Metric Self Calibration from Screw-Transform Manifolds

Russell A. Manning Charles R. Dyer

Department of Computer Sciences
University of Wisconsin
Madison, Wisconsin 53706

Abstract

This paper introduces a method for metric self-calibration that is based on a novel decomposition of the fundamental matrix between two views taken by a camera with fixed internal parameters. The method blends important advantages of the Kruppa constraints and the modulus constraint: it works directly from fundamental matrices and uses a reduced-parameter representation for stability. General properties of the new decomposition are also developed, including an intuitive interpretation of the three free parameters of internal calibration. The approach is demonstrated on both real and synthetic data.

1. Introduction

It is common for a video camera used in machine-vision applications to have unvarying internal calibration, especially over short periods of time. As such a camera is moved around a scene, any two positions of the camera can be related by a *screw transformation*. This means that, given any two positions, the camera in the first position can be rotated around some fixed axis in space and then translated parallel to the axis to end up in the second position. Using this observation, we introduce a new representation of the fundamental matrix between any two views taken by an internally-fixed camera. The new representation, a more general form of an equation given in [3], decomposes the fundamental matrix into terms involving the angle of rotation around the screw axis, the amount of translation parallel to the screw axis, and three vectors related to the internal calibration of the camera.

The new decomposition makes it possible to use the fundamental matrix between two views to determine information about the screw transformation and the camera calibration, which is useful because the fundamental matrix can be determined directly from point correspondences (or by other means, like optical flow or identification of planar homographies, when reliable point correspondences are not available). We develop some of the important properties of the decomposition and, in particular, demonstrate a novel method for camera self calibration.

The support of the National Science Foundation under grant IIS-9988426 is gratefully acknowledged.

Camera self calibration is the process of finding the internal parameters of a camera (as they would be measured in a Euclidean coordinate system up to an overall scaling factor) directly from views taken by the camera, without any knowledge of scene geometry. Faugeras et al. [2, 6] introduced an important, simple, general-purpose method for self calibration based on the Kruppa constraints. Besides simplicity, the Kruppa method has two major advantages: (1) it works directly from fundamental matrices, and (2) because of this, only a small number of the camera views actually need to overlap (since a fundamental matrix can be determined from any two overlapping views and only three fundamental matrices are required by the method).

More recently, Pollefeys [8, 9] has developed an alternative and apparently more robust self-calibration method based on his *modulus constraint*. Pollefeys' method follows a striated approach to metric self calibration, first creating a projective scene reconstruction, then improving this to an affine reconstruction before finally determining metric calibration and reconstruction. The apparent stability of Pollefeys' method comes from reducing the number of free parameters during the affine reconstruction step. When finding the plane at infinity to upgrade from projective to affine reconstruction, there are naively three free parameters, but Pollefeys demonstrates that there are actually only two free parameters because of the modulus constraint. In the Kruppa method, where metric calibration is determined in one nonlinear minimization step, there are five free parameters and thus more flexibility for the solution to erroneously fit noisy data. A serious drawback to Pollefeys' method is the requirement that all cameras be put in the same projective basis as an initial step. In practice, this is usually achieved by finding a set of feature points that are either visible in all views or can be projectively transferred between all views (see description in [7]). The Kruppa method works directly from fundamental matrices and so is applicable whenever fundamental matrices can be determined, even if a common projective basis cannot be found.

Our approach to metric self calibration has the strengths of both of these methods: it works directly from fundamental matrices (giving it the advantages of the Kruppa method) and it reduces the number of free parameters from five in the Kruppa method down to three, thus improving numerical

$$
\begin{aligned}
\mathbf{F} &= \mathbf{F}^{-} + \mathbf{F}^{+} && (1)\\
\mathbf{F}^{-} &= [\sin\theta\mathbf{h}_2 + \gamma\cos\theta\mathbf{h}_3]_\times && (2)\\
\mathbf{F}^{+} &= \frac{1-\cos\theta}{|\mathbf{H}|}\left[(\mathbf{h}_1\times\mathbf{h}_3)(\mathbf{h}_1\times\mathbf{h}_2)^\top + (\mathbf{h}_1\times\mathbf{h}_2)(\mathbf{h}_1\times\mathbf{h}_3)^\top\right] + \\
&\quad \frac{\gamma\sin\theta}{|\mathbf{H}|}\left[(\mathbf{h}_1\times\mathbf{h}_3)(\mathbf{h}_1\times\mathbf{h}_3)^\top + (\mathbf{h}_2\times\mathbf{h}_3)(\mathbf{h}_2\times\mathbf{h}_3)^\top\right] && (3)
\end{aligned}
$$

Figure 1: Decomposition of the fundamental matrix based on screw transforms.

stability as in Pollefeys' method. Our method has several other strengths which will be highlighted later as they arise during the mathematical development of the algorithm. Finally, it should be mentioned that there exist several other methods for self calibration from general camera motions (e.g., [10]), each with their own strengths and weaknesses, but space prevents us from describing them further. The mathematics underlying our method is distinctly different from the mathematics underlying other methods, giving our algorithm a combination of beneficial properties not shared by any other method.

2. Rising turntable formulation of the fundamental matrix

If two identical cameras are placed at different positions and orientations anywhere in space, then there exists a unique screw transformation that will take the first camera and make it exactly overlap the second camera [1]. A *screw transformation* is defined as a single rotation around a fixed axis in space followed by a single translation parallel to the axis. Because the two cameras are related by a screw transformation, we could alternatively think of the views as being captured under a *fixed-camera formulation* of the problem. In this formulation, only the first camera is used, fixed in position and orientation: The single camera views the scene and the scene is interpreted as moving on a *rising turntable* whose rotation axis matches the screw transform axis and which undergoes an equal and opposite screw transformation to the original.

In this section we develop a formula for the fundamental matrix between two views of a rising turntable taken by a fixed camera. We choose the world coordinate system so that the rotation axis coincides with the z-axis and the first camera is located at $(1,0,0)^\top$. This makes the two camera matrices

$$
\begin{aligned}
\mathbf{\Pi}_A &= \mathbf{K}\,\mathbf{R}\left[\begin{array}{ccc|c} 1 & 0 & 0 & -1\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0 \end{array}\right]\\
\mathbf{\Pi}_B &= \mathbf{\Pi}_A\mathbf{S}(-\gamma,-\theta)
\end{aligned}
$$

where $\mathbf{K}$ is an upper triangular matrix representing the fixed internal calibration of the camera and $\mathbf{R}$ is a rotation matrix giving the fixed tilt of the camera relative to the world coordinate system. The remaining matrix is defined by

$$
\mathbf{S}(\gamma,\theta) = \left[\begin{array}{ccc|c} \cos\theta & -\sin\theta & 0 & 0\\ \sin\theta & \cos\theta & 0 & 0\\ 0 & 0 & 1 & \gamma\\ 0 & 0 & 0 & 1 \end{array}\right]
$$

This matrix transforms the coordinates of points in the scene before projection into the fixed camera, thus producing the motion of the rising turntable. Throughout the paper, we will need to reference the column vectors of the two camera matrices, which we will do using the representation $\mathbf{\Pi}_A = [\,\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3 \mid \mathbf{h}_4\,]$ and $\mathbf{\Pi}_B = [\,\mathbf{g}_1\ \mathbf{g}_2\ \mathbf{g}_3 \mid \mathbf{g}_4\,]$ where $\mathbf{h}_i,\ \mathbf{g}_i \in \Re^3$. Notice that $\mathbf{h}_4 = -\mathbf{h}_1$, which is used below in finding the epipole.

Using the camera matrices $\mathbf{\Pi}_A$ and $\mathbf{\Pi}_B$, we can write the fundamental matrix between the two views as

$$
\begin{aligned}
\mathbf{F} &= [-\mathbf{e}_B]_\times(\mathbf{H}^\infty) && (4)\\
&= [(1-\cos\theta)\mathbf{h}_1 + \sin\theta\mathbf{h}_2 + \gamma\mathbf{h}_3]_\times(\mathbf{H}^\infty) && (5)
\end{aligned}
$$

where

$$
\mathbf{H}^\infty = [\,\mathbf{g}_1\ \mathbf{g}_2\ \mathbf{g}_3\,][\,\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3\,]^{-1} \qquad (6)
$$

and $\mathbf{e}_B$ is the epipole as it appears in the second camera, given by $\mathbf{e}_B = \mathbf{\Pi}_B(1,0,0,1)^\top = \mathbf{\Pi}_A(\cos\theta,-\sin\theta,-\gamma,1)^\top$. Eq. 4 appears frequently (e.g., [4]) and $\mathbf{H}^\infty$ is discussed further in Section 3.2.

One of the main goals of this paper is to present the alternative formulation of the fundamental matrix shown in Fig. 1 (generalized from an equation in [3]). We prove that Eq. 4 and Eq. 1 represent the same function as follows: Since $\mathbf{F}$ is a matrix and all matrices represent linear transformations, it only needs to be shown that the expressions in Eq. 4 and Eq. 1 act the same way on three basis vectors spanning $\Re^3$ (in which case they will act the same way on all linear combinations of those basis vectors and thus on all elements of $\Re^3$). $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{h}_3$ are linearly independent (for any nondegenerate camera matrix) and thus form a basis for $\Re^3$. $\mathbf{h}_1$ is the direction of the x-axis as represented in camera A's coordinates (alternatively, $\mathbf{h}_1$ is the image of the vanishing point of the x-axis as seen in camera A).

As such, $\mathbf{h}_1 = \mathbf{\Pi}_A(1,0,0,0)^\top$. $\mathbf{H}^\infty$ transforms the representation of directions (homogeneous vectors with fourth coordinate 0) from camera basis A to camera basis B, so $\mathbf{H}^\infty \mathbf{h}_1 = \mathbf{\Pi}_B(1,0,0,0)^\top$ (this is also clear just by looking at the definition of $\mathbf{H}^\infty$ in Eq. 6). Applying this same reasoning to $\mathbf{h}_2$ and $\mathbf{h}_3$ yields:

$$\mathbf{H}^\infty \mathbf{h}_1 = \mathbf{\Pi}_B(1,0,0,0)^\top = \cos\theta \mathbf{h}_1 - \sin\theta \mathbf{h}_2 \quad (7)$$
$$\mathbf{H}^\infty \mathbf{h}_2 = \mathbf{\Pi}_B(0,1,0,0)^\top = \sin\theta \mathbf{h}_1 + \cos\theta \mathbf{h}_2 \quad (8)$$
$$\mathbf{H}^\infty \mathbf{h}_3 = \mathbf{\Pi}_B(0,0,1,0)^\top = \mathbf{h}_3 \quad (9)$$

It is now a straight-forward exercise to multiply both versions of the fundamental matrix by $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{h}_3$ and show equality of the results in each case.

3. Self calibration

Surprisingly, it is possible to find metric camera calibration (i.e., to find the matrix $\mathbf{K}$ up to a scale factor) directly from camera views without knowledge of scene measurements [2]. In this section, we develop some useful properties of Eq. 1 that lead to a new method for self calibration.

Let $\mathbf{H} = \mathbf{KR} = [\mathbf{h}_1 \mathbf{h}_2 \mathbf{h}_3]$ and notice that

$$\mathbf{HH}^\top = \mathbf{KRR}^\top \mathbf{K}^\top = \mathbf{KK}^\top \quad (10)$$

since rotation matrices are unitary. The matrix $\mathbf{KK}^\top$ is called the *dual image of the absolute conic* and is denoted ω^*. Thus if $\mathbf{H}$ can be found, then ω^* can be found and $\mathbf{K}$ can be determined by Cholesky factorization of ω^*. Our goal becomes finding $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{h}_3$, which is where Eq. 1 gets utilized.

3.1. Parameterizing the projected axes of world coordinates

The fundamental matrix $\mathbf{F}$ between views A and B can be found directly from the images themselves by identifying point correspondences or by other means. Once $\mathbf{F}$ has been found, the relationship in Eq. 1 puts constraints on $\mathbf{H}$, θ, and γ. Assume $\mathbf{\Pi}_A$ has been scaled so that $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{h}_3$ exactly satisfy Eq. 1. Also assume $\|\mathbf{e}_B\| = \|\mathbf{e}_A\| = 1$.

$\mathbf{F}^+$ and $\mathbf{F}^-$ can be determined from $\mathbf{F}$ because

$$\mathbf{F}^+ = \frac{1}{2}(\mathbf{F} + \mathbf{F}^\top) \quad \text{and} \quad \mathbf{F}^- = \frac{1}{2}(\mathbf{F} - \mathbf{F}^\top) \quad (11)$$

Notice that $\mathbf{h}_3 \cdot \mathbf{F}^+\mathbf{h}_3 = 0$. This property allows the image of $\mathbf{h}_3$ to be parameterized by a single real variable κ. One way to do this is by expanding the equation $\mathbf{h}_3 \cdot \mathbf{F}^+\mathbf{h}_3 = 0$ and then using the quadratic equation. A more elegant parameterization is (see Appendix B)

$$\mathbf{h}_3 \cong (\kappa\mathbf{I} - \mathbf{M})^{-1}\mathbf{e}_B \quad (12)$$

where $\mathbf{M}$ is any (invertible) matrix satisfying $\mathbf{F} = [\mathbf{e}_B]_\times \mathbf{M}$. The notation $\cong$ indicates equality up to a scale. Once $\mathbf{h}_3$ is known up to a scale then $\mathbf{h}_1$ can be found via

$$\mathbf{h}_1 \cong (\mathbf{F}^+\mathbf{m}) \times (\mathbf{F}^+\mathbf{h}_3) \quad (13)$$

where $\mathbf{m}$ is defined by $\mathbf{F}^- = [\mathbf{m}]_\times$, making $\mathbf{m} = (\mathbf{F}^-_{(32)}, \mathbf{F}^-_{(13)}, \mathbf{F}^-_{(12)})^\top$. The notation $\mathbf{M}_{(ij)}$ indicates the entry of matrix $\mathbf{M}$ in row i and column j. Since we will need to refer to the exact scaling factors that make Eq. 12 and Eq. 13 equalities, we define the quantities s_1, s_3, $\underline{\mathbf{h}}_1$, and $\underline{\mathbf{h}}_3$ by

$$s_3\mathbf{h}_3 = \underline{\mathbf{h}}_3 = (\kappa\mathbf{I} - \mathbf{M})^{-1}\mathbf{e}_B$$
$$s_1\mathbf{h}_1 = \underline{\mathbf{h}}_1 = (\mathbf{F}^+\mathbf{m}) \times (\mathbf{F}^+\underline{\mathbf{h}}_3)$$

To summarize so far: If $\mathbf{F}$ and κ are known, $\mathbf{M}$ can be chosen using $\mathbf{F}$ and then $\underline{\mathbf{h}}_1$ and $\underline{\mathbf{h}}_3$ can be found. If, in addition, θ is known, then $\mathbf{h}_2$ and s_1 can also be found as the following algorithm shows:

---

(1) Find the unique null eigenvector $(\sigma_1, \sigma_2)^\top$ of $\left[\ \mathbf{F}^+\underline{\mathbf{h}}_1 \quad -\mathbf{F}^-\underline{\mathbf{h}}_3\ \right]$. This eigenvector is $(1/s_1, \gamma/s_3)^\top$ up to an unknown scale factor ϕ.

(2) Since $\mathbf{F}^+\mathbf{h}_3 = (1-\cos\theta)(\mathbf{h}_1 \times \mathbf{h}_3)$, the following overdetermined system can be solved for ϕ (note $\phi\mathbf{h}_1 = \sigma_1\underline{\mathbf{h}}_1$):

$$\phi\mathbf{F}^+\underline{\mathbf{h}}_3 = \sigma_1(1-\cos\theta)(\underline{\mathbf{h}}_1 \times \underline{\mathbf{h}}_3)$$

(3) Find $\phi\mathbf{h}_2$ using

$$\phi\mathbf{h}_2 = \frac{1}{\sin\theta}(\phi\mathbf{m} - \sigma_2\cos\theta\underline{\mathbf{h}}_3)$$

(Remarks) Using Eq. 3 and Eq. 2, step (1) comes from $\mathbf{F}^+\mathbf{h}_1 = \gamma\sin\theta(\mathbf{h}_2 \times \mathbf{h}_3)$ and $\mathbf{F}^-\mathbf{h}_3 = \sin\theta(\mathbf{h}_2 \times \mathbf{h}_3)$. Since the matrix has two columns, its rank is 1 or 2; its rank must be at least 1 (nonzero matrix) and is clearly less than 2. Step (3) comes from the definition of $\mathbf{m}$ and Eq. 2: $\mathbf{m} = \sin\theta\mathbf{h}_2 + \gamma\cos\theta\mathbf{h}_3$. Also note $\sigma_2\underline{\mathbf{h}}_3 = \phi\gamma\mathbf{h}_3$.

In summary, $\mathbf{h}_1$, $\mathbf{h}_2$, and $\underline{\mathbf{h}}_3$, which are the vanishing points of the x, y, and z-axes as seen in the first camera view, respectively, can be determined directly from $\mathbf{F}$ provided two real parameters κ and θ are known. Furthermore, by the method just described, $\mathbf{H}$ can be determined up to a single unknown real parameter: the scale of $\mathbf{h}_3$, which is γ. Once $\mathbf{H}$ is determined, the metric internal calibration of the camera can be found and metric scene reconstruction is possible. To complete the algorithm for finding $\mathbf{H}$ from κ, θ, and γ, add the following step to the previous algorithm:

---

(4) Use γ (and ϕ) to extract s_3 from the eigenvector in step (1). Then $\mathbf{h}_3 = \underline{\mathbf{h}}_3/s_3$.

Naively, since $\mathbf{K}$ is an upper triangular matrix and we are only interested in $\mathbf{K}$ up to a scale factor, we know $\mathbf{K}$ has at most 5 degrees of freedom. Our analysis shows that

Figure 2: 6×5 matrix and corresponding null eigenvector used when finding $\mathbf{H}^\infty$.

$$\begin{bmatrix} \mathbf{M}_{(11)} + \mathbf{M}_{(22)} + \mathbf{M}_{(33)} & \mathbf{e}_B^\top & -(1+2\cos\theta) \\ \mathbf{M}\underline{\mathbf{h}}_3 & \mathbf{e}_B\underline{\mathbf{h}}_3^\top & -\underline{\mathbf{h}}_3 \\ (\mathbf{l}_{12})_x\mathbf{q}_y - (\mathbf{l}_{12})_y\mathbf{q}_x & (-(\mathbf{l}_{12})_y\xi, (\mathbf{l}_{12})_x\xi, 0) & 0 \\ (\mathbf{l}_{12})_x\mathbf{q}_z - (\mathbf{l}_{12})_z\mathbf{q}_x & (-(\mathbf{l}_{12})_z\xi, 0, (\mathbf{l}_{12})_x\xi) & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \lambda\mathbf{a} \\ 1 \end{bmatrix} = \mathbf{0}$$

$\mathbf{K}$ can be parameterized by three real numbers κ, θ, and γ. The fact that $\mathbf{K}$ has only three degrees of freedom has been shown before (e.g., [7]). Here we have demonstrated a specific parameterization, one which has a great deal of intuitive meaning: θ is the rotation angle between the views, κ corresponds to the vanishing point of the rotation axis, and γ is the amount of translation along the screw axis during the screw transformation.

3.2. Additional properties of the rising-turntable formulation

The matrix introduced in Eq. 6 is called the *homography induced by the plane at infinity*. $\mathbf{H}^\infty$ transforms directions represented in camera A's coordinate system into directions represented in camera B's coordinate system. If $\mathbf{H}^\infty$ can be found, the cameras are said to be affinely calibrated and it is possible to perform affine scene reconstruction (scene reconstruction that is equivalent to metric reconstruction transformed by an invertible 3×3 matrix).

Consider the fixed-camera formulation of how the views were captured, in which the camera is fixed in position and the scene sits on a rising turntable. The vanishing point of the rotation axis, as seen in the first view, is given by $\mathbf{h}_3$. As the turntable rises, neither the rotation axis nor the fixed camera change and thus $\mathbf{h}_3$ always describes the vanishing point of the rotation axis as seen in the camera. Thus

$$(\mathbf{H}^\infty)\mathbf{h}_3 \cong \mathbf{h}_3 \tag{14}$$

Similarly, consider all the planes that are perpendicular to the rotation axis (in world coordinates). Just as parallel lines have a vanishing point, parallel planes have a vanishing line. As seen in camera A, the vanishing line for the planes that are parallel to the rotation axis will contain the vanishing points of both the x and y axes, which are given by $\mathbf{h}_1$ and $\mathbf{h}_2$. Thus we denote this vanishing line by $\mathbf{l}_{12}$ (here we use $\mathbf{l}_{12}$ to denote both the concept of this particular vanishing line and the specific representation of this line in view A). As with the vanishing point of the screw axis, $\mathbf{l}_{12}$ will project onto the same line in all camera views as the turntable rises. Thus

$$(\mathbf{H}^\infty)^\top \mathbf{l}_{12} \cong \mathbf{l}_{12} \tag{15}$$

(If $\mathbf{T_{ab}}$ is a 3×3 invertible matrix that transforms basis A into basis B and Ψ is a plane through the origin, then it is an easy-to-prove fact of linear algebra that $\mathbf{T_{ab}}^\top \mathbf{l}_b \cong \mathbf{l}_a$ where $\mathbf{l}_a$ denotes the vector perpendicular to Ψ in basis A and $\mathbf{l}_b$ denotes the vector perpendicular to Ψ in basis B.)

Eq. 14 and Eq. 15 state that $\mathbf{h}_3$ is an eigenvector of $\mathbf{H}^\infty$ and $\mathbf{l}_{12}$ is an eigenvector of $(\mathbf{H}^\infty)^\top$. Since $\mathbf{H}^\infty = \mathbf{KRK}^{-1}$, $\mathbf{H}^\infty$ is conjugate to a rotation matrix and thus has the same eigenvalues as $\mathbf{R}$. $\mathbf{R}$ has only one real eigenvalue, which is 1, and $\mathbf{h}_3$ is the eigenvector corresponding to this unique real eigenvalue.

Assuming $\gamma \neq 0$ (i.e., assuming the cameras are in general position), $\mathbf{H}^\infty$ can be found from $\mathbf{F}$ using only the two real numbers θ and κ (if they are known) as follows:

---

(1) Use κ and θ to find $\mathbf{h}_1$, $\mathbf{h}_2$, and $\underline{\mathbf{h}}_3$ as described in Section 3.1.

(2) Find the line $\mathbf{l}_{12}$ in image A that goes through $\mathbf{h}_1$ and $\mathbf{h}_2$:

$$\mathbf{l}_{12} = \frac{\mathbf{h}_1 \times \mathbf{h}_2}{||\mathbf{h}_1 \times \mathbf{h}_2||}$$

(3) Let $\xi = \mathbf{e}_B \cdot \mathbf{l}_{12}$. (If $\xi \ll 1$ then γ was too small and the views represent pure turntable motion.)

(4) Find the null eigenvector $(\lambda, \lambda\mathbf{a}, 1)^\top$ of the matrix in Fig. 2, where $\mathbf{M}$ is from Eq. 12 and $\mathbf{q} = \mathbf{M}^\top \mathbf{l}_{12}$.

(5) Having determined $\mathbf{a}$ in step (4), find $\mathbf{H}^\infty$ using

$$\mathbf{H}^\infty \cong \mathbf{M} + \mathbf{e}_B\mathbf{a}^\top \tag{16}$$

(Remarks) Eq. 16 has been used by many authors (e.g., [5]). Only the matrix used in step 4 needs further explanation: Rows 2 through 4: Letting σ_1 and σ_2 be the scaling factors for Eq. 14 and Eq. 16 leads to $\mathbf{H}^\infty\mathbf{h}_3 = \sigma_1\mathbf{h}_3$ and $\sigma_2\mathbf{H}^\infty = \mathbf{M} + \mathbf{e}_B\mathbf{a}^\top$. These equations combine to produce $\underline{\mathbf{h}}_3 = \lambda(\mathbf{M}\underline{\mathbf{h}}_3 + \mathbf{e}_B\underline{\mathbf{h}}_3^\top\mathbf{a})$ for $\lambda = 1/(\sigma_1\sigma_2)$. Rows 5 and 6: Using Eq. 15 and Eq. 16 gives $\mathbf{l}_{12} \cong \mathbf{M}^\top\mathbf{l}_{12} + \mathbf{e}_B^\top\mathbf{l}_{12}\mathbf{a} = \mathbf{q} + \xi\mathbf{a}$. Cross-multiply the x and y coordinates on each side to eliminate the unknown scalar. Derivation of row 1 is explained in Appendix C.

Thus once the fundamental matrix $\mathbf{F}$ between two views taken by the same camera has been determined, the set of all possible $\mathbf{H}^\infty$'s for the two cameras is parameterized by two real numbers, θ and κ. Furthermore, θ is the angle of rotation between the two cameras and thus has a very intuitive meaning. κ does not have an intuitive meaning by itself, but can easily be combined with $\mathbf{F}$ via Eq. 12 to yield $\underline{\mathbf{h}}_3$, which is intuitively understood as the vanishing point of the rotation axis. If θ is known, then $\mathbf{H}^\infty$ is known up to a single real parameter. If θ can be restricted to a narrow range (such as, $0 < \theta < .05$ radians) then the set of possible $\mathbf{H}^\infty$'s is similarly restricted. If the rotation axis is known or can be restricted to be within a certain area of the image, then $\mathbf{H}^{\infty}$ is also similarly restricted. The ability to restrict $\mathbf{H}^{\infty}$ using the kinds of knowledge just discussed is not directly shared by the modulus constraint.

To conclude this section, we make one final observation: In Eq. 16, every $\mathbf{a} \in \Re^3$ leads to an $\mathbf{H}^{\infty}$ that satisfies Eq. 4. However, the modulus constraint forces all legal choices of $\mathbf{a}$ to lie on a two-dimensional manifold embedded in $\Re^3$. The numbers κ and θ parameterize the modulus-constraint manifold, and do so in an intuitive manner.

4. Manifold intersection algorithm for metric self calibration

We now show how the parameterization presented in Section 3.1 can be used for metric self calibration. Our method is distinct from all previous self calibration methods and shares some advantages of Faugeras' method based on the Kruppa constraints and Pollefeys' method based on the modulus constraint.

We can give $\mathbf{K}$ the specific representation:

$$\mathbf{K} = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & 1 \end{bmatrix}$$

and think of $\mathbf{K}$ as being a point somewhere in a 5-dimensional space; we term this space *K-space*. We take a series of n views with the camera, of which m pairs of views contain enough shared information to find fundamental matrices $\mathbf{F}_1, \ldots, \mathbf{F}_m$.

For a specific fundamental matrix $\mathbf{F}_i$, each triple $\langle\kappa, \theta, \gamma\rangle$ yields a point in K-space that might be the true $\mathbf{K}$. Thus each $\mathbf{F}_i$ defines a 3-dimensional manifold $\mathbf{M}_i$ in K-space containing the true $\mathbf{K}$; each manifold is parameterized by $\langle\kappa, \theta, \gamma\rangle$. The intersection of all the $\mathbf{M}_i$ must contain the point representing the true $\mathbf{K}$. Thus if $m \geq 3$, $\mathbf{K}$ can be found. If one or more of the parameters κ, θ, or γ are known for a particular view pair (for instance, if the angle of rotation between two views is known), then m can be less than 3. Of course, in practice many more image pairs than the minimum are used for stability. Since it is not necessary to put all the cameras in the same projective basis, every pairwise fundamental matrix can be used and so it is easy to make m large.

Our algorithm involves "sketching out" the manifolds $\mathbf{M}_i$ to determine the unique intersection point $\mathbf{K}$. We sketch a manifold $\mathbf{M}_i$ by randomly selecting triples $\langle\kappa, \theta, \gamma\rangle$ to yield random points on $\mathbf{M}_i$. The particular approach we have implemented uses a voting scheme as follows:

---

(1) A volume $\mathcal{V}$ of K-space that contains the point $\mathbf{K}$ is chosen (see comments after the algorithm).

(2) $\mathcal{V}$ is subdivided into small voxels (5-dimensional hypercubes). A finer subdivision slows down the algorithm but improves success. However, because real data contains noise, the voxels must be large enough to contain the near intersection of all manifolds.

(3) For each manifold i a triple $\langle\kappa, \theta, \gamma\rangle$ is randomly selected and the corresponding point in K-space is determined. The voxel $\mathbf{v}$ containing this point is then determined and a vote is cast for $\mathbf{v}$ provided manifold i has not already voted for $\mathbf{v}$.

(4) Step (3) is repeated until one voxel $\mathbf{w}$ receives enough votes.

(5) "Zoom in" step: A new volume $\mathcal{U}$ half the size of $\mathcal{V}$ is centered around the winning voxel $\mathbf{w}$, and the algorithm returns to step (2) using the smaller volume $\mathcal{U}$ in place of $\mathcal{V}$.

(6) The algorithm continues until a sufficient resolution has been reached (i.e., the volume $\mathcal{V}$ is small enough).

After several "zoom in" steps, the manifolds become approximately linear (i.e., they begin to look "flat") within the reduced search region. Once three or more manifolds are approximately linear, the point of intersection can be determined in a single step by fitting hyperplanes to the locally "flat" manifolds and then intersecting the hyperplanes (by solving a linear system). Not only does this observation provide a way to greatly speed up the algorithm, but it also defines a stopping condition: The algorithm finishes when enough manifolds are approximately linear and the linear approximations all intersect at a single point (within some tolerance). Our implementation uses this criterion, and in the experiments of Section 5, almost all trial runs reached the linearity stage after 4 or 5 zoom in steps.

In step (1), the initial search region can always be set to the hypercube with side length 2 centered at the origin. Simply make sure to divide each randomly generated manifold point (when thought of as a calibration matrix) by its Frobenius norm, forcing each matrix component into the range $[-1, 1]$. Rescaling is permissible because the internal calibration is only defined up to a scale factor.

Because of noise, in real applications the manifolds will not all intersect at a single, well-defined point. All self-calibration algorithms have to deal with this fact whether they explicitly calculate the manifolds or not. By "sketching out" the manifolds, our algorithm has the advantage that it can find a small *region* that is intersected by all or most of the manifolds, and it can provide some measure of confidence that this region contains the true solution: the more manifolds a region contains and the smaller it is, the more reliable it is. More importantly, our approach is inherently robust to outliers. If any $\mathbf{F}_i$ is severely wrong (and thus an outlier), it will generate a manifold that does not contain the true $\mathbf{K}$. Votes generated by this manifold will not coincide with votes generated by the other (inlying) manifolds and thus this erroneous manifold will not influence the

MEDIAN CALIBRATION ERROR					
	average noise added per point				
	5 pixels	2 pixels	1 pixels	0.5 pixels	0.25 pixels
100 points	0.0629	0.0345	0.0172	0.00806	0.00567
60 points	0.0701	0.0489	0.0233	0.00878	0.00771
30 points	0.1880	0.0583	0.0359	0.01400	0.00855
10 points	0.5390	0.1450	0.1350	0.03280	0.03640

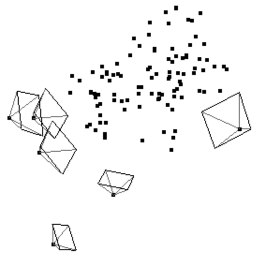

Figure 3: **Table of median errors for differing noise levels and numbers of scene points for synthetic data trials. At right is shown the scene and cameras for one trial with 100 scene points (all trials used 6 cameras).**

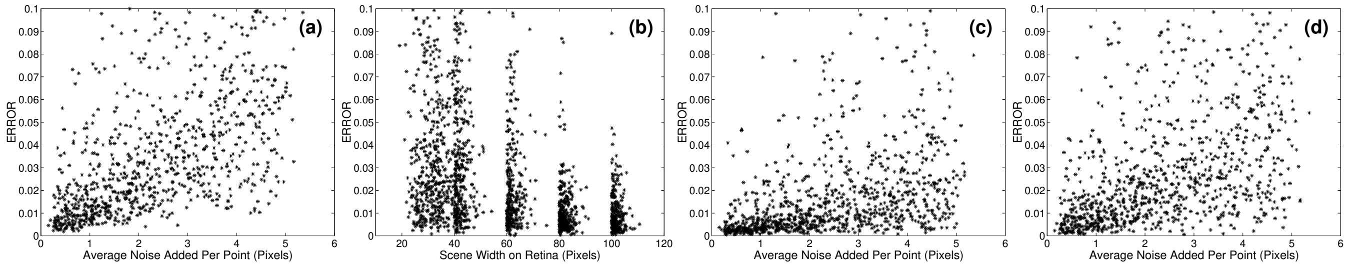

Figure 4: **(a) error vs. noise, (b) error vs. scene size on the retina, (c) error (in components on the main diagonal of $\mathbf{K}$) vs. noise, (d) error (in principal point) vs. noise**

zooming-in process of the algorithm. After a few zoom-in steps, the incorrect manifold will be outside volume $\mathcal{V}$ and will be completely ignored.

Another advantage to our approach is that certain kinds of additional information can be easily and naturally incorporated into the algorithm to make the search task easier and more robust. For example, if the angle of rotation between any pair of views is known, then the manifold for this pair is only 2-dimensional, greatly narrowing the search. If it is known that $\mathbf{K}_{(12)}$ (the skew factor) is close to 0, then the search space is 4 dimensional; only manifold points that have their skew factor close to 0 need to be considered. A similar effect happens if it is known that $\mathbf{K}_{(11)}$ and $\mathbf{K}_{(22)}$ are almost equal.

5. Experiments

Extensive experiments with synthetic and real data were performed to test the manifold intersection algorithm.

Simulations

For each trial run, a scene consisting of 10 to 100 points positioned randomly with uniform density inside a unit sphere was generated. The scene was viewed by 6 or 7 cameras positioned randomly but such that every point in the scene was visible from every camera. All cameras had the same internal parameters, which were generated randomly for each trial using realistic ranges for each parameter. The image size for each camera was 500×500 pixels; this is important when interpreting the results of the trial runs.

Noise was added to the projected position of each scene point. Distribution of the noise was uniform, meaning, for example, if the mean of the added noise was 5 pixels then anywhere between 0 and 10 pixels of noise was added in a random direction with an equal likelihood for each amount. The error between the true internal calibration matrix $\mathbf{K}$ and the calculated calibration $\mathbf{M}$ was calculated as

$$\textsc{error}(\mathbf{K}, \mathbf{M}) = \textsc{frob}(\mathbf{K}/\textsc{frob}(\mathbf{K}) - \mathbf{M}/\textsc{frob}(\mathbf{M}))$$

where $\textsc{frob}(\mathbf{M})$ is the Frobenius norm of matrix $\mathbf{M}$.

The table in Fig. 3 shows how error was related to the number of points in the scene and the average (mean) of the noise added to each point on the image plane. When generating this table, all trials used 6 cameras. One conclusion is that tracking extra scene points can make up for a lack of accuracy in locating the points on the image plane.

Each trial used in generating the graphs in Fig. 4 consisted of 60 scene points and 7 cameras. Fig. 4(b) demonstrates how error is related to the size of the scene as it appears on each camera's image plane. For every trial run used in this graph, the amount of noise added per point had a mean of 2 pixels. Retinal scene size was taken to be the smallest retinal scene size for any view in the trial. The graph shows how the algorithm becomes more stable as the scene covers more of the image plane.

Fig. 4(a) shows how error increases with higher noise levels. Interestingly, error increased at different rates for

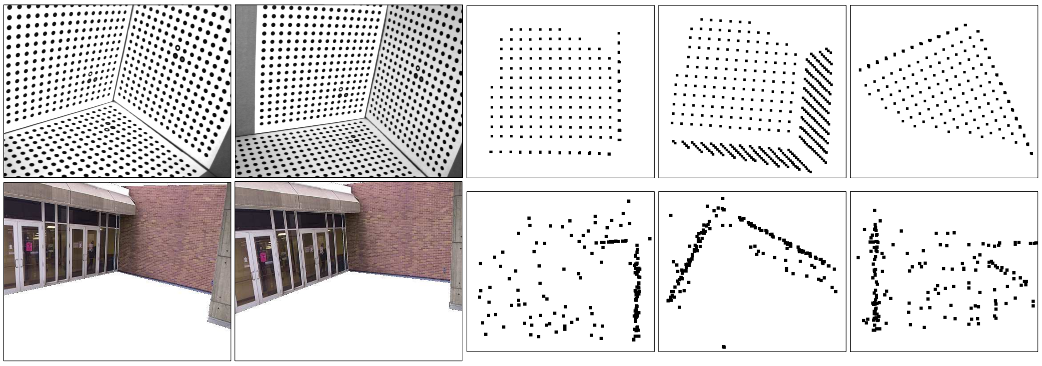

Figure 5: ***(top row)*** **Two views from an experiment using a real camera and a calibration object, and three views of the reconstructed feature points.** ***(bottom row)*** **Two views from an experiment using automatic point tracking, and three views of the reconstructed feature points.**

different components of the calibration matrix, as demonstrated in Fig. 4(c)-(d). Fig. 4(c) shows the growth of error only in the diagonal elements of the calculated internal calibration matrix (i.e., $\mathbf{K}_{(11)}$ and $\mathbf{K}_{(22)}$) while Fig. 4(d) shows growth of error in the last column of $\mathbf{K}$ (i.e., $\mathbf{K}_{(13)}$ and $\mathbf{K}_{(23)}$, which form the principal point). The graphs demonstrate that error in the calculated principal point grows at a much higher rate and has more randomness than error in the diagonal elements, and that the error values in Fig. 4(a) are largely a result of errors in the calculated principal point.

Further experiments

Fig. 5 shows the results of two more experiments. The first experiment used a real camera and an accurate calibration target. The target consisted of three walls meeting at right angles. Covering the walls were uniform dot patterns (including some reference dots) printed with a laser printer. The center of each dot was automatically recovered using a separate algorithm; correspondences were determined automatically via the reference dots. The algorithm was run using 8 views of the calibration object and the object was reconstructed using the calculated internal calibration. The reconstructions show perpendicular walls and uniform, rectilinear spacing of the dot centers, confirming an accurate metric reconstruction.

In the second experiment, 7 views of a synthetic scene were used. The synthetic scene was based on a real scene, and photographs from the original scene were texture mapped onto the model to achieve a high level of realism. Although the scene was computer generated, no additional information (other than the views themselves) was used by the algorithm. Points were automatically tracked between the views using an off-the-shelf point tracking algorithm and the automatically-determined point correspondences were then fed into the manifold intersection algorithm to determine the internal calibration. Three views of the reconstructed feature points are given in the figure. The center reconstruction is an overhead view showing that the two main walls are perpendicular to each other. In the left-most reconstruction, points on the narrow roof that overhangs the doorway are visible and an approximate right angle between the overhang and the wall is evident. The right-most reconstruction shows an edge-on view of the other wall; note the feature points are not perfectly in the same plane, indicating a certain amount of noise in the automatically-tracked points.

6. Concluding remarks

An overview of this paper including its main contributions is as follows: When two views of a static scene are taken by a camera with fixed internal parameters, it is always possible to interpret the camera as being fixed in position and the scene as sitting on a rising turntable. Using this reinterpretation, we have presented a new decomposition of the fundamental matrix between the views (Eq. 1). When the fundamental matrix between two views is known, we have shown through this decomposition that the internal calibration matrix is parameterized by only 3 parameters, each of which has an intuitive meaning. The 3 parameters generate a "screw-transform" manifold in K-space. We have presented an algorithm for metric self-calibration that works by intersecting screw-transform manifolds. We have also presented some additional useful properties of the underlying mathematical framework.

A. Finding rotation angle from relative calibration

The relative calibration $\mathbf{H}^\infty$ between two internally-equal cameras can be written as $\mathbf{KRK}^{-1}$, where $\mathbf{R}$ is a rotation matrix. The rotation angle θ associated with $\mathbf{R}$ can be found from $\mathbf{H}^\infty$ as follows: Let

$$\mathbf{H}^\infty = \mathbf{KRK}^{-1} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & j \end{bmatrix} \tag{17}$$

Because $\mathbf{H}^{\infty}$ is conjugate to $\mathbf{R}$, it has the same eigenvalues as the rotation matrix; these eigenvalues have the form: $\lambda_1 = \omega$, $\lambda_2 = \omega\exp(i\theta)$, and $\lambda_3 = \omega\exp(-i\theta)$. The λ_i can be found from $\mathbf{H}^{\infty}$ using the characteristic equation

$$\begin{aligned} 0 &= \det(\mathbf{H}^{\infty} - \lambda\mathbf{I}) \\ &= \lambda^3 - \lambda^2(a+e+j) - \lambda(-ae-aj-ej+cg+bd \\ &\quad +fh) - (aej+bfg+cdh-cge-bdj-fha) \\ &= \lambda^3 + \alpha\lambda^2 + \beta\lambda + \gamma \qquad (18) \end{aligned}$$

where

$$\begin{aligned} \alpha &= -(a+e+j) \\ \gamma &= -(aej+bfg+cdh-cge-bdj-fha) \end{aligned}$$

Thus (using a well-known property from algebra)

$$\begin{aligned} \gamma &= -\lambda_1\lambda_2\lambda_3 = -\omega^3 \\ \alpha &= -(\lambda_1+\lambda_2+\lambda_3) \\ &= -\omega - \omega\exp(i\theta) - \omega\exp(-i\theta) \\ &= -\omega(1+2\cos\theta) \end{aligned}$$

and θ can be found via

$$\theta = \arccos\left[\frac{1}{2}\left(\frac{\alpha}{(\gamma)^{\frac{1}{3}}} - 1\right)\right] \qquad (19)$$

Note that θ can only be determined up to a sign by this method (due to the symmetry of the eigenvalues). If available, additional information may be used to disambiguate the sign.

B. Parameterization of $\mathbf{h}_3$

Eq. 12 is derived via

$$\begin{aligned} (\mathbf{H}^{\infty})\mathbf{h}_3 &\cong \mathbf{h}_3 \\ (\mathbf{M} + \mathbf{e}\mathbf{a}^{\top})\mathbf{h}_3 &\cong \mathbf{h}_3 \\ (\mathbf{M} + \mathbf{e}\mathbf{a}^{\top})\mathbf{h}_3 &= k_1\mathbf{h}_3 \\ \mathbf{e}\mathbf{a}^{\top}\mathbf{h}_3 &= (k_1\mathbf{I} - \mathbf{M})\mathbf{h}_3 \\ k_2\mathbf{e} &= (k_1\mathbf{I} - \mathbf{M})\mathbf{h}_3 \\ (k_1\mathbf{I} - \mathbf{M})^{-1}\mathbf{e} &\cong \mathbf{h}_3 \qquad (20) \end{aligned}$$

C. Alternative expansion of the characteristic equation of $\mathbf{H}^{\infty}$

$\mathbf{H}^{\infty}$ has an alternative representation as $\mathbf{M} + \mathbf{e}\mathbf{a}^{\top}$ (see Eq. 16), where $\mathbf{M}$ and $\mathbf{e}$ can be found directly from the fundamental matrix between the views. Expanding the characteristic equation for $\mathbf{H}^{\infty}$ using this alternative representation can yield some useful constraints. In what follows, we use $|\mathbf{Q}|$ for the determinant of matrix $\mathbf{Q}$. The characteristic equation is

$$0 = |\mathbf{H}^{\infty} - \lambda\mathbf{I}| = |\mathbf{M} + \mathbf{e}\mathbf{a}^{\top} - \lambda\mathbf{I}| \qquad (21)$$

Define $\mathbf{g}_i$ and $\mathbf{u}_i$ by

$$[\,\mathbf{g}_1\ \ \mathbf{g}_2\ \ \mathbf{g}_3\,] = \mathbf{M} \quad \text{and} \quad [\,\mathbf{u}_1\ \ \mathbf{u}_2\ \ \mathbf{u}_3\,] = \mathbf{I}$$

Using a triple-product for the determinant, the right-hand side of Eq. 21 can be written

$$(\mathbf{g}_1 + a_1\mathbf{e} - \lambda\mathbf{u}_1)\cdot(\mathbf{g}_2 + a_2\mathbf{e} - \lambda\mathbf{u}_2)\times(\mathbf{g}_3 + a_3\mathbf{e} - \lambda\mathbf{u}_3)$$

Expansion leads to

$$0 = \kappa_3(\omega\lambda)^3 + \kappa_2(\omega\lambda)^2 + \kappa_1(\omega\lambda)^1 + \kappa_0 \qquad (22)$$

where

$$\begin{aligned} \kappa_3 &= -1 \\ \kappa_2 &= |\mathbf{g}_1\mathbf{u}_2\mathbf{u}_3| + a_1|\mathbf{e}\mathbf{u}_2\mathbf{u}_3| + |\mathbf{u}_1\mathbf{g}_2\mathbf{u}_3| + a_2|\mathbf{u}_1\mathbf{e}\mathbf{u}_3| \\ &\quad + |\mathbf{u}_1\mathbf{u}_2\mathbf{g}_3| + a_3|\mathbf{u}_1\mathbf{u}_2\mathbf{e}| \\ \kappa_1 &= |\mathbf{g}_1\mathbf{g}_2\mathbf{u}_3| - a_2|\mathbf{g}_1\mathbf{e}\mathbf{u}_3| - |\mathbf{g}_1\mathbf{u}_2\mathbf{g}_3| - a_3|\mathbf{g}_1\mathbf{u}_2\mathbf{e}| \\ &\quad - a_1|\mathbf{e}\mathbf{u}_2\mathbf{g}_3| - |\mathbf{u}_1\mathbf{g}_2\mathbf{g}_3| - a_3|\mathbf{u}_1\mathbf{g}_2\mathbf{e}| - a_2|\mathbf{u}_1\mathbf{e}\mathbf{g}_3| \\ \kappa_0 &= |\mathbf{M}| + a_3|\mathbf{g}_1\mathbf{g}_2\mathbf{e}| + a_2|\mathbf{g}_1\mathbf{e}\mathbf{g}_3| + a_1|\mathbf{e}\mathbf{g}_2\mathbf{g}_3| \\ &\quad - a_1|\mathbf{e}\mathbf{g}_2\mathbf{u}_3| \end{aligned}$$

Dividing Eq. 22 by the coefficient of λ^3 allows Eq. 22 and Eq. 18 to be equated. In this way, row 1 of the matrix in Fig. 2 is derived by equating α in Eq. 18 with $-\kappa_2/\omega$. In Fig. 2, λ is $1/\omega^2$.

References

[1] O. Bottema. *Theoretical Kinematics*. North-Holland Publishing Company, New York, 1979.

[2] O. D. Faugeras, Q.-T. Luong, and S. J. Maybank. Camera self-calibration: theory and experiments. In *Proc. European Conf. on Computer Vision*, pages 321–334, 1992.

[3] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3D model construction for turn-table sequences. In R. Koch and L. V. Gool, editors, *Proc. Workshop on 3D Structure from Multiple Images of Large-Scale Environments (SMILE '98)*, pages 155–170. Springer, 1998.

[4] R. I. Hartley. Projective reconstruction and invariants from multiple images. *IEEE Trans. Pattern Analysis and Machine Intell.*, 16(10):1036–1041, 1994.

[5] Q.-T. Luong and T. Viéville. Canonic representations for the geometries of multiple projective views. In *Proc. Third European Conf. on Computer Vision*, pages 589–597, 1994.

[6] S. Maybank and O. D. Faugeras. A theory of self calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[7] M. Pollefeys. *Self-Calibration and Metric 3D Reconstruction from Uncalibrated Image Sequences*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 1999.

[8] M. Pollefeys and L. Van Gool. A stratified approach to metric self-calibration. In *Proc. Computer Vision and Pattern Recognition Conf.*, pages 407–412, 1997.

[9] M. Pollefeys and L. Van Gool. Stratified self-calibration with the modulus constraint. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(8):707–724, August 1999.

[10] W. Triggs. Autocalibration and the absolute quadric. In *Proc. Computer Vision and Pattern Recognition Conf.*, pages 609–614, 1997.