \sum_{\kappa=1}^{M}\Big(A_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) - C_\kappa(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) - A_\kappa(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)})\Big)^2, \tag{56}$$

which can be rewritten as

$$K = \sum_{\kappa=1}^{M}\sum_{i,j,k,l=1}^{3} B_{ijkl}T_{ij}T_{kl}, \tag{57}$$

where we define the $3\times3\times3\times3$ tensor $\mathcal{B} = (B_{ijkl})$ as follows:

$$\begin{aligned}
B_{ijkl} = \sum_{\kappa=1}^{M}\Big[ & A_\kappa^2\Big((\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l\Big) \\
& +\frac{1}{4}C_\kappa^2\Big((\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j \\
& (\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l + (\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l\Big) \\
& -\frac{1}{2}A_\kappa C_\kappa\Big((\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l \\
& +(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(1)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(1)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(1)})_k(\boldsymbol{u}^\dagger_{\kappa(2)})_l \\
& -(\boldsymbol{u}^\dagger_{\kappa(2)})_i(\boldsymbol{u}^\dagger_{\kappa(2)})_j(\boldsymbol{u}^\dagger_{\kappa(2)})_k(\boldsymbol{u}^\dagger_{\kappa(1)})_l\Big)\Big].
\end{aligned} \tag{58}$$

The function $K$ appears to take its minimum $K = 0$ for $\boldsymbol{T} = \boldsymbol{O}$, but we must recall the scale indeterminacy of $\boldsymbol{T}$. Doubling $\boldsymbol{T}$ in eq. (22) means multiplying the rectifying matrix $\boldsymbol{A}$ by $\sqrt{2}$. Hence, the vector $\boldsymbol{m}_i$ in eq. (15) is also multiplied by $\sqrt{2}$. However, we can still obtain a solution compatible with the observed data $\{\boldsymbol{p}_\alpha\}$ if we divide $a_\alpha$, $b_\alpha$, and $c_\alpha$ in eq. (13) by $\sqrt{2}$. Hence, we do not lose generality if we impose normalization $\|\boldsymbol{T}\| = 1$, where the matrix norm is defined by $\|\boldsymbol{T}\| = \sqrt{\sum_{i,j=1,3} T_{ij}^2}$. Then, the solution $\boldsymbol{T}$ that minimizes eq. (57) is given by the eigenmatrix of unit norm of tensor $\mathcal{B}$ for the smallest eigenvalue [6]. This is obtained by first computing the 6-D unit eigenvector $\boldsymbol{\tau} = (\tau_i)$ of the following $6\times6$ matrix $\boldsymbol{B}$ for the smallest eigenvalue [6]:

$$\boldsymbol{B} = \begin{pmatrix}
B_{1111} & B_{1122} & B_{1133} & \sqrt{2}B_{1123} & \sqrt{2}B_{1131} & \sqrt{2}B_{1112} \\
B_{2211} & B_{2222} & B_{2233} & \sqrt{2}B_{2223} & \sqrt{2}B_{2231} & \sqrt{2}B_{2212} \\
B_{3311} & B_{3322} & B_{3333} & \sqrt{2}B_{3323} & \sqrt{2}B_{3331} & \sqrt{2}B_{3312} \\
\sqrt{2}B_{2311} & \sqrt{2}B_{2322} & \sqrt{2}B_{2333} & 2B_{2323} & 2B_{2331} & 2B_{2312} \\
\sqrt{2}B_{3111} & \sqrt{2}B_{3122} & \sqrt{2}B_{3133} & 2B_{3123} & 2B_{3131} & 2B_{3112} \\
\sqrt{2}B_{1211} & \sqrt{2}B_{1222} & \sqrt{2}B_{1233} & 2B_{1223} & 2B_{1231} & 2B_{1212}
\end{pmatrix}. \tag{59}$$

The eigenvector $\boldsymbol{\tau}$ is not uniquely determined if the smallest eigenvalue is a multiple root. In that case, we return an error message and stop. Otherwise, the metric matrix $\boldsymbol{T}$ is given by

$$\boldsymbol{T} = \begin{pmatrix} \tau_1 & \tau_6/\sqrt{2} & \tau_5/\sqrt{2} \\ \tau_6/\sqrt{2} & \tau_2 & \tau_4/\sqrt{2} \\ \tau_5/\sqrt{2} & \tau_4/\sqrt{2} & \tau_3 \end{pmatrix}. \tag{60}$$

However, the sign of the eigenvector $\boldsymbol{\tau}$ is indeterminate. Since $\boldsymbol{T}$ should be positive semidefinite, we select the sign that makes $\det\boldsymbol{T} \geq 0$.

### C. Computation of the Translations

If the metric matrix $\boldsymbol{T}$ is determined, the values of $\{\zeta_\kappa\}$ and $\{\beta_\kappa\}$ are determined from the metric condition of eqs. (51), which can be rewritten as follows:

$$\begin{pmatrix} 1 & \tilde{t}_{x\kappa}^2 \\ 1 & \tilde{t}_{y\kappa}^2 \\ 0 & \tilde{t}_{x\kappa}\tilde{t}_{y\kappa} \end{pmatrix}\begin{pmatrix} 1/\zeta_\kappa^2 \\ \beta_\kappa^2 \end{pmatrix} = \begin{pmatrix} (\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) \\ (\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \\ (\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \end{pmatrix}. \tag{61}$$

This is overdetermination, so we compute the least-squares solution given by the following normal equation:

$$\begin{pmatrix} 2 & \tilde{t}_{x\kappa}^2 + \tilde{t}_{y\kappa}^2 \\ \tilde{t}_{x\kappa}^2 + \tilde{t}_{y\kappa}^2 & \tilde{t}_{x\kappa}^4 + \tilde{t}_{y\kappa}^4 + \tilde{t}_{x\kappa}^2\tilde{t}_{y\kappa}^2 \end{pmatrix}\begin{pmatrix} 1/\zeta_\kappa^2 \\ \beta_\kappa^2 \end{pmatrix} = \begin{pmatrix} (\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) + (\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \\ \tilde{t}_{x\kappa}^2(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) + \tilde{t}_{y\kappa}^2(\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \\ \quad + \tilde{t}_{x\kappa}\tilde{t}_{y\kappa}(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)}) \end{pmatrix}. \tag{62}$$

The solution is indeterminate if $\tilde{t}_{x\kappa} \approx 0$ and $\tilde{t}_{y\kappa} \approx 0$. In that case, we let $\beta_\kappa^2 = 0$ and solve only the first equation for $1/\zeta_\kappa^2$ in the form

$$\frac{1}{\zeta_\kappa^2} = \frac{(\boldsymbol{u}^\dagger_{\kappa(1)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(1)}) + (\boldsymbol{u}^\dagger_{\kappa(2)}, \boldsymbol{T}\boldsymbol{u}^\dagger_{\kappa(2)})}{2}. \tag{63}$$

From the resulting $1/\zeta_\kappa^2$ and $\beta_\kappa^2$, we let $\zeta_\kappa = 1/\sqrt{1/\zeta_\kappa^2}$ and $\beta_\kappa = \sqrt{\beta_\kappa^2}$. If $1/\zeta_\kappa^2 \leq 0$, we let $\zeta_\kappa$ be a sufficiently large value. If $\beta_\kappa^2 < 0$, we let $\beta_\kappa = 0$.

Letting the projection matrix $\Pi_\kappa$ and the projection vector $\pi_\kappa$ in the first of eqs. (8) be in the form of eqs. (36), we can obtain $t_{x\kappa}$ and $t_{y\kappa}$ in the form

$$\begin{pmatrix} t_{x\kappa} \\ t_{y\kappa} \end{pmatrix} = \zeta_\kappa \begin{pmatrix} \tilde{t}_{x\kappa} \\ \tilde{t}_{y\kappa} \end{pmatrix}. \qquad (64)$$

## D. Computation of the Rotations

If we let $\lambda_1 \geq \lambda_2 \geq \lambda_3$ be the eigenvalues of the metric matrix $\boldsymbol{T}$ and $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}$ the orthonormal system of the corresponding unit eigenvectors, $\boldsymbol{T}$ has the following spectral decomposition:

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \boldsymbol{v}_3 \end{pmatrix} \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3) \begin{pmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \boldsymbol{v}_3 \end{pmatrix}^\top. \qquad (65)$$

Since we choose the sign of $\boldsymbol{T}$ so that $|\boldsymbol{T}| \geq 0$, we have either $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ or $\lambda_1 \geq 0 \geq \lambda_2 \geq \lambda_3$. The latter may occur when $\lambda_1 \approx 0$ due to data inaccuracy. In that case, we let $\lambda_1 = 0$ and change the signs of $\lambda_2$ and $\lambda_3$.

From eq. (65), the rectifying matrix $\boldsymbol{A}$ has the form

$$\boldsymbol{A} = \pm \begin{pmatrix} \sqrt{\lambda_1}\boldsymbol{v}_1 & \sqrt{\lambda_2}\boldsymbol{v}_2 & \sqrt{\lambda_3}\boldsymbol{v}_3 \end{pmatrix} \boldsymbol{Q}, \qquad (66)$$

where $\boldsymbol{Q}$ is an arbitrary rotation matrix. This indeterminacy corresponds to the fact that we can arbitrarily define the orientation of the object coordinate system. The double sign $\pm$ implies the existence of the mirror image solution. So, we choose one solution by selecting $+$ and letting $\boldsymbol{Q} = \boldsymbol{I}$. From eq. (15), the vector $\boldsymbol{m}_i$, or the $i$th column of the matrix $\boldsymbol{M}$, is given by

$$\boldsymbol{m}_i = \sqrt{\lambda_i} \begin{pmatrix} (\boldsymbol{u}^\dagger_{1(1)}, \boldsymbol{v}_i) \\ (\boldsymbol{u}^\dagger_{1(2)}, \boldsymbol{v}_i) \\ (\boldsymbol{u}^\dagger_{2(1)}, \boldsymbol{v}_i) \\ \vdots \\ (\boldsymbol{u}^\dagger_{M(2)}, \boldsymbol{v}_i) \end{pmatrix}. \qquad (67)$$

The three columns $\boldsymbol{m}_1$, $\boldsymbol{m}_2$ and $\boldsymbol{m}_3$ determines the matrix $\boldsymbol{M}$, which determines the 3-D vectors $\{\boldsymbol{m}^\dagger_{\kappa(a)}\}$. Let $\boldsymbol{r}^\dagger_{\kappa(i)}$ the $i$th column of the transpose $\boldsymbol{R}_\kappa^\top$. Eq. (19) is rewritten as

$$\zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(1)} = \boldsymbol{r}^\dagger_{\kappa(1)} - \beta_\kappa t_{x\kappa} \boldsymbol{r}^\dagger_{\kappa(3)},$$
$$\zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(2)} = \boldsymbol{r}^\dagger_{\kappa(2)} - \beta_\kappa t_{y\kappa} \boldsymbol{r}^\dagger_{\kappa(3)}. \qquad (68)$$

Since $\{\boldsymbol{r}^\dagger_{\kappa(1)}, \boldsymbol{r}^\dagger_{\kappa(2)}, \boldsymbol{r}^\dagger_{\kappa(3)}\}$ is a right-handed orthonormal system, the vector product of eqs. (68) on both sides is

$$\zeta_\kappa^2 \boldsymbol{m}^\dagger_{\kappa(1)} \times \boldsymbol{m}^\dagger_{\kappa(2)} = \beta_\kappa t_{x\kappa} \boldsymbol{r}^\dagger_{\kappa(1)} + \beta_\kappa t_{y\kappa} \boldsymbol{r}^\dagger_{\kappa(2)} + \boldsymbol{r}^\dagger_{\kappa(3)}. \qquad (69)$$

Solving eqs. (68) and (69) for $\boldsymbol{r}^\dagger_{\kappa(1)}$, $\boldsymbol{r}^\dagger_{\kappa(2)}$, and $\boldsymbol{r}^\dagger_{\kappa(3)}$, we obtain

$$\boldsymbol{r}^\dagger_{\kappa(3)} = \zeta_\kappa \Big( \frac{\zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(1)} \times \boldsymbol{m}^\dagger_{\kappa(2)} - \beta_\kappa (t_{x\kappa} \boldsymbol{m}^\dagger_{\kappa(1)} + t_{y\kappa} \boldsymbol{m}^\dagger_{\kappa(2)})}{1 + \beta_\kappa^2 (t_{x\kappa}^2 + t_{y\kappa}^2)} \Big),$$
$$\boldsymbol{r}^\dagger_{\kappa(1)} = \zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(1)} + \beta_\kappa t_{x\kappa} \boldsymbol{r}^\dagger_{\kappa(3)},$$
$$\boldsymbol{r}^\dagger_{\kappa(2)} = \zeta_\kappa \boldsymbol{m}^\dagger_{\kappa(2)} + \beta_\kappa t_{y\kappa} \boldsymbol{r}^\dagger_{\kappa(3)}. \qquad (70)$$

However, the resulting $\{\boldsymbol{r}^\dagger_{\kappa(1)}, \boldsymbol{r}^\dagger_{\kappa(2)}, \boldsymbol{r}^\dagger_{\kappa(3)}\}$ may not be strictly orthonormal in the presence of noise in the data. In order to make them strictly orthonormal, we compute the following SVD:

$$\begin{pmatrix} \boldsymbol{r}^\dagger_{\kappa(1)} & \boldsymbol{r}^\dagger_{\kappa(2)} & \boldsymbol{r}^\dagger_{\kappa(3)} \end{pmatrix} = \boldsymbol{V}_\kappa \boldsymbol{\Lambda}_\kappa \boldsymbol{U}_\kappa^\top. \qquad (71)$$

An optimal rotation $\boldsymbol{R}_\kappa$ that best fits $\{\boldsymbol{r}^\dagger_{\kappa(1)}, \boldsymbol{r}^\dagger_{\kappa(2)}, \boldsymbol{r}^\dagger_{\kappa(3)}\}$ is given as follows [5]:

$$\boldsymbol{R}_\kappa = \boldsymbol{U}_\kappa \boldsymbol{V}_\kappa^\top. \qquad (72)$$

## E. Mirror Image Solution

If we choose "$-$" for the "$\pm$" in eq. (66), the vectors $\{\boldsymbol{m}_i\}$ given by eq. (67) change their signs. However, we can still obtain a solution compatible with eq. (13) if we change the signs of $a$, $b$ and $c$. So, the shape vector solution $\{\boldsymbol{s}_\alpha\}$ has its mirror image solution $\{-\boldsymbol{s}_\alpha\}$.

Changing the signs of the vectors $\{\boldsymbol{m}_i\}$ means changing the signs of the vectors $\{\boldsymbol{m}^\dagger_{\kappa(a)}\}$. If we take out equations that involve $\boldsymbol{R}_\kappa$ from eq. (46), we find that another solution $\boldsymbol{R}'_\kappa$ exists such that

$$\begin{pmatrix} 1 & 0 & -\beta_\kappa t_{x\kappa} \\ 0 & 1 & -\beta_\kappa t_{y\kappa} \end{pmatrix} \boldsymbol{R}_\kappa = -\begin{pmatrix} 1 & 0 & -\beta_\kappa t_{x\kappa} \\ 0 & 1 & -\beta_\kappa t_{y\kappa} \end{pmatrix} \boldsymbol{R}'_\kappa. \qquad (73)$$

Transposing both sides and letting

$$\boldsymbol{R}'_\kappa \boldsymbol{R}_\kappa^\top = \boldsymbol{\Omega}_\kappa, \qquad (74)$$

we can rewrite eq. (73) in the form

$$\boldsymbol{\Omega}_\kappa \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -\beta_\kappa t_{x\kappa} & -\beta_\kappa t_{y\kappa} \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ \beta_\kappa t_{x\kappa} & \beta_\kappa t_{y\kappa} \end{pmatrix}. \qquad (75)$$

This means that $\boldsymbol{\Omega}_\kappa$ is a rotation that maps $(1, 0, -\beta_\kappa t_{x\kappa})$ and $(0, 1, -\beta_\kappa t_{y\kappa})$ to $(-1, 0, \beta_\kappa t_{x\kappa})$ and $(0, -1, \beta_\kappa t_{y\kappa})$, respectively. Hence, $\boldsymbol{\Omega}_\kappa$ is a rotation by angle 180° around an axis perpendicular to the plane passing by $(1, 0, -\beta_\kappa t_{x\kappa})^\top$, $(0, 1, -\beta_\kappa t_{y\kappa})^\top$, and the origin $O$. The unit vector along the axis is given by

$$\boldsymbol{n}_\kappa = N[\begin{pmatrix} 1 \\ 0 \\ -\beta_\kappa t_{x\kappa} \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \\ -\beta_\kappa t_{y\kappa} \end{pmatrix}]. \qquad (76)$$

This is rewritten as the first of eqs. (49). The rotation $\boldsymbol{\Omega}_\kappa$ is then written in the form of the second of eqs. (49). From eq. (74), the mirror image rotation $\boldsymbol{R}'_\kappa$ is given by the third of eqs. (49).