IPSJ Transactions on Computer Vision and Applications Vol. 2 59–70 (Mar. 2010)

Research Paper

Compact Fundamental Matrix Computation

Kenichi Kanatani[†1] and Yasuyuki Sugaya[†2]

A very compact algorithm is presented for fundamental matrix computation from point correspondences over two images. The computation is based on the maximum likelihood (ML) principle, minimizing the reprojection error. The rank constraint is incorporated by the EFNS procedure. Although our algorithm produces the same solution as all existing ML-based methods, it is probably the most practical of all, being small and simple. By numerical experiments, we confirm that our algorithm behaves as expected.

1. Introduction

Computing the fundamental matrix from point correspondences is the first step of many computer vision applications including camera calibration, image rectification, structure from motion, and new view generation[7),26)]. Although its robustness is critical in practice, procedures for removing outlying matches depend on computation for inliers. For example, RANSAC-type methods randomly select matches and compute the fundamental matrix by hypothesizing that they are inliers. Then, the solution that has a maximum support is adopted[7),26)]. In this paper, we focus on computation for inliers.

Since feature points extracted from images have uncertainty to some degree, we need statistical optimization, modeling the uncertainty as "noise" that obeys a certain probability distribution. The standard model is independent Gaussian noise, which is coupled with maximum likelihood (ML) estimation. This results in the minimization of so called the reprojection error, also known as the *Gold Standard*[7)].

Although all existing ML-based methods minimize the same function, vast differences exist in their computational processes. This is mainly due to the fact

†1 Okayama University
†2 Toyohashi University of Technology

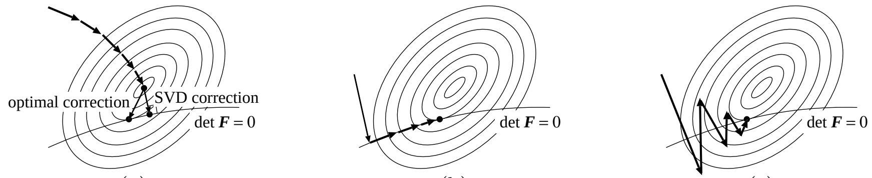


Fig. 1 (a) A posteriori correction. (b) Internal access. (c) External access.

that the fundamental matrix is constrained to have rank 2. The strategies for incorporating this constraint are roughly classified into three categories:

A posteriori correction. The fundamental matrix is first computed without considering the rank constraint and is modified a posteriori so as to satisfy it (Fig. 1(a)). If the rank constraint is not considered, the computation is vastly simplified[7),26)]. The crudest method, yet widely used, is to minimize the square sum of the epipolar equation, called *least squares*, *algebraic distance minimization*, or *8-point algorithm*[5)]. The *Taubin method*[24)] incorporates the data covariance matrices in the simplest way. These two yield the solution with simple algebraic manipulations[10),13)]. For incorporating the ML viewpoint, one needs iterations, for which many schemes exist including *FNS* (*Fundamental Numerical Scheme*)[3)], *HEIV* (*Heteroscedastic Errors-In-Variables*)[16),17)], and the *projective Gauss-Newton iterations*[11)]. For imposing the rank constraint, the most naive method, yet widely used, is to compute the SVD of the computed fundamental matrix and replace the smallest singular value by 0[5)]. A more sophisticated method is the *optimal correction*[9),17)]: the solution is moved in the statistically mostly likely direction until it satisfies the rank constraint (Fig. 1(a)).

Internal access. The fundamental matrix is parameterized so that the rank constraint is identically satisfied and is optimized in the ("internal") parameter space (Fig. 1(b)). Many types of such parameterization have been proposed including algebraic elimination of the rank constraint and the expression in terms of epipoles[18),26),27)]. Bartoli and Sturm[1)] regarded the SVD of the fundamental matrix as its parameterization and did search in an augmented space. Sugaya and Kanatani[19)] directly searched a 7-D space by the

Levenberg-Marquardt (LM) method.

External access. We do iterations in the ("external") 9-D space of the fundamental matrix in such a way that an optimal solution that satisfies the rank constraint automatically results (Fig. 1(c)). This concept was first introduced by Chojnacki et al.[4], who presented a scheme called *CFNS* (*Constrained Fundamental Numerical Scheme*).

In this paper, we present a new method based on the external access principle. Its description is far more compact than any of existing ML-based methods. Although there is no accuracy gain, since all ML-based methods minimize the same function, the compactness of the algorithm is of great advantage. The reason the non-optimal 8-point algorithm[5] is still widely used is probably for fear of coding a complicated program and uneasiness at relying on "download". Our algorithm is simple enough to code oneself⋆1, consisting only of vector and matrix operations in 9-D, just like the popular 8-point algorithm, yet producing an optimal solution. We describe our algorithm in Sec. 2 and give a derivation in Sec. 3. In Sec. 4, we confirm its performance by numerical experiments. We conclude in Sec. 5.

2. Optimal Fundamental Matrix Computation

Given two images of the same scene, suppose a point (x, y) in the first image corresponds to (x', y') in the second. We represent the corresponding points by 3-D vectors

$$\boldsymbol{x} = \begin{pmatrix} x/f_0 \\ y/f_0 \\ 1 \end{pmatrix}, \qquad \boldsymbol{x}' = \begin{pmatrix} x'/f_0 \\ y'/f_0 \\ 1 \end{pmatrix}, \tag{1}$$

where f_0 is a scaling constant of the order of the image size⋆2. As is well known, $\boldsymbol{x}$ and $\boldsymbol{x}'$ satisfy the *epipolar equation*,

$$(\boldsymbol{x}, \boldsymbol{F}\boldsymbol{x}') = 0, \tag{2}$$

where and hereafter we denote the inner product of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ by $(\boldsymbol{a}, \boldsymbol{b})$.

⋆1 But one can try ours if one wishes:
`http://www.iim.ics.tut.ac.jp/~sugaya/public-e.html`

⋆2 This is for stabilizing numerical computation[5]. In our experiments, we set $f_0 = 600$ pixels.

The matrix $\boldsymbol{F}$ is of rank 2 and called the *fundamental matrix*; it depends on the relative positions and orientations of the two cameras and their intrinsic parameters (e.g., their focal lengths) but not on the scene or the choice of the corresponding points. As can be seen from Eq. (2), the scale of the fundamental matrix $\boldsymbol{F}$ is indeterminate, so we normalize it to unit Frobenius norm $\|\boldsymbol{F}\| = 1$.

Suppose N correspondence pairs $\{\boldsymbol{x}_\alpha,\ \boldsymbol{x}'_\alpha\}_{\alpha=1}^N$ are detected. If the noise in their x- and y-coordinates is assumed to be independent, identical, and Gaussian, maximum likelihood (ML) is equivalent to minimizing the *reprojection error*

$$E = \sum_{\alpha=1}^{N} \Big(\|\boldsymbol{x}_\alpha - \bar{\boldsymbol{x}}_\alpha\|^2 + \|\boldsymbol{x}'_\alpha - \bar{\boldsymbol{x}}'_\alpha\|^2 \Big), \tag{3}$$

with respect to $\bar{\boldsymbol{x}}_\alpha$, $\bar{\boldsymbol{x}}'_\alpha$, and $\boldsymbol{F}$ subject to

$$(\bar{\boldsymbol{x}}_\alpha, \boldsymbol{F}\bar{\boldsymbol{x}}'_\alpha) = 0, \qquad \alpha = 1, ..., N. \tag{4}$$

No simple procedure exists for minimizing Eq. (3) subject to Eq. (4) and the rank constraint on $\boldsymbol{F}$. Many researchers minimized the Sampson error (to be discussed later) that approximates Eq. (3)[7),26)]. Alternatively, the minimization is done in an "augmented" parameter space, as done by Bartoli and Sturm[1], computing a tentative 3-D reconstruction and adjusting the camera positions and the intrinsic parameters so that the resulting projection images are as close to the input images as possible. Such a strategy is called *bundle adjustment*[25]. However, search in a high dimensional space, in particular if one wants a globally optimal solution, requires a large amount of computation[6),8)].

We now present a dramatically compact formulation, which is equivalent to bundle adjustment but not in high dimensions; the computation is done in the 9-D space of $\boldsymbol{F}$. Define 9-D vectors

$$
\boldsymbol{u} = \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix}, \qquad \boldsymbol{u}^\dagger \equiv \mathcal{N}[\begin{pmatrix} u_5u_9 - u_8u_6 \\ u_6u_7 - u_9u_4 \\ u_4u_8 - u_7u_5 \\ u_8u_3 - u_2u_9 \\ u_9u_1 - u_3u_7 \\ u_7u_2 - u_1u_8 \\ u_2u_6 - u_5u_3 \\ u_3u_4 - u_6u_1 \\ u_1u_5 - u_4u_2 \end{pmatrix}], \tag{5}
$$

where $\mathcal{N}[\,\cdot\,]$ denotes normalization to unit norm. The vector $\boldsymbol{u}$ encodes the nine elements of the fundamental matrix $\boldsymbol{F}$. The normalization $\|\boldsymbol{F}\| = 1$ is equivalent to $\|\boldsymbol{u}\| = 1$. The vector $\boldsymbol{u}^\dagger$ encodes the nine elements of the cofactor $\boldsymbol{F}^\dagger$ of $\boldsymbol{F}$, so we call $\boldsymbol{u}^\dagger$ the *cofactor vector* of $\boldsymbol{u}$. We denote by "$\det \boldsymbol{u}$" the determinant of the matrix $\boldsymbol{F}$ corresponding to $\boldsymbol{u}$.

In order to emphasize the compactness of our algorithm, we state it first and then give its derivation, which is straightforward but rather lengthy. The main routine of our algorithm goes as follows:

main

---

(1) Initialize $\boldsymbol{u}$, and let $E_0 = \infty$ (a sufficiently large number), $\hat{x}_\alpha = x_\alpha$, $\hat{y}_\alpha = y_\alpha$, $\hat{x}'_\alpha = x'_\alpha$, $\hat{y}'_\alpha = y'_\alpha$, and $\tilde{x}_\alpha = \tilde{y}_\alpha = \tilde{x}'_\alpha = \tilde{y}'_\alpha = 0$, $\alpha = 1, ..., N$.

(2) Compute the following 9-D vectors $\boldsymbol{\xi}_\alpha$ and the 9×9 matrices $V_0[\boldsymbol{\xi}_\alpha]$:

$$
\boldsymbol{\xi}_\alpha = \begin{pmatrix} \hat{x}_\alpha\hat{x}'_\alpha + \hat{x}'_\alpha\tilde{x}_\alpha + \hat{x}_\alpha\tilde{x}'_\alpha \\ \hat{x}_\alpha\hat{y}'_\alpha + \hat{y}'_\alpha\tilde{x}_\alpha + \hat{x}_\alpha\tilde{y}'_\alpha \\ f_0(\hat{x}_\alpha + \tilde{x}_\alpha) \\ \hat{y}_\alpha\hat{x}'_\alpha + \hat{x}'_\alpha\tilde{y}_\alpha + \hat{y}_\alpha\tilde{x}'_\alpha \\ \hat{y}_\alpha\hat{y}'_\alpha + \hat{y}'_\alpha\tilde{y}_\alpha + \hat{y}_\alpha\tilde{y}'_\alpha \\ f_0(\hat{y}_\alpha + \tilde{y}_\alpha) \\ f_0(\hat{x}'_\alpha + \tilde{x}'_\alpha) \\ f_0(\hat{y}'_\alpha + \tilde{y}'_\alpha) \\ f_0^2 \end{pmatrix}, \tag{6}
$$

$$
V_0[\boldsymbol{\xi}_\alpha] = \begin{pmatrix}
\hat{x}_\alpha^2 + \hat{x}'^2_\alpha & \hat{x}'_\alpha\hat{y}'_\alpha & f_0\hat{x}'_\alpha & \hat{x}_\alpha\hat{y}_\alpha & 0 & 0 & f_0\hat{x}_\alpha & 0 & 0 \\
\hat{x}'_\alpha\hat{y}'_\alpha & \hat{x}_\alpha^2 + \hat{y}'^2_\alpha & f_0\hat{y}'_\alpha & 0 & \hat{x}_\alpha\hat{y}_\alpha & 0 & 0 & f_0\hat{x}_\alpha & 0 \\
f_0\hat{x}'_\alpha & f_0\hat{y}'_\alpha & f_0^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hat{x}_\alpha\hat{y}_\alpha & 0 & 0 & \hat{y}_\alpha^2 + \hat{x}'^2_\alpha & \hat{x}'_\alpha\hat{y}'_\alpha & f_0\hat{x}'_\alpha & f_0\hat{y}_\alpha & 0 & 0 \\
0 & \hat{x}_\alpha\hat{y}_\alpha & 0 & \hat{x}'_\alpha\hat{y}'_\alpha & \hat{y}_\alpha^2 + \hat{y}'^2_\alpha & f_0\hat{y}'_\alpha & 0 & f_0\hat{y}_\alpha & 0 \\
0 & 0 & 0 & f_0\hat{x}'_\alpha & f_0\hat{y}'_\alpha & f_0^2 & 0 & 0 & 0 \\
f_0\hat{x}_\alpha & 0 & 0 & f_0\hat{y}_\alpha & 0 & 0 & f_0^2 & 0 & 0 \\
0 & f_0\hat{x}_\alpha & 0 & 0 & f_0\hat{y}_\alpha & 0 & 0 & f_0^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}. \tag{7}
$$

(3) Call *EFNS* (described below) to update $\boldsymbol{u}$.

(4) Update $\tilde{x}_\alpha$, $\tilde{y}_\alpha$, $\tilde{x}'_\alpha$, $\tilde{y}'_\alpha$, $\hat{x}_\alpha$, $\hat{y}_\alpha$, $\hat{x}'_\alpha$, and $\hat{y}'_\alpha$ by

$$
\begin{pmatrix} \tilde{x}_\alpha \\ \tilde{y}_\alpha \end{pmatrix} \leftarrow \frac{(\boldsymbol{u}, \boldsymbol{\xi}_\alpha)}{(\boldsymbol{u}, V[\hat{\boldsymbol{\xi}}_\alpha]\boldsymbol{u})} \begin{pmatrix} u_1 & u_2 & u_3 \\ u_4 & u_5 & u_6 \end{pmatrix} \begin{pmatrix} \hat{x}'_\alpha \\ \hat{y}'_\alpha \\ f_0 \end{pmatrix},
$$

$$
\begin{pmatrix} \tilde{x}'_\alpha \\ \tilde{y}'_\alpha \end{pmatrix} \leftarrow \frac{(\boldsymbol{u}, \boldsymbol{\xi}_\alpha)}{(\boldsymbol{u}, V[\hat{\boldsymbol{\xi}}_\alpha]\boldsymbol{u})} \begin{pmatrix} u_1 & u_4 & u_7 \\ u_2 & u_5 & u_8 \end{pmatrix} \begin{pmatrix} \hat{x}_\alpha \\ \hat{y}_\alpha \\ f_0 \end{pmatrix},
$$

$$
\hat{x}_\alpha \leftarrow x_\alpha - \tilde{x}_\alpha, \quad \hat{y}_\alpha \leftarrow y_\alpha - \tilde{y}_\alpha, \quad \hat{x}'_\alpha \leftarrow x'_\alpha - \tilde{x}'_\alpha, \quad \hat{y}'_\alpha \leftarrow y'_\alpha - \tilde{y}_\alpha. \tag{8}
$$

(5) Compute the reprojection error

$$
E = \sum_{\alpha=1}^{N} (\tilde{x}_\alpha^2 + \tilde{y}_\alpha^2 + \tilde{x}'^2_\alpha + \tilde{y}'^2_\alpha). \tag{9}
$$

If $E \approx E_0$, return $\boldsymbol{u}$ and stop. Else, let $E_0 \leftarrow E$ and go back to Step (2).

In our preliminary version[14)], the least-squares method (see Appendix A.1) was suggested for the initialization in Step (1). Later, we have experimentally found that the Taubin method[24)] (see Appendix A.2) results in better convergence performance in the presence of large noise. In our preliminary version[14)], we stopped the iterations when the update of $\boldsymbol{u}$ becomes sufficiently small, but we have later experimentally found that the use of the reprojection error as in Step

(5) exhibits stabler convergence, because the EFNS procedure in Step (3), as described below, is iterated with respect to $\boldsymbol{u}$:

EFNS

(1) Compute the following 9×9 matrices $\boldsymbol{M}$ and $\boldsymbol{L}$:

$$\boldsymbol{M} = \sum_{\alpha=1}^{N} \frac{\boldsymbol{\xi}_\alpha \boldsymbol{\xi}_\alpha^\top}{(\boldsymbol{u}, V_0[\boldsymbol{\xi}_\alpha]\boldsymbol{u})}, \qquad \boldsymbol{L} = \sum_{\alpha=1}^{N} \frac{(\boldsymbol{u}, \boldsymbol{\xi}_\alpha)^2 V_0[\boldsymbol{\xi}_\alpha]}{(\boldsymbol{u}, V_0[\boldsymbol{\xi}_\alpha]\boldsymbol{u})^2}. \tag{10}$$

(2) Compute the cofactor vector $\boldsymbol{u}^\dagger$ in Eqs. (5) and the 9×9 projection matrix

$$\boldsymbol{P}_{\mathbf{u}^\dagger} \equiv \boldsymbol{I} - \boldsymbol{u}^\dagger \boldsymbol{u}^{\dagger\top}. \tag{11}$$

(3) Compute the following 9×9 matrices:

$$\boldsymbol{X} = \boldsymbol{M} - \boldsymbol{L}, \qquad \boldsymbol{Y} = \boldsymbol{P}_{\mathbf{u}^\dagger} \boldsymbol{X} \boldsymbol{P}_{\mathbf{u}^\dagger}. \tag{12}$$

(4) Compute the two unit eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ of $\boldsymbol{Y}$ for the smallest eigenvalues, and compute

$$\hat{\boldsymbol{u}} = (\boldsymbol{u}, \boldsymbol{v}_1)\boldsymbol{v}_1 + (\boldsymbol{u}, \boldsymbol{v}_2)\boldsymbol{v}_2. \tag{13}$$

(5) Compute

$$\boldsymbol{u}' = \mathcal{N}[\boldsymbol{P}_{\mathbf{u}^\dagger} \hat{\boldsymbol{u}}]. \tag{14}$$

(6) If $\boldsymbol{u}' \approx \boldsymbol{u}$ up to sign, return $\boldsymbol{u}'$ and stop. Else, let $\boldsymbol{u} \leftarrow \mathcal{N}[\boldsymbol{u} + \boldsymbol{u}']$ and go back to Step (1).

In our preliminary version[14)], two eigenvectors for the eigenvalues having the smallest absolute value were computed in Step (4). Later, we have experimentally found that computing two eigenvectors for the smallest eigenvalues results in better convergence in the presence of large noise. The same observation was made for the FNS of Chojnacki et al.[3)], for which computing the smallest eigenvalue, rather than the eigenvalue having the smallest absolute value, exhibits better convergence (see 11), 13) for experimental comparison).

3. Derivation

The derivation of the main routine and the EFNS procedure is rather lengthy. To make understanding easy, we describe the basic logic here. The omitted details are given in the Appendix.

3.1 Derivation of the Main Routine

3.1.1 First Approximation.

We want to compute $\bar{\boldsymbol{x}}_\alpha$ and $\bar{\boldsymbol{x}}'_\alpha$ that minimize Eq. (3) subject to Eq. (4), but we may alternatively write

$$\bar{\boldsymbol{x}}_\alpha = \boldsymbol{x}_\alpha - \Delta\boldsymbol{x}_\alpha, \qquad \bar{\boldsymbol{x}}'_\alpha = \boldsymbol{x}'_\alpha - \Delta\boldsymbol{x}'_\alpha, \tag{15}$$

and compute the correction terms $\Delta\boldsymbol{x}_\alpha$ and $\Delta\boldsymbol{x}'_\alpha$. Substituting Eqs. (15) into Eq. (4), we have

$$E = \sum_{\alpha=1}^{N} \Big(\|\Delta\boldsymbol{x}_\alpha\|^2 + \|\Delta\boldsymbol{x}'_\alpha\|^2 \Big). \tag{16}$$

The epipolar equation, Eq. (4), becomes

$$(\boldsymbol{x}_\alpha - \Delta\boldsymbol{x}_\alpha, \boldsymbol{F}(\boldsymbol{x}'_\alpha - \Delta\boldsymbol{x}'_\alpha)) = 0. \tag{17}$$

Ignoring the second order term in the correction terms, we obtain

$$(\boldsymbol{F}\boldsymbol{x}'_\alpha, \Delta\boldsymbol{x}_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \Delta\boldsymbol{x}'_\alpha) = (\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha). \tag{18}$$

Since the correction should be done in the image plane, we have the constraints

$$(\boldsymbol{k}, \Delta\boldsymbol{x}_\alpha) = 0, \qquad (\boldsymbol{k}, \Delta\boldsymbol{x}'_\alpha) = 0, \tag{19}$$

where we define $\boldsymbol{k} \equiv (0, 0, 1)^\top$. Introducing Lagrange multipliers for Eqs. (18) and (19), we obtain $\Delta\boldsymbol{x}_\alpha$ and $\Delta\boldsymbol{x}'_\alpha$ that minimize Eq. (16) as follows (see Appendix A.3):

$$\Delta\boldsymbol{x}_\alpha = \frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha}{(\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)},$$
$$\Delta\boldsymbol{x}'_\alpha = \frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha}{(\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)}. \tag{20}$$

Here, $\boldsymbol{P}_{\mathbf{k}}$ is the 3×3 projection matrix along $\boldsymbol{k}$:

$$\boldsymbol{P}_{\mathbf{k}} \equiv \boldsymbol{I} - \boldsymbol{k}\boldsymbol{k}^\top. \tag{21}$$

Substituting Eqs. (20) into Eq. (16), we obtain (see Appendix A.4)

$$E = \sum_{\alpha=1}^{N} \frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)^2}{(\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)}, \tag{22}$$

which is known as the *Sampson error*[7)]. Suppose we have obtained the matrix

$\boldsymbol{F}$ that minimizes Eq. (22) subject to $\det \boldsymbol{F} = 0$. Writing it as $\hat{\boldsymbol{F}}$ and substituting it into Eqs. (15), we obtain

$$\hat{\boldsymbol{x}}_\alpha = \boldsymbol{x}_\alpha - \frac{(\boldsymbol{x}_\alpha, \hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha}{(\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha) + (\hat{\boldsymbol{F}}^\top \boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top \boldsymbol{x}_\alpha)},$$
$$\hat{\boldsymbol{x}}'_\alpha = \boldsymbol{x}'_\alpha - \frac{(\boldsymbol{x}_\alpha, \hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top \boldsymbol{x}_\alpha}{(\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha) + (\hat{\boldsymbol{F}}^\top \boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top \boldsymbol{x}_\alpha)}. \tag{23}$$

3.1.2 Higher Order Correction.

The solution of Eqs. (23) is only a first approximation. So, we estimate the true solution $\bar{\boldsymbol{x}}_\alpha$ and $\bar{\boldsymbol{x}}'_\alpha$ by writing, instead of Eqs. (15),

$$\bar{\boldsymbol{x}}_\alpha = \hat{\boldsymbol{x}}_\alpha - \Delta\hat{\boldsymbol{x}}_\alpha, \qquad \bar{\boldsymbol{x}}'_\alpha = \hat{\boldsymbol{x}}'_\alpha - \Delta\hat{\boldsymbol{x}}'_\alpha, \tag{24}$$

and computing the correction terms $\Delta\hat{\boldsymbol{x}}_\alpha$ and $\Delta\hat{\boldsymbol{x}}'_\alpha$, which are small quantities of higher order than $\Delta\boldsymbol{x}_\alpha$ and $\Delta\boldsymbol{x}'_\alpha$. Substitution of Eqs. (24) into Eq. (3) yields

$$E = \sum_{\alpha=1}^{N}\Big(\|\tilde{\boldsymbol{x}}_\alpha + \Delta\hat{\boldsymbol{x}}_\alpha\|^2 + \|\tilde{\boldsymbol{x}}'_\alpha + \Delta\hat{\boldsymbol{x}}'_\alpha\|^2\Big), \tag{25}$$

where we define

$$\tilde{\boldsymbol{x}}_\alpha = \boldsymbol{x}_\alpha - \hat{\boldsymbol{x}}_\alpha, \qquad \tilde{\boldsymbol{x}}'_\alpha = \boldsymbol{x}'_\alpha - \hat{\boldsymbol{x}}'_\alpha. \tag{26}$$

The epipolar equation, Eq. (4), now becomes

$$(\hat{\boldsymbol{x}}_\alpha - \Delta\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}(\hat{\boldsymbol{x}}'_\alpha - \Delta\hat{\boldsymbol{x}}'_\alpha)) = 0. \tag{27}$$

Ignoring second order term in $\Delta\hat{\boldsymbol{x}}_\alpha$ and $\Delta\hat{\boldsymbol{x}}'_\alpha$, we have

$$(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \Delta\hat{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \Delta\hat{\boldsymbol{x}}'_\alpha) = (\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha). \tag{28}$$

This is a higher order approximation of Eq. (4) than Eq. (18). Introducing Lagrange multipliers to Eq. (28) and the constraints

$$(\boldsymbol{k}, \Delta\hat{\boldsymbol{x}}_\alpha) = 0, \qquad (\boldsymbol{k}, \Delta\hat{\boldsymbol{x}}'_\alpha) = 0, \tag{29}$$

we obtain $\Delta\hat{\boldsymbol{x}}_\alpha$ and $\Delta\hat{\boldsymbol{x}}'_\alpha$ as follows (see Appendix A.5):

$$\Delta\hat{\boldsymbol{x}}_\alpha = \frac{\Big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)} - \tilde{\boldsymbol{x}}_\alpha,$$
$$\Delta\hat{\boldsymbol{x}}'_\alpha = \frac{\Big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)} - \tilde{\boldsymbol{x}}'_\alpha. \tag{30}$$

The reprojection error of Eq. (25) now has the form (see Appendix A.6)

$$E = \sum_{\alpha=1}^{N}\frac{\Big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)^2}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)}. \tag{31}$$

Suppose we have obtained the matrix $\boldsymbol{F}$ that minimizes this subject to $\det \boldsymbol{F} = 0$. Writing it as $\hat{\boldsymbol{F}}$ and substituting it into Eqs. (30), we obtain from Eqs. (26) the solution

$$\hat{\hat{\boldsymbol{x}}}_\alpha = \boldsymbol{x}_\alpha - \frac{\Big((\hat{\boldsymbol{x}}_\alpha, \hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha) + (\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)\boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha}{(\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha) + (\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha)},$$
$$\hat{\hat{\boldsymbol{x}}}'_\alpha = \boldsymbol{x}'_\alpha - \frac{\Big((\hat{\boldsymbol{x}}_\alpha, \hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha) + (\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)\boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha}{(\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha) + (\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha)}. \tag{32}$$

The resulting $\{\hat{\hat{\boldsymbol{x}}}_\alpha, \hat{\hat{\boldsymbol{x}}}'_\alpha\}$ are a better approximation than $\{\hat{\boldsymbol{x}}_\alpha, \hat{\boldsymbol{x}}'_\alpha\}$. Rewriting $\{\hat{\hat{\boldsymbol{x}}}_\alpha, \hat{\hat{\boldsymbol{x}}}'_\alpha\}$ as $\{\hat{\boldsymbol{x}}_\alpha, \hat{\boldsymbol{x}}'_\alpha\}$, we repeat this computation until the iterations converge. In the end, $\Delta\hat{\boldsymbol{x}}_\alpha$ and $\Delta\hat{\boldsymbol{x}}'_\alpha$ in Eq. (27) become $\mathbf{0}$, and the epipolar equation is exactly satisfied.

3.1.3 Compact 9-D Description.

The above algorithm is greatly simplified by using the 9-D vector encoding of Eqs. (5). The definition of $\boldsymbol{\xi}_\alpha$ in Eq. (6) and $V_0[\boldsymbol{\xi}_\alpha]$ in Eq. (7) implies the following identities:

$$(\hat{\boldsymbol{x}}_\alpha, \hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha) + (\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\hat{\boldsymbol{F}}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha) = \frac{(\boldsymbol{u}, \boldsymbol{\xi}_\alpha)}{f_0^2}, \tag{33}$$
$$(\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}\boldsymbol{x}'_\alpha) + (\hat{\boldsymbol{F}}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\hat{\boldsymbol{F}}^\top\boldsymbol{x}_\alpha) = \frac{(\boldsymbol{u}, V_0[\boldsymbol{\xi}_\alpha]\boldsymbol{u})}{f_0^2}. \tag{34}$$

If we define $\tilde{\boldsymbol{x}}_\alpha$ and $\tilde{\boldsymbol{x}}'_\alpha$ by Eqs. (26), we obtain from Eqs. (32) the update form in Eqs. (8). If we let $\hat{x}_\alpha = x_\alpha$, $\hat{y}_\alpha = y_\alpha$, $\hat{x}'_\alpha = x'_\alpha$, $\hat{y}'_\alpha = y'_\alpha$, and $\tilde{x}_\alpha = \tilde{y}_\alpha = \tilde{x}'_\alpha = \tilde{y}'_\alpha = 0$, as in the Step (1) of the main routine, the update form of Eqs. (8) is equivalent to Eqs. (23). Thus, the main routine is completed except Step (3), where we need to minimize Eqs. (22) and (31) subject to $\det \boldsymbol{F} = 0$. We now describe our EFNS procedure for this.

3.2 Derivation of EFNS

3.2.1 Problem.

Using the identities in Eqs. (33) and (34), we can rewrite Eq. (31) as

$$E = \frac{1}{f_0^2}\sum_{\alpha=1}^{N}\frac{(\boldsymbol{u}, \boldsymbol{\xi}_\alpha)^2}{(\boldsymbol{u}, V_0[\boldsymbol{\xi}_\alpha]\boldsymbol{u})}. \tag{35}$$

If we let $\hat{x}_\alpha = x_\alpha$, $\hat{y}_\alpha = y_\alpha$, $\hat{x}'_\alpha = x'_\alpha$, $\hat{y}'_\alpha = y'_\alpha$, and $\tilde{x}_\alpha = \tilde{y}_\alpha = \tilde{x}'_\alpha = \tilde{y}'_\alpha = 0$, as in the Step (1) of the main routine, this reduces to the Sampson error in Eq. (22). Thus, the remaining problem is to minimize Eq. (35) subject to $\det \boldsymbol{u} = 0$.

3.2.2 Geometry.

The necessary and sufficient condition for E to be stationary at a point $\boldsymbol{u}$ on the 8-D unit sphere $\mathcal{S}^8$ in $\mathcal{R}^9$ is that its gradient $\nabla_{\mathbf{u}} E$ is orthogonal to the hypersurface defined by $\det \boldsymbol{u} = 0$. Direct manipulation shows

$$\boldsymbol{u}^\dagger = \mathcal{N}[\nabla_{\mathbf{u}} \det \boldsymbol{u}]. \tag{36}$$

In other words, $\boldsymbol{u}^\dagger$ is the unit surface normal to the hypersurface defined by $\det \boldsymbol{u} = 0$. It follows that $\nabla_{\mathbf{u}} E$ should be parallel to the cofactor vector $\boldsymbol{u}^\dagger$ at the stationary point. Differentiating Eq. (35) with respect to $\boldsymbol{u}$, we see that

$$\nabla_{\mathbf{u}} E = \frac{2}{f_0^2}\boldsymbol{X}\boldsymbol{u}, \tag{37}$$

where $\boldsymbol{X}$ is the matrix in Eqs. (12). Using the projection matrix $\boldsymbol{P}_{\mathbf{u}^\dagger}$ in Eq. (11), we can express the parallelism of $\nabla_{\mathbf{u}} E$ and $\boldsymbol{u}^\dagger$ as

$$\boldsymbol{P}_{\mathbf{u}^\dagger}\boldsymbol{X}\boldsymbol{u} = \boldsymbol{0}. \tag{38}$$

The rank constraint $\det \boldsymbol{u} = 0$ is equivalently written as

$$(\boldsymbol{u}^\dagger, \boldsymbol{u}) = 0, \tag{39}$$

which is a direct consequence of the identity $\boldsymbol{F}^\dagger \boldsymbol{F} = (\det \boldsymbol{F})\boldsymbol{I}$. In terms of the projection matrix $\boldsymbol{P}_{\mathbf{u}^\dagger}$, the rank constraint Eq. (39) is equivalently written as

$$\boldsymbol{P}_{\mathbf{u}^\dagger}\boldsymbol{u} = \boldsymbol{u}. \tag{40}$$

It follows that the stationarity condition of Eq. (38) is written as

$$\boldsymbol{Y}\boldsymbol{u} = \boldsymbol{0}, \tag{41}$$

where $\boldsymbol{Y}$ is the matrix defined in Eqs. (12). Our task is to compute the solution $\boldsymbol{u}$ that satisfies the stationarity condition of Eq. (41) and the rank constraint Eq. (40).

3.2.3 Justification of the Procedure.

We now show that the desired solution can be obtained by the EFNS routine in Sec. 2. To show this, we prove that when the iterations have converged, the eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ of $\boldsymbol{Y}$ both have eigenvalue 0. From the definition of $\boldsymbol{Y}$ in Eqs. (12) and $\boldsymbol{P}_{\mathbf{u}^\dagger}$ in Eq. (11), the cofactor vector $\boldsymbol{u}^\dagger$ is always an eigenvector of $\boldsymbol{Y}$ with eigenvalue 0. This means that either $\boldsymbol{v}_1$ or $\boldsymbol{v}_2$ has eigenvalue 0. Suppose $\boldsymbol{v}_1$ has nonzero eigenvalue λ ($\neq 0$). Then, $\boldsymbol{v}_2 = \pm\boldsymbol{u}^\dagger$. By construction, the vector $\hat{\boldsymbol{u}}$ in Eq. (13) belongs to the linear span of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ ($= \pm\boldsymbol{u}^\dagger$), which are mutually orthogonal, and the vector $\boldsymbol{u}'$ in Eq. (14) is a projection of $\hat{\boldsymbol{u}}$ within that linear span onto the direction orthogonal to $\boldsymbol{u}^\dagger$. Hence, $\boldsymbol{u}'$ should coincide with $\pm\boldsymbol{v}_1$. After the iterations have converged, we have $\boldsymbol{u} = \boldsymbol{u}'$ ($= \pm\boldsymbol{v}_1$), so $\boldsymbol{u}$ is an eigenvector of $\boldsymbol{Y}$ with eigenvalue λ, i.e., $\boldsymbol{Y}\boldsymbol{u} = \lambda\boldsymbol{u}$. Computing the inner product with $\boldsymbol{u}$ on both sides, we have

$$(\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{u}) = \lambda. \tag{42}$$

On the other hand, $\boldsymbol{u}$ ($= \pm\boldsymbol{v}_1$) is orthogonal to the cofactor vector $\boldsymbol{u}^\dagger$ ($= \pm\boldsymbol{v}_2$), so $\boldsymbol{P}_{\mathbf{u}^\dagger}\boldsymbol{u} = \boldsymbol{u}$. Hence,

$$(\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{u}) = (\boldsymbol{u}, \boldsymbol{P}_{\mathbf{u}^\dagger}\boldsymbol{X}\boldsymbol{P}_{\mathbf{u}^\dagger}\boldsymbol{u}) = (\boldsymbol{u}, \boldsymbol{X}\boldsymbol{u}) = 0, \tag{43}$$

because from the definition of $\boldsymbol{X}$ in Eqs. (12) we see that $(\boldsymbol{u}, \boldsymbol{X}\boldsymbol{u}) = 0$ is an identity in $\boldsymbol{u}$. In fact, we can confirm from the definition of $\boldsymbol{M}$ and $\boldsymbol{L}$ in Eqs. (10) that $(\boldsymbol{u}, \boldsymbol{M}\boldsymbol{u}) = (\boldsymbol{u}, \boldsymbol{L}\boldsymbol{u})$ holds identically in $\boldsymbol{u}$. Since Eqs. (42) and (43) contradict our assumption that $\lambda \neq 0$, $\boldsymbol{v}_1$ is also an eigenvector of $\boldsymbol{Y}$ with eigenvalue 0. Thus, Eqs. (40) and (41) both hold, so $\boldsymbol{u}$ is the desired solution.

3.2.4 Observations.

The EFNS was first introduced by Kanatani and Sugaya[12] as a general constrained parameter estimation in abstract terms. It is a straightforward extension of the FNS of Chojnacki et al.[3] (See Appendix A.7) to an arbitrary number of constraints. In fact, if we replace $\boldsymbol{P}_{\mathbf{u}^\dagger}$ in Eqs. (12) by the identity $\boldsymbol{I}$, the resulting procedure reduces to FNS. For this reason, Kanatani and Sugaya[12] called it *EFNS* (*Extended FNS*). They applied it to minimization of the Sampson error in Eq. (22) and pointed out that the CFNS of Chojnacki et al.[4] does not necessarily converge to a correct solution while EFNS does (see Appendix A.8). Our finding here is that we can also minimize the reprojection error by *repeated* use of EFNS after introducing the *intermediate variables* $\boldsymbol{\xi}_\alpha$ and $V_0[\boldsymbol{\xi}_\alpha]$ as in Eqs. (6) and (7).

The justification of EFNS described earlier relies on the premise that the iterations converge. As pointed out in 12), if we let $\boldsymbol{u} \leftarrow \boldsymbol{u}'$ in the Step (6) of the EFNS routine, the next value of $\boldsymbol{u}'$ computed in Step (5) often reverts to the former value of $\boldsymbol{u}$, falling in infinite looping. So, the "midpoint" $(\boldsymbol{u}' + \boldsymbol{u})/2$ is normalized to a unit vector $\mathcal{N}[\boldsymbol{u}' + \boldsymbol{u}]$. This greatly improves convergence. In fact, we have confirmed that this technique also improves the convergence of FNS, which sometimes oscillates in the presence of very large noise.

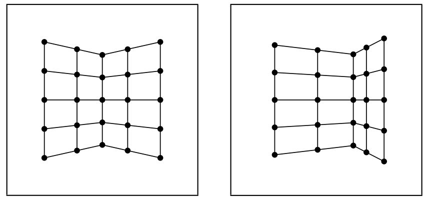
Theoretically speaking, our algorithm may not produce a global minimum of the reprojection error in Eq. (3). The problem is not the main routine, which is essentially the optimal triangulation of Kanatani et al.[15]. However, EFNS is not theoretically guaranteed to reach the absolute minimum of E in Eq. (35), although we have never experienced the contrary in all our experiments.

If the EFNS routine is replaced by a method that can always reach the global minimum of the Sampson error, e.g., by incorporating the branch and bound principle[6),8)], our scheme automatically converts to a method that always computes the global minimum of the reprojection error. However, such a conversion may lose the efficiency and compactness of our scheme.

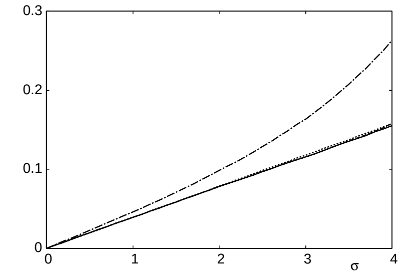
4. Performance Confirmation

Figure 2(a) shows simulated images of two planar grid surfaces. The image size is 600×600 pixels with 1200 pixel focal length. We added random Gaussian noise of mean 0 and standard deviation σ to the x- and y-coordinates of each grid point independently and from them computed the fundamental matrix.

Since all existing ML-based methods minimize the same reprojection error, their mutual accuracy comparison does not make sense. Rather, our concern is if our algorithm really converges to a correct solution. To see this, we compare our algorithm with a carefully tuned alternative method: we compute an initial solution by least squares, from which we start the FNS of Chojnacki et al.[3], and the resulting solution is optimally corrected to satisfy the rank constraint



(a)



(b)

Fig. 2 (a) Simulated images of planar grid surfaces. (b) The RMS error vs. noise level σ. The plot for our algorithm (solid line) and plot for the alternative method (dashed line) completely overlap. Chained line: the 8-point algorithm. Dotted line: KCR lower bound.

(see Appendix A.9). From it, we start a direct 7-D search, using the Levenberg-Marquardt (LM) method[19].

Figure 2(b) plots, for each σ, the root mean square (RMS) of $\|\boldsymbol{P}_{\mathcal{U}}\hat{\boldsymbol{u}}\|$ for the computed solution $\hat{\boldsymbol{u}}$ over 10,000 independent trials with different noise, where $\boldsymbol{P}_{\mathcal{U}}$ ($\equiv \boldsymbol{I} - \boldsymbol{u}\boldsymbol{u}^\top - \boldsymbol{u}^\dagger\boldsymbol{u}^{\dagger\top}$) denotes projection onto the space of deviations from the true solution $\boldsymbol{u}$ that satisfies the rank constraint $\det \boldsymbol{u} = 0$. As a reference, the chained line shows the corresponding result of the 8-point algorithm[5] (least squares followed by SVD rank correction), and the dotted line indicates the theoretical accuracy limit (*KCR lower bound*[2),9)]).

Fig. 2(b) actually shows the plot for our algorithm (solid line) and the plot for the alternative method (dashed line), but they overlap so completely that we cannot tell the difference. Thus, the same solution is reached although their paths of approach may be different (Fig. 1). We also see that the accuracy almost coincides with the theoretical limit, so no further improvement can be hoped for. As predicted, the 8-point algorithm performs poorly. Doing many experiments (not all shown here), we observed the following:

(1) The main routine converges after a few (at most four) iterations.

(2) If we skip Step (3) and stop at Step (5) without doing any further iterations, we obtain the Sampson solution. Yet, it coincides with the final solution up to three to four decimal places. The high accuracy of the Sampson solution

was also noted by Zhang[26].

(3) If initialized by least squares, the 7-D search does not necessarily arrive at the true minimum of the reprojection error, being trapped to local minima, as reported in 20). After a careful tuning as described above, the solution coincides with our algorithm, which directly arrives at the same solution without any such tuning.

5. Concluding Remarks

We have presented a very compact algorithm for computing the fundamental matrix from point correspondences over two images based on the ML principle using the EFNS procedure. The computation consists of vector and matrix operations in the 9-D space of $\boldsymbol{F}$ as the 8-point algorithm. By numerical experiments, we have confirmed that our algorithm behaves satisfactorily. Because of its compactness and good performance, we expect it to be a standard tool for fundamental matrix computation.

The performance of our algorithm depends on the performance of the EFNS routine, which minimizes the Sampson error in Eq. (35) subject to the rank constraint $\det \boldsymbol{u} = 0$. If another method is available for minimizing Eq. (35), we can substitute it for the EFNS, and the main routine minimizes the reprojection error in Eq. (3). The problem of finding a better method for Sampson error minimization is open for future research.

Acknowledgments The authors thank Mike Brooks and Wojciech Chojnacki of the University of Adelaide, Australia and Nikolai Chernov of the University of Alabama at Birmingham, U.S.A. for helpful discussions on this problem.

References

1) Bartoli, A. and Sturm, P.: Nonlinear estimation of fundamental matrix with minimal parameters, *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol.26, No.3, pp.426–432 (2004).
2) Chernov, N. and Lesort, C.: Statistical efficiency of curve fitting algorithms, *Comp. Stat. Data Anal.*, Vol.47, No.4, pp.713–728 (2004).
3) Chojnacki, W., Brooks, M.J., van den Hengel, A. and Gawley, D.: On the fitting of surfaces to data with covariances, *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol.22, No.11, pp.1294–1303 (2000).
4) Chojnacki, W., Brooks, M.J., van den Hengel, A. and Gawley, D.: A new constrained parameter estimator for computer vision applications, *Image Vis. Comput.*, Vol.22, No.2, pp.85–91 (2004).
5) Hartley, R.I.: In defense of the eight-point algorithm, *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol.19, No.6, pp.580–593 (1997).
6) Hartley, R. and Kahl, F.: Optimal algorithms in multiview geometry, *Proc. 8th Asian Conf. Computer Vision*, Tokyo, Japan, November 2007, Vol.1, pp.13–34 (2007).
7) Hartley, R. and Zisserman, A.: *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge (2000).
8) Kahl, F. and Henrion, D.: Globally optimal estimates for geometric reconstruction problems, *Int. J. Comput. Vis.*, Vol.74, No.1, pp.3–15 (2007).
9) Kanatani, K.: *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier Science, Amsterdam, The Netherlands, 1996; reprinted, Dover, New York (2005)
10) Kanatani, K.: Statistical optimization for geometric fitting: Theoretical accuracy analysis and high order error analysis, *Int. J. Compt. Vis.*, Vol.80, No.2, pp.167–188 (2008).
11) Kanatani, K. and Sugaya, Y.: High accuracy fundamental matrix computation and its performance evaluation, *IEICE Trans. Inf. & Syst.*, Vol.E90-D, No.2, pp.579–585 (2007).
12) Kanatani, K. and Sugaya, Y.: Extended FNS for constrained parameter estimation, *Proc. 10th Meeting Image Recognition Understanding*, Hiroshima, Japan, July 2007, pp.219–226 (2007).
13) Kanatani, K. and Sugaya, Y.: Performance evaluation of iterative geometric fitting algorithms, *Comp. Stat. Data Anal.*, Vol.52, No.2, pp.1208–1222 (2007).
14) Kanatani, K. and Sugaya, Y.: Compact fundamental matrix computation, *Proc. IEEE Pacific Rim Symp. Image Video Tech.*, Tokyo, Japan, January 2009, pp.179–190 (2009).
15) Kanatani, K., Sugaya, Y. and Niitsuma, H.: Triangulation from two views revisited: Hartley-Sturm vs. optimal correction, *Proc. 19th British Mach. Vis. Conf.*, September 2008, pp.173–182 (2008).
16) Leedan, Y. and Meer, P.: Heteroscedastic regression in computer vision: Problems with bilinear constraint, *Int. J. Comput. Vis.*, Vol.37, No.2, pp.127–150 (2000).
17) Matei, J. and Meer, P.: Estimation of nonlinear errors-in-variables models for computer vision applications, *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol.28, No.10, pp.1537–1552 (2006).
18) Migita, T. and Shakunaga, T.: One-dimensional search for reliable epipole estimation, *Proc. IEEE Pacific Rim Symp. Image Video Tech.*, Hsinchu, Taiwan, December 2006, pp.1215–1224 (2006)
19) Sugaya, Y. and Kanatani, K.: High accuracy computation of rank-constrained

fundamental matrix, *Proc. 18th British Mach. Vis. Conf.*, September 2007, Vol.1, pp.282–291 (2007).
20) Sugaya, Y. and Kanatani, K.: Highest accuracy fundamental matrix computation, *Proc. 8th Asian Conf. Comput. Vis.*, Tokyo, Japan, November 2008, Vol. 2, pp. 311–321 (2008).
21) Tagawa, N., Toriu, T. and Endoh, T.: Un-biased linear algorithm for recovering three-dimensional motion from optical flow, *IEICE Trans. Inf. & Syst.*, Vol.E76-D, No.10, pp.1263–1275 (1993).
22) Tagawa, N., Toriu, T. and Endoh, T.: Estimation of 3-D motion from optical flow with unbiased objective function, *IEICE Trans. Inf. & Syst.*, Vol.E77-D, No.10, pp.1148–1161 (1994).
23) Tagawa, N., Toriu, T. and Endoh, T.: 3-D motion estimation from optical flow with low computational cost and small variance, *IEICE Trans. Inf. & Syst.*, Vol.E79-D, No.3, pp.230–241 (1996).
24) Taubin, G.: Estimation of planar curves, surfaces, and non-planar space curves defined by implicit equations with applications to edge and range image segmentation, *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol.13, No.11, pp.1115–1138 (1991).
25) Triggs, B., McLauchlan, P.F., Hartley, R.I. and Fitzgibbon, A.: Bundle adjustment—A modern synthesis, *Vision Algorithms: Theory and Practice*, Triggs, B. Zisserman, A. and Szeliski, R.: (Eds.), Springer, Berlin (2000).
26) Zhang, Z.: Determining the epipolar geometry and its uncertainty: A review, *Int. J. Comput. Vis.*, Vol.27, No.2, pp.161–195 (1998).
27) Zhang, Z., Loop, C.: Estimating the fundamental matrix by transforming image points in projective space, *Comput. Vis. Image Understand.*, Vol.82, No.2, pp.174–180 (2001).

Appendix

A.1 Least Squares

We minimize

$$J_{\rm LS} = \sum_{\alpha=1}^{N} (\boldsymbol{u}, \boldsymbol{\xi}_\alpha)^2 = (\boldsymbol{u}, \boldsymbol{M}_{\rm LS}\boldsymbol{u}), \tag{44}$$

where we define

$$\boldsymbol{M}_{\rm LS} = \sum_{\alpha=1}^{N} \boldsymbol{\xi}_\alpha \boldsymbol{\xi}_\alpha^\top. \tag{45}$$

The function $J_{\rm LS}$ can be minimized by the unit eigenvector of the 9×9 matrix $\boldsymbol{M}_{\rm LS}$ for the smallest eigenvalue. Alternatively, we can compute the SVD of the $9 \times N$ matrix $(\boldsymbol{\xi}_1 \cdots \boldsymbol{\xi}_N)$.

A.2 Taubin Method

We approximate the denominators in Eq. (35) by their average. Disregarding the constant multiplier, which does not affect the minimization, we let

$$J_{\rm TB} = \frac{\sum_{\alpha=1}^{N} (\boldsymbol{u}, \boldsymbol{\xi}_\alpha)^2}{\sum_{\alpha=1}^{N} (\boldsymbol{u}, V_0[\boldsymbol{\xi}_\alpha]\boldsymbol{u})} = \frac{(\boldsymbol{u}, \boldsymbol{M}_{\rm LS}\boldsymbol{u})}{(\boldsymbol{u}, \boldsymbol{N}_{\rm TB}\boldsymbol{u})}, \tag{46}$$

where we define

$$\boldsymbol{N}_{\rm TB} \equiv \sum_{\alpha=1}^{N} V_0[\boldsymbol{\xi}_\alpha]. \tag{47}$$

The function $J_{\rm TB}$ is minimized by solving the generalized eigenvalue problem

$$\boldsymbol{M}_{\rm LS}\boldsymbol{u} = \lambda \boldsymbol{N}_{\rm TB}\boldsymbol{u} \tag{48}$$

for the smallest generalized eigenvalue. However, we cannot directly solve this, because $\boldsymbol{N}_{\rm TB}$ is not positive definite. So, we decompose $\boldsymbol{\xi}_\alpha$, $\boldsymbol{u}$, and $V_0[\boldsymbol{\xi}_\alpha]$ in the form

$$\boldsymbol{\xi}_\alpha = \begin{pmatrix} \boldsymbol{z}_\alpha \\ f_0^2 \end{pmatrix}, \quad \boldsymbol{u} = \begin{pmatrix} \boldsymbol{v} \\ F_{33} \end{pmatrix}, \quad V_0[\boldsymbol{\xi}_\alpha] = \begin{pmatrix} V_0[\boldsymbol{z}_\alpha] & \boldsymbol{0} \\ \boldsymbol{0}^\top & 0 \end{pmatrix}. \tag{49}$$

and define 8×8 matrices $\tilde{\boldsymbol{M}}_{\rm TB}$ and $\tilde{\boldsymbol{N}}_{\rm TB}$

$$\tilde{\boldsymbol{M}}_{\rm TB} = \sum_{\alpha=1}^{N} \tilde{\boldsymbol{z}}_\alpha \tilde{\boldsymbol{z}}_\alpha^\top, \qquad \tilde{\boldsymbol{N}}_{\rm TB} = \sum_{\alpha=1}^{N} V_0[\boldsymbol{z}_\alpha], \tag{50}$$

where

$$\tilde{\boldsymbol{z}}_\alpha = \boldsymbol{z}_\alpha - \bar{\boldsymbol{z}}, \qquad \bar{\boldsymbol{z}} = \frac{1}{N}\sum_{\alpha=1}^{N} \boldsymbol{z}_\alpha. \tag{51}$$

Then, Eq. (48) splits into two equations

$$\tilde{\boldsymbol{M}}_{\rm TB}\boldsymbol{v} = \lambda \tilde{\boldsymbol{N}}_{\rm TB}\boldsymbol{v}, \qquad (\boldsymbol{v}, \bar{\boldsymbol{z}}) + f_0^2 F_{33} = 0. \tag{52}$$

We compute the unit generalized eigenvector $\boldsymbol{v}$ of the first equation for the smallest generalized eigenvalue λ. The second equation gives F_{33}, and $\boldsymbol{u}$ is given by

$$\boldsymbol{u} = \mathcal{N}\Big[\begin{pmatrix} \boldsymbol{v} \\ F_{33} \end{pmatrix}\Big]. \tag{53}$$

The idea of replacing the denominator of each summand by the average of all the denominators originates from the paper by Taubin[24], hence the name of this method. However, a similar idea was already presented in 1990s by Tagawa et al.[21)–23)] in relation to 3-D interpretation of optical flow.

A.3 Derivation of Eqs. (20)

Introducing Lagrange multipliers λ_α, μ_α, and μ'_α for the constraints of Eqs. (18) and (19) to Eq. (16), we let the derivatives of

$$\sum_{\alpha=1}^{N}\Big(\|\Delta\boldsymbol{x}_\alpha\|^2 + \|\Delta\boldsymbol{x}'_\alpha\|^2\Big) - \sum_{\alpha=1}^{N}\lambda_\alpha\Big((\boldsymbol{F}\boldsymbol{x}'_\alpha, \Delta\boldsymbol{x}_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \Delta\boldsymbol{x}'_\alpha) - (\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}_\alpha)\Big) - \sum_{\alpha=1}^{N}\mu_\alpha(\boldsymbol{k}, \Delta\boldsymbol{x}_\alpha) - \sum_{\alpha=1}^{N}\mu'_\alpha(\boldsymbol{k}, \Delta\boldsymbol{x}'_\alpha), \tag{54}$$

with respect to $\Delta\boldsymbol{x}_\alpha$ and $\Delta\boldsymbol{x}'_\alpha$ be $\boldsymbol{0}$, resulting in

$$2\Delta\boldsymbol{x}_\alpha - \lambda_\alpha\boldsymbol{F}\boldsymbol{x}'_\alpha - \mu_\alpha\boldsymbol{k} = \boldsymbol{0}, \quad 2\Delta\boldsymbol{x}'_\alpha - \lambda_\alpha\boldsymbol{F}^\top\boldsymbol{x}_\alpha - \mu'_\alpha\boldsymbol{k} = \boldsymbol{0}. \tag{55}$$

Multiplying the projection matrix $\boldsymbol{P}_{\mathbf{k}}$ in Eq. (21) on both sides from left and noting that $\boldsymbol{P}_{\mathbf{k}}\Delta\boldsymbol{x}_\alpha = \Delta\boldsymbol{x}_\alpha$, $\boldsymbol{P}_{\mathbf{k}}\Delta\boldsymbol{x}'_\alpha = \Delta\boldsymbol{x}'_\alpha$, and $\boldsymbol{P}_{\mathbf{k}}\boldsymbol{k} = \boldsymbol{0}$, we have

$$2\Delta\boldsymbol{x}_\alpha - \lambda_\alpha\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha = \boldsymbol{0}, \quad 2\Delta\boldsymbol{x}'_\alpha - \lambda_\alpha\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha = \boldsymbol{0}. \tag{56}$$

Hence, we obtain

$$\Delta\boldsymbol{x}_\alpha = \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha, \quad \Delta\boldsymbol{x}'_\alpha = \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha. \tag{57}$$

Substituting these into (18), we have

$$(\boldsymbol{F}\boldsymbol{x}'_\alpha, \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha) = (\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha), \tag{58}$$

and hence

$$\frac{\lambda_\alpha}{2} = \frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)}{(\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)}. \tag{59}$$

Substituting this into Eqs. (57), we obtain Eqs. (20).

A.4 Derivation of Eq. (22)

If we substitute Eqs. (20) into Eq. (16), the reprojection error E becomes

$$\begin{aligned}
&\sum_{\alpha=1}^{N}\Big(\Big\|\frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha}{(\boldsymbol{x}'_\alpha, \boldsymbol{F}^\top\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)}\Big\|^2 + \Big\|\frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha}{(\boldsymbol{x}'_\alpha, \boldsymbol{F}^\top\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)}\Big\|^2\Big) \\
&= \sum_{\alpha=1}^{N}\frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)^2(\|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha\|^2 + \|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha\|^2)}{\Big((\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)\Big)^2} \\
&= \sum_{\alpha=1}^{N}\frac{(\boldsymbol{x}_\alpha, \boldsymbol{F}\boldsymbol{x}'_\alpha)^2}{(\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) + (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)},
\end{aligned} \tag{60}$$

where we have noted that due to the identity $\boldsymbol{P}_{\mathbf{k}}^2 = \boldsymbol{P}_{\mathbf{k}}$ we have $\|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha\|^2 = (\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha) = (\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}^2\boldsymbol{F}\boldsymbol{x}'_\alpha) = (\boldsymbol{F}\boldsymbol{x}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\boldsymbol{x}'_\alpha)$. Similarly, we have $\|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha\|^2 = (\boldsymbol{F}^\top\boldsymbol{x}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\boldsymbol{x}_\alpha)$.

A.5 Derivation of Eqs. (30)

Introducing Lagrange multipliers λ_α, μ_α, and μ'_α for the constraints of Eqs. (28) and (29) to Eq. (25), we let the derivatives of

$$\sum_{\alpha=1}^{N}\Big(\|\tilde{\boldsymbol{x}}_\alpha + \Delta\hat{\boldsymbol{x}}_\alpha\|^2 + \|\tilde{\boldsymbol{x}}'_\alpha + \Delta\hat{\boldsymbol{x}}'_\alpha\|^2\Big) - \sum_{\alpha=1}^{N}\lambda_\alpha\Big((\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \Delta\hat{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \Delta\hat{\boldsymbol{x}}'_\alpha) - (\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha)\Big) - \sum_{\alpha=1}^{N}\mu_\alpha(\boldsymbol{k}, \Delta\hat{\boldsymbol{x}}_\alpha) - \sum_{\alpha=1}^{N}\mu'_\alpha(\boldsymbol{k}, \Delta\hat{\boldsymbol{x}}'_\alpha), \tag{61}$$

with respect to $\Delta\hat{\boldsymbol{x}}_\alpha$ and $\Delta\hat{\boldsymbol{x}}'_\alpha$ be $\boldsymbol{0}$, resulting in

$$\begin{aligned}
&2(\tilde{\boldsymbol{x}}_\alpha + \Delta\hat{\boldsymbol{x}}_\alpha) - \lambda_\alpha\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha - \mu_\alpha\boldsymbol{k} = \boldsymbol{0}, \\
&2(\tilde{\boldsymbol{x}}'_\alpha + \Delta\hat{\boldsymbol{x}}'_\alpha) - \lambda_\alpha\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha - \mu'_\alpha\boldsymbol{k} = \boldsymbol{0}.
\end{aligned} \tag{62}$$

Multiplying $\boldsymbol{P}_{\mathbf{k}}$ on both sides from left, we have

$$2\tilde{\boldsymbol{x}}_\alpha + 2\Delta\hat{\boldsymbol{x}}_\alpha - \lambda_\alpha\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha = \boldsymbol{0}, \quad 2\tilde{\boldsymbol{x}}_\alpha + 2\Delta\hat{\boldsymbol{x}}'_\alpha - \lambda_\alpha\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha = \boldsymbol{0}. \tag{63}$$

Hence, we have

$$\Delta\hat{\boldsymbol{x}}_\alpha = \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha - \tilde{\boldsymbol{x}}_\alpha, \qquad \Delta\hat{\boldsymbol{x}}'_\alpha = \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha - \tilde{\boldsymbol{x}}'_\alpha. \tag{64}$$

Substituting these into (27), we obtain

$$(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha - \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \frac{\lambda_\alpha}{2}\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha - \tilde{\boldsymbol{x}}'_\alpha) = (\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha), \tag{65}$$

and hence

$$\frac{\lambda_\alpha}{2} = \frac{(\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)}. \tag{66}$$

Substituting this into Eqs. (64), we obtain Eqs. (30).

A.6 Derivation of Eq. (31)

If we substitute Eqs. (30) into Eq. (25), the reprojection error E becomes

$$\begin{aligned}
&\sum_{\alpha=1}^{N}\Bigg(\Big\|\frac{\big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\big)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)}\Big\|^2 \\
&\quad + \Big\|\frac{\big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\big)\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)}\Big\|^2\Bigg) \\
&= \sum_{\alpha=1}^{N}\Big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\Big)^2\Big(\|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha\|^2 \\
&\quad + \|\boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha\|^2\Big)\Big/\Big((\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)\Big)^2 \\
&= \sum_{\alpha=1}^{N}\frac{\big((\hat{\boldsymbol{x}}_\alpha, \boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \tilde{\boldsymbol{x}}_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \tilde{\boldsymbol{x}}'_\alpha)\big)^2}{(\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}\hat{\boldsymbol{x}}'_\alpha) + (\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha, \boldsymbol{P}_{\mathbf{k}}\boldsymbol{F}^\top\hat{\boldsymbol{x}}_\alpha)}.
\end{aligned} \tag{67}$$

A.7 FNS

The *FNS* (*Fundamental Numerical Scheme*) of Chojnacki et al.[3] is based on the fact that the derivative of Eq. (35) with respect to $\boldsymbol{u}$ has the form

$$\nabla_{\mathbf{u}}E = \frac{2}{f_0^2}\boldsymbol{X}\boldsymbol{u}, \tag{68}$$

where $\boldsymbol{X}$ is the 9×9 matrix in Eqs. (12). The FNS solves

$$\boldsymbol{X}\boldsymbol{u} = \boldsymbol{0}. \tag{69}$$

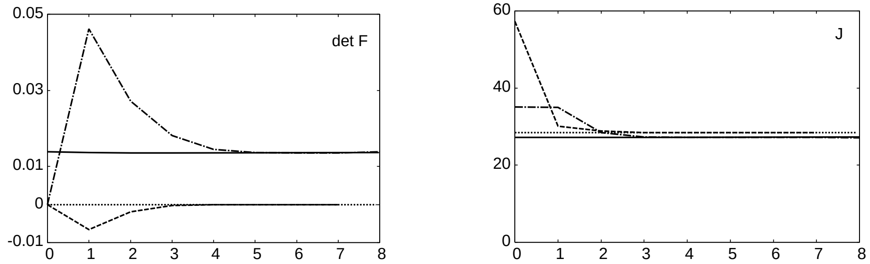


Fig. 3 The convergence of $\det\boldsymbol{F}$ and the residual J by CFNS for different initializations (σ = 1): LS (solid lines), SVD-corrected LS (dashed lines), and the true value (chained lines). All solutions are SVD-corrected in the final step.

by the following iterations:

(1) Initialize $\boldsymbol{u}$.

(2) Compute the 9×9 matrix $\boldsymbol{X}$ in Eqs. (12).

(3) Solve the eigenvalue problem

$$\boldsymbol{X}\boldsymbol{u}' = \lambda\boldsymbol{u}', \tag{70}$$

and compute the unit eigenvector $\boldsymbol{u}'$ for the smallest eigenvalue λ.

(4) If $\boldsymbol{u}' \approx \boldsymbol{u}$ up to sign, return $\boldsymbol{u}'$ and stop. Else, let $\boldsymbol{u} \leftarrow \boldsymbol{u}'$ and go back to Step (2).

A.8 Convergence of CFNS

In short, the *CFNS* (*Constrained Fundamental Numerical Scheme*) of Chojnacki et al.[4] is a method that imposes a *penalty* for $\det\boldsymbol{F} \neq 0$ in the course of the FNS iterations, while our EFNS *projects* the intermediate solution onto the subspace of $\det\boldsymbol{F} = 0$ in the course of the FNS iterations.

If we use CFNS to the example in Fig. 2(a), we obtain Fig. 3, which shows a typical instance ($\sigma = 1$) of the convergence of $\det\boldsymbol{F}$ and the Sampson error residual J ($= f_0^2E$) from different initial values: least-squares (LS) solution (solid lines), the SVD-corrected (i.e. forced to have determinant 0 via SVD) LS solution, and the true value (chained lines). In the end, the solutions are corrected to have determinant 0 via SVD, as prescribed by Chojnacki et al.[4]. The dotted lines show the values expected to converged to.

The LS solution has an excessively low residual J, since the rank constraint $\det\boldsymbol{F} = 0$ is ignored. So, J needs to be *increased* to achieve $\det\boldsymbol{F} = 0$, but

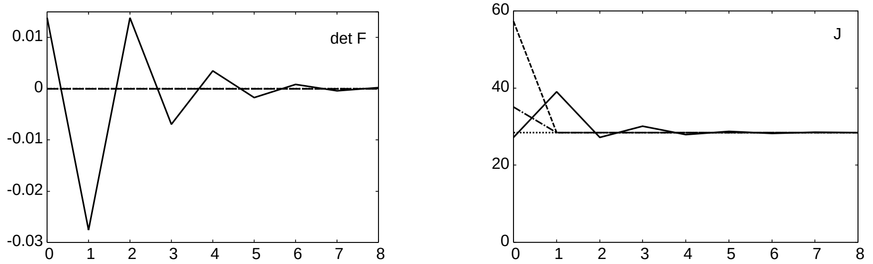


Fig. 4 The corresponding results by EFNS.

CFNS fails to increase J. As a result, $\det \boldsymbol{F}$ remains nonzero and drops to 0 by the final SVD correction, causing a sudden upward jump in J. If we start from SVD-corrected LS, the residual J first increases, making $\det \boldsymbol{F}$ nonzero, but in the end both J and $\det \boldsymbol{F}$ converge in an expected way. In contrast, the true value has a very large J, so CFNS tries to decrease it at the cost of an increase in $\det \boldsymbol{F}$, which never reverts to 0 until the final SVD. Figure 4 shows corresponding results by EFNS. Both J and $\det \boldsymbol{F}$ converge to their correct values with stably attenuating oscillations.

We mean by "convergence" the state of the same solution repeating itself in the course of iterations. In mathematical terms, the resulting solution is a *fixed point* of the iteration operator. Chojnacki et al.[4] proved that their solution is a fixed point of CFNS. They expected to arrive at that solution by their scheme. As demonstrated in Fig. 3, however, CFNS has many other fixed points, and which to arrive at depends on initialization. In contrast, any fixed point of EFNS is *necessarily* the desired solution, as proved in Sec. 3.2.2.

A.9 Optimal Correction

The fundamental matrix $\boldsymbol{F}$ computed without considering the rank constraint is moved in the statistically mostly likely direction until it satisfies the rank constraint. The procedure goes as follows:

(1) Compute the 9×9 matrix $\boldsymbol{M}$ in Eqs. (10) and the 9×9 matrix $V_0[\boldsymbol{u}]$ by

$$V_0[\boldsymbol{u}] = \boldsymbol{M}_8^-, \tag{71}$$

where $(\cdot)_r^-$ denotes pseudoinverse constrained to have rank r.

(2) Update $\boldsymbol{u}$ as follows:

$$\boldsymbol{u} \leftarrow \mathcal{N}[\boldsymbol{u} - \frac{1}{3}\frac{(\boldsymbol{u}, \boldsymbol{u}^\dagger)V_0[\boldsymbol{u}]\boldsymbol{u}^\dagger}{(\boldsymbol{u}^\dagger, V_0[\boldsymbol{u}]\boldsymbol{u}^\dagger)}]. \tag{72}$$

(3) If $(\boldsymbol{u}, \boldsymbol{u}^\dagger) \approx 0$, return $\boldsymbol{u}$ and stop. Else, update the matrix $V_0[\boldsymbol{u}]$ in the following form and go back to Step (3):

$$V_0[\boldsymbol{u}] \leftarrow \boldsymbol{P}_{\boldsymbol{u}} V_0[\boldsymbol{u}] \boldsymbol{P}_{\boldsymbol{u}}, \tag{73}$$

Recall that $\boldsymbol{u}^\dagger$ is the cofactor vector of $\boldsymbol{u}$. Note that $\det \boldsymbol{F} = (\boldsymbol{u}, \boldsymbol{u}^\dagger)/3$, which is a direct consequence of the identity $\boldsymbol{F}^\dagger \boldsymbol{F} = (\det \boldsymbol{F})\boldsymbol{I}$.

(Received March 25, 2009)
(Revised October 6, 2009)
(Accepted October 7, 2009)
(Released March 11, 2010)

(Communicated by *Fay Huang*)

Kenichi Kanatani received his B.E., M.S., and Ph.D. in applied mathematics from the University of Tokyo in 1972, 1974 and 1979, respectively. After serving as Professor of computer science at Gunma University, Gunma, Japan, he is currently Professor of computer science at Okayama University, Okayama, Japan. He is the author of many books on computer vision and received many awards including the best paper awards from IPSJ (1987) and IEICE (2005). He is an IEEE Fellow.

Yasuyuki Sugaya received his B.E., M.S., and Ph.D. in computer science from the University of Tsukuba, Ibaraki, Japan, in 1996, 1998, and 2001, respectively. From 2001 to 2006, he was Assistant Professor of computer science at Okayama University, Okayama, Japan. Currently, he is Associate Professor of information and computer sciences at Toyohashi University of Technology, Toyohashi, Aichi, Japan. His research interests include image processing and computer vision. He received the IEICE best paper award in 2005.