Three-View Multibody Structure from Motion

René Vidal, *Member*, *IEEE*, and Richard Hartley, *Member*, *IEEE*

Abstract—We propose a geometric approach to 3D motion segmentation from point correspondences in three perspective views. We demonstrate that after applying a polynomial embedding to the point correspondences, they become related by the so-called multibody trilinear constraint and its associated multibody trifocal tensor, which are natural generalizations of the trilinear constraint and the trifocal tensor to multiple motions. We derive a rank constraint on the embedded correspondences from which one can estimate the number of independent motions, as well as linearly solve for the multibody trifocal tensor. We then show how to compute the epipolar lines associated with each image point from the common root of a set of univariate polynomials and the epipoles by solving a pair of plane clustering problems using Generalized Principal Component Analysis (GPCA). The individual trifocal tensors are then obtained from the second-order derivatives of the multibody trilinear constraint. Given epipolar lines and epipoles or trifocal tensors, one can immediately obtain an initial clustering of the correspondences. We use this clustering to initialize an iterative algorithm that alternates between the computation of the trifocal tensors and the segmentation of the correspondences. We test our algorithm on various synthetic and real scenes and compare it with other algebraic and iterative algorithms.

Index Terms—Multibody structure from motion, 3D motion segmentation, multibody trilinear constraint, multibody trifocal tensor, Generalized PCA (GPCA).

1 Introduction

One of the most important problems in visual motion analysis is that of reconstructing a 3D scene from a collection of images taken by a moving camera. At present, the algebraic and geometric aspects of this problem are very well understood. For example, it is known that two, three, and multiple views of a scene are related by the so-called bilinear, trilinear, and multilinear constraints, respectively. Also, there are various algorithms for performing the reconstruction task, both geometric and optimization-based [14].

However, most of these algorithms assume that the scene is *static*, that is, either the camera or a single object in the scene moves; hence, they can only estimate a single motion model from the image measurements. In practice, most scenes are *dynamic*, that is, both the camera and multiple objects in the 3D world move independently. Thus, one is faced with the more challenging *multibody structure from motion* problem of recovering multiple motion models from the image data, without knowing the assignment of data points to motion models.

1.1 Previous Work

Multibody structure from motion has received increasing attention over the past few years. Existing approaches [26] solve this problem by successive computation of dominant motions using methods from robust statistics such as Random Sample Consensus (RANSAC) [7]. First, a single motion model is computed by applying RANSAC to all image measurements. Then, the measurements that fit this dominant motion model well (inliers) are removed from the data set, and RANSAC is reapplied to the remaining points to obtain a second motion model. The process is repeated until most measurements have been assigned to a model. Alternative approaches [6] first cluster the features corresponding to the same motion using, for example, K-Means or spectral clustering, and then estimate a single motion model for each group using standard structure from motion algorithms. This can also be done in a probabilistic framework by alternating between feature clustering and single-body motion estimation using the Expectation-Maximization (EM) algorithm [3]. When the probabilistic model generating the data is known, this iterative method provides an optimal estimate in the maximum likelihood sense. However, it is well known that EM is very sensitive to initialization [25].

In order to deal with the initialization problem, recent work has concentrated on the geometry of dynamic scenes, including the analysis of multiple points moving linearly with constant speed [11], [21], multiple points moving in a plane [24], multiple translating planes [33], and self-calibration from multiple motions [8], [12]. Vidal et al. [30] propose a polynomial factorization algorithm for segmenting purely translating objects. Wolf and Shashua [34] derived a bilinear constraint in $\mathbb{R}^6$, which, together with a combinatorial scheme, segments two rigid-body motions from two perspective views. Vidal et al. [32] propose a generalization of the epipolar constraint and of the fundamental matrix to multiple rigid-body motions, which leads to a motion segmentation algorithm based on factoring products of epipolar constraints to retrieve the fundamental matrices associated with each one of the motions. Vidal and Ma [28] extend this method to most two-view motion models such as affine, translational, and planar homographies by fitting and differentiating complex polynomials. All these two-view algorithms are algebraic; hence, they do not require initialization.

Although in general two views are sufficient for solving the motion estimation and segmentation problem, there are some degenerate situations in which two-view algorithms

- *R. Vidal is with the Center for Imaging Science, Department of Biomedical Engineering, The Johns Hopkins University, 308B Clark Hall, 3400 N. Charles St., Baltimore MD 21218. E-mail: rvidal@cis.jhu.edu.*
- *R. Hartley is with the Department of Information Engineering, Australian National University, Canberra ACT 0200, Australia. E-mail: Richard.Hartley@anu.edu.au.*

Manuscript received 5 June 2006; revised 23 Nov. 2006; accepted 23 Feb. 2007; published online 21 May 2007.
Recommended for acceptance by M. Pollefeys.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0420-0606.
Digital Object Identifier no. 10.1109/TPAMI.2007.1179.

0162-8828/08/$25.00 © 2008 IEEE Published by the IEEE Computer Society

may fail. For instance, if the scene consists of a planar object that is moving on its own plane and the camera is also moving in the same plane, then one cannot tell from two views whether the scene consists of one or two motions. Unfortunately, real video sequences are commonly close to this type of degenerate configurations. In such cases, a minimum of three views is needed in order to properly segment the two motions. To the best of our knowledge, other than [13], there is no previous work addressing motion estimation and segmentation from three perspective views. The only existing works on multiframe 3D motion segmentation are for points moving on a line in three perspective views [22], for multiple translating objects from line correspondences in three perspective views [23], and for rigid-body motions in three or more affine views [1], [2], [5], [9], [10], [15], [17], [18], [27], [35], [36].

1.2 Paper Contributions and Outline

In this paper, we present a geometric approach to the estimation and segmentation of an *unknown* number of rigid-body motions from a set of point correspondences in *three* perspective views. Our approach algebraically eliminates the feature clustering stage and directly solves for the motion parameters in an algebraic fashion. This is achieved by fitting a multibody motion model to all the image measurements and then factorizing this model to obtain the individual motion parameters. The final result is a natural generalization of the classical three-view geometry (trilinear constraint, trifocal tensor, and seven-point algorithm) to the case of multiple rigid-body motions.

Section 2 studies the three-view geometry and algebra of the multibody structure from motion problem. We introduce the *multibody trilinear constraint* as a geometric relationship between the motion parameters and the image points that is satisfied by all the correspondences, regardless of the body with which they are associated. We show that this constraint is trilinear on a polynomial embedding of the correspondences and linear on the so-called *multibody trifocal tensor* $\mathcal{T}$, an algebraic structure encoding the parameters of all rigid-body motions. We then study the geometric properties of $\mathcal{T}$ and show that it can be used for transferring points and lines from a pair of views to the other.

Section 3 presents a geometric algorithm for estimating the number of motions, the motion parameters, and the clustering of the correspondences. We first derive a rank constraint on the matrix of embedded correspondences from which one can estimate the number of independent motions n, as well as linearly solve for the multibody trifocal tensor $\mathcal{T}$. Given n and $\mathcal{T}$, we show that one can compute the epipolar lines associated with each correspondence from the common root of a set of univariate polynomials. By applying this process to all the correspondences, we obtain a collection of epipolar lines that must intersect at the n epipoles. The estimation of the epipoles is then shown to be equivalent to a pair of plane clustering problems, which we solve algebraically using Generalized Principal Component Analysis (GPCA) [31], [29]. Given epipolar lines and epipoles or trifocal tensors, one can immediately obtain an initial clustering of the correspondences. We use this clustering to initialize an iterative algorithm that alternates between the computation of the trifocal tensors and the segmentation of the correspondences. We test our algorithm on various synthetic and real dynamic scenes and compare it with other algebraic and iterative algorithms.

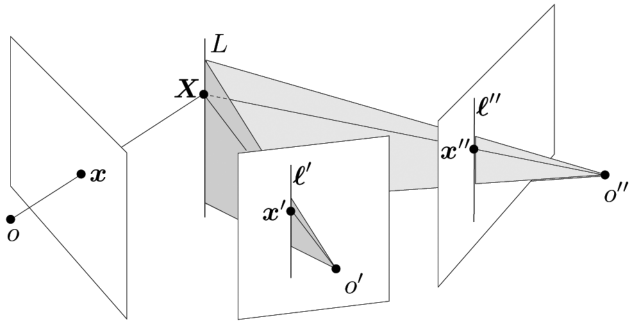


Fig. 1. Three-view geometry: projections x, ℓ', and ℓ'' of a point X and a line L in a 3D space onto three perspective views. The relative motion among the three views is encoded by the trifocal tensor $T \in \mathbb{R}^{3\times3\times3}$. The intersections of the lines (o, o') and (o, o'') with the second and third image planes, respectively, are the so-called epipoles e' and e''. (This figure and Fig. 2 are adapted from figures in [14].)

2 Multibody Three-View Geometry

This section establishes the basic geometric relationships among three perspective views of multiple rigid-body motions. We first review the trilinear constraint and its associated trifocal tensor for the case of a single rigid-body motion. We then generalize these notions to multiple motions via a polynomial embedding that leads to the so-called multibody trilinear constraint and its associated multibody trifocal tensor. We also study transfer properties of the multibody trifocal tensor from a pair of views to the other.

2.1 The Trilinear Constraint and the Trifocal Tensor

Let $\boldsymbol{x} \leftrightarrow \boldsymbol{\ell}' \leftrightarrow \boldsymbol{\ell}''$ be a point-line-line correspondence in three perspective views, as illustrated in Fig. 1. Also, let

$$\mathtt{P} = \begin{bmatrix} I & \mathbf{0} \end{bmatrix},\ \mathtt{P}' = \begin{bmatrix} R' & \boldsymbol{e}' \end{bmatrix} \text{ and } \mathtt{P}'' = \begin{bmatrix} R'' & \boldsymbol{e}'' \end{bmatrix} \in \mathbb{R}^{3\times4} \quad (1)$$

be the camera matrices in the first, second, and third views, where $\boldsymbol{e}' \in \mathbb{P}^2$ and $\boldsymbol{e}'' \in \mathbb{P}^2$ are the epipoles in the second and third views, respectively. Then, the multiple-view matrix [19]

$$\begin{bmatrix} \boldsymbol{\ell}'^\top R'\boldsymbol{x} & \boldsymbol{\ell}'^\top \boldsymbol{e}' \\ \boldsymbol{\ell}''^\top R''\boldsymbol{x} & \boldsymbol{\ell}''^\top \boldsymbol{e}'' \end{bmatrix} \in \mathbb{R}^{2\times2} \quad (2)$$

must have rank 1; hence, its determinant must be zero, that is,

$$\boldsymbol{\ell}'^\top (R'\boldsymbol{x}\boldsymbol{e}''^\top - \boldsymbol{e}'\boldsymbol{x}^\top R''^\top)\boldsymbol{\ell}'' = 0. \quad (3)$$

This is the well-known point-line-line *trilinear constraint* among the three views [14], which we will denote as

$$\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T = 0, \quad (4)$$

where $T \in \mathbb{R}^{3\times3\times3}$ is the so-called *trifocal tensor*.

Notation. *For ease of notation, we will drop the summation and the subscripts in trilinear expressions such as $\sum_{ijk} x_i \ell'_j \ell''_k T_{ijk}$ and write them as shown in (4). Similarly, we will write $\boldsymbol{x}T$ to represent the matrix whose (jk)th entry is $\sum_i x_i T_{ijk}$ and $\boldsymbol{x}\boldsymbol{\ell}'T$ to represent the vector whose kth entry is $\sum_{ij} x_i \ell'_j T_{ijk}$. The notation is somewhat condensed and inexact, since the particular indices that are being summed over are not specified. However, the meaning should in all cases be clear from the context.*

2.2 The Multibody Trilinear Constraint

Consider now a scene containing n rigid-body motions with associated trifocal tensors $\{T_i \in \mathbb{R}^{3\times3\times3}\}_{i=1}^n$, where T_i is the trifocal tensor associated with the motion of the $i\text{th}$ object relative to the moving camera among the three views. We assume that the motions of the objects relative to the camera are such that all the trifocal tensors are different up to a scale factor. We also assume that the given images correspond to 3D points in general configuration in $\mathbb{R}^3$, that is, they do not all lie in any critical surface, for example.

Let $\boldsymbol{x} \leftrightarrow \boldsymbol{\ell}' \leftrightarrow \boldsymbol{\ell}''$ be an arbitrary point-line-line correspondence associated with *any* of the n motions. Then, there exists a trifocal tensor T_i satisfying the trilinear constraint in (3) or (4). Thus, regardless of the motion associated with the correspondence, the following constraint must be satisfied by the number of independent motions n, the trifocal tensors $\{T_i\}_{i=1}^n$, and the correspondence $\boldsymbol{x} \leftrightarrow \boldsymbol{\ell}' \leftrightarrow \boldsymbol{\ell}''$:

$$\prod_{i=1}^{n}(\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T_i) = 0. \tag{5}$$

We call (5) the *multibody trilinear constraint*, because it is a natural generalization of the *trilinear constraint* valid for $n = 1$.

2.3 The Multibody Trifocal Tensor

The multibody trilinear constraint eliminates the problem of clustering the correspondences from the motion segmentation problem by taking the product of all trilinear constraints. Although taking the product is not the only way of algebraically eliminating feature segmentation, the product has the advantage of leading to a polynomial equation in $(\boldsymbol{x}, \boldsymbol{\ell}', \boldsymbol{\ell}'')$ with a nice algebraic structure. Indeed, the multibody constraint is a homogeneous polynomial of degree n in each of $\boldsymbol{x}$, $\boldsymbol{\ell}'$, or $\boldsymbol{\ell}''$. Now, suppose $\boldsymbol{x} = (x_1, x_2, x_3)^\top$. We may enumerate all the possible monomials $x_1^{n_1}x_2^{n_2}x_3^{n_3}$ of degree n in (5) and write them in some chosen order as a vector:

$$\widetilde{\boldsymbol{x}} = (x_1^n, x_1^{n-1}x_2, x_1^{n-1}x_3, x_1^{n-2}x_2^2, \ldots, x_3^n)^\top. \tag{6}$$

This vector has dimension $M_n = (n+1)(n+2)/2$. The map $\boldsymbol{x} \mapsto \widetilde{\boldsymbol{x}}$ is known as the polynomial embedding of degree n in the machine learning community and as the Veronese map of degree n in the algebraic geometry community. The vectors $\widetilde{\boldsymbol{\ell}'}$ and $\widetilde{\boldsymbol{\ell}''}$ are defined similarly in terms of $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$.

Now, note that (5) is a sum of the terms of degree n in each of $\boldsymbol{x}$, $\boldsymbol{\ell}'$, and $\boldsymbol{\ell}''$. Thus, each term is a product of degree n monomials in $\boldsymbol{x}$, $\boldsymbol{\ell}'$, and $\boldsymbol{\ell}''$. We may therefore define a 3D *multibody trifocal tensor* $\mathcal{T} \in \mathbb{R}^{M_n \times M_n \times M_n}$ containing the coefficients of each of the monomials occurring in the product (5) and write the multibody constraint (5) as

$$\widetilde{\boldsymbol{x}}\,\widetilde{\boldsymbol{\ell}'}\,\widetilde{\boldsymbol{\ell}''}\,\mathcal{T} = 0, \tag{7}$$

where the summation over all the entries of the vectors $\widetilde{\boldsymbol{x}}$, $\widetilde{\boldsymbol{\ell}'}$, and $\widetilde{\boldsymbol{\ell}''}$ is implied. The important point to observe is that although (7) has degree n in the entries of $\boldsymbol{x}$, $\boldsymbol{\ell}'$, and $\boldsymbol{\ell}''$, it is in fact *linear* in the entries of $\widetilde{\boldsymbol{x}}$, $\widetilde{\boldsymbol{\ell}'}$, and $\widetilde{\boldsymbol{\ell}''}$. Since (7) is a trilinear constraint on $\widetilde{\boldsymbol{x}}$, $\widetilde{\boldsymbol{\ell}'}$, and $\widetilde{\boldsymbol{\ell}''}$, we will refer to both (5) and (7) as the *multibody trilinear constraint* from now on.

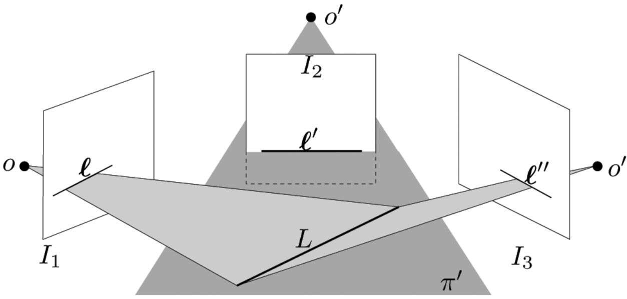


Fig. 2. A line $\boldsymbol{\ell}''$ in the third image defines a 3D plane, which intersects π' in a line L. This line is then imaged as the line $\boldsymbol{\ell} = \boldsymbol{\ell}'\boldsymbol{\ell}''T$ in the first view. Therefore, the trifocal tensor T maps a pair of corresponding lines $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$ on the second and third images onto a corresponding line $\boldsymbol{\ell}$ in the first image.

2.4 Transfer Properties of the Multibody Trifocal Tensor

An important property of the trifocal tensor T is that of transferring points and lines from a pair of views to the other [14]. For example, if $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$ are corresponding lines in the second and third views, then $\boldsymbol{\ell} = \boldsymbol{\ell}'\boldsymbol{\ell}''T$ is a corresponding line in the first view, as illustrated in Fig. 2. Similarly, if $\boldsymbol{x}$ is a point in the first view and $\boldsymbol{\ell}'$ is a corresponding line in the second view, then $\boldsymbol{x}'' = \boldsymbol{x}\boldsymbol{\ell}'T$ is the corresponding point in the third view. Likewise, $\boldsymbol{x}' = \boldsymbol{x}\boldsymbol{\ell}''T$ is the point in the second view corresponding to $(\boldsymbol{x}, \boldsymbol{\ell}'')$.

We now discuss the transfer properties of the multibody trifocal tensor $\mathcal{T}$. Although in principle these properties are natural generalizations of the corresponding properties of the individual trifocal tensors $\{T_i\}_{i=1}^n$, in the multibody case, the situation is more complex, because $\mathcal{T}$ incorporates information about *all* the motions at the same time. Indeed, if $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$ are two lines in the second and third views, then $\boldsymbol{\ell}_i = \boldsymbol{\ell}'\boldsymbol{\ell}''T_i$ is a corresponding line in the first view according to the $i\text{th}$ motion. Now, from the multibody trifocal constraint, we have

$$\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}'}\widetilde{\boldsymbol{\ell}''}\mathcal{T} = \prod_{i=1}^{n}(\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T_i) = \prod_{i=1}^{n}(\boldsymbol{x}^\top\boldsymbol{\ell}_i) \tag{8}$$

hence, the vector $\widetilde{\boldsymbol{\ell}'}\widetilde{\boldsymbol{\ell}''}\mathcal{T} \in \mathbb{R}^{M_n}$ represents the coefficients of the homogeneous polynomial in $\boldsymbol{x}$:

$$q_n(\boldsymbol{x}) = (\boldsymbol{x}^\top\boldsymbol{\ell}_1)(\boldsymbol{x}^\top\boldsymbol{\ell}_2)\cdots(\boldsymbol{x}^\top\boldsymbol{\ell}_n). \tag{9}$$

Therefore, given $\widetilde{\boldsymbol{\ell}'}\widetilde{\boldsymbol{\ell}''}\mathcal{T}$, we can compute the lines $\{\boldsymbol{\ell}_i\}_{i=1}^n$ by factorizing the homogeneous polynomial of degree n, $q_n(\boldsymbol{x})$, into a product of n homogeneous polynomials of degree one $\{(\boldsymbol{\ell}_i^\top\boldsymbol{x})\}_{i=1}^n$. A technique for performing such a factorization can be found in [30].[1] We can interpret this factorization process as a generalization of the conventional line transfer property of the multibody trifocal tensor to multiple motions. In essence, the multibody trifocal tensor $\mathcal{T}$ allows us to "transfer" a pair of lines $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$ in the second and third views to a set of n lines in the first view, as shown geometrically in Fig. 3. In an entirely analogous

[1] It was shown in [30] that this polynomial factorization problem has a unique solution (up to a scale for each factor) that is algebraically equivalent to solving for the roots of a polynomial of degree n in *one* variable plus solving a linear system in n variables.

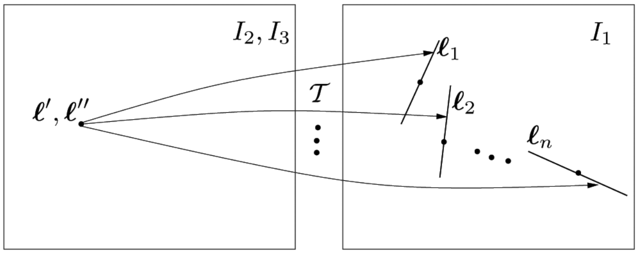


Fig. 3. The multibody trifocal tensor $\mathcal{T}$ maps each pair of corresponding lines ℓ' and ℓ'' in the second and third views to n lines $\ell_1, \dots, \ell_n$ in the first view. These n transferred lines correspond to the n different motions encoded in the trifocal tensor $\mathcal{T}$. (This figure and Fig. 5 are adapted from figures in [20].)

fashion, the factorization of $\widetilde{x}\widetilde{\ell'}\mathcal{T}$ gives the n corresponding points x_i'' in the third view, and the factorization of $\widetilde{x}\widetilde{\ell''}\mathcal{T}$ gives the n corresponding points x_i' in the second view.

In some cases, one may be interested in transferring two corresponding lines ℓ' and ℓ'' to their corresponding line ℓ according to, say, the $i\text{th}$ motion. Notice that this cannot be done from the multibody trifocal tensor $\mathcal{T}$ alone, which encodes information about the n motions. As discussed earlier, the factorization of $\widetilde{\ell'}\widetilde{\ell''}\mathcal{T}$ will give n lines $\{\ell_i\}_{i=1}^n$. Since the line associated with the ith motion must pass through a corresponding point x in the first view, given the n lines $\{\ell_i\}_{i=1}^n$, we can identify ℓ_i as the one that minimizes $(\ell_k^\top x)^2$ for $k = 1, \dots, n$.

There is, however, a simpler and more elegant way of computing the line ℓ associated with a point-line-line correspondence, which is by looking at the derivatives of the multibody trilinear constraint, thus avoiding polynomial factorization.[2] We begin by considering the derivative of the multibody trilinear constraint with respect to its first argument

$$\frac{\partial}{\partial x}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T}) = \frac{\partial}{\partial x}\prod_{i=1}^n (x\ell'\ell''T_i) = \sum_{i=1}^n (\ell'\ell''T_i)\prod_{k\neq i}(x\ell'\ell''T_k).$$

We notice that if we evaluate this derivative at a correspondence $x \leftrightarrow \ell' \leftrightarrow \ell''$ associated with the $i\text{th}$ motion, that is, the correspondence is such that $x\ell'\ell''T_i = 0$, then all the terms in the above summation but the $i\text{th}$ vanish. Thus, we obtain

$$\frac{\partial}{\partial x}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T})\bigg|_{x\ell'\ell''T_i=0} = (\ell'\ell''T_i)\prod_{k\neq i}(x\ell'\ell''T_k) \sim (\ell'\ell''T_i),$$

which from the properties of the trifocal tensor T_i gives a line ℓ in the first view. Notice that this line ℓ in the first view is *transferred* from the two lines in the second and third views according to the *unknown* $i\text{th}$ trifocal tensor T_i. That is, the multibody trifocal tensor enables us to transfer corresponding lines according to their own motion, without having to know the motion with which the correspondence is associated. We therefore have the following result.

Theorem 1 (Line transfer from corresponding lines in the second and third views to the first). *The derivative of the multibody trilinear constraint with respect to its first argument evaluated at a correspondence (x, ℓ', ℓ'') gives a line ℓ in the first view passing though x, that is,*

$$\ell = \frac{\partial}{\partial x}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T}) \quad \text{and} \quad \ell^\top x = 0. \tag{10}$$

Different choices for ℓ' and ℓ'' will give different lines ℓ.

In a similar fashion, if we now consider the derivative of the multibody trilinear constraint with respect to its second argument and evaluate it at a correspondence (x, ℓ', ℓ'') associated with the $i\text{th}$ motion, then we obtain

$$\frac{\partial}{\partial \ell'}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T})\bigg|_{x\ell'\ell''T_i=0} = (x\ell''T_i)\prod_{k\neq i}(x\ell'\ell''T_k) \sim (x\ell''T_i).$$

From the properties of T_i, this gives a corresponding point x' in the second view. Similarly, the derivative with respect to the third argument gives the corresponding point in the third view x''. We therefore have the following result.

Theorem 2 (Point transfer from the first to the second and third views). *The derivative of the multibody trilinear constraint with respect to its second and third arguments evaluated at a correspondence (x, ℓ', ℓ'') gives the corresponding point in the second and third views x' and x'', respectively, that is,*

$$\frac{\partial}{\partial \ell'}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T}) \sim x' \text{ and } \frac{\partial}{\partial \ell''}(\widetilde{x}\widetilde{\ell'}\widetilde{\ell''}\mathcal{T}) \sim x''. \tag{11}$$

3 Multibody Motion Estimation and Segmentation from Three Perspective Views

In this section, we present a linear algorithm for segmenting a scene consisting of multiple rigid motions. More specifically, we assume that we are given a set of point correspondences $\{x_j \leftrightarrow x_j' \leftrightarrow x_j''\}_{j=1}^N$ and show how to estimate the number of independent motions n, the individual trifocal tensors $\{T_i\}_{i=1}^n$, and the clustering of the correspondences. Our algorithm proceeds as follows: In Section 3.1, we show how to compute the number of motions n and the multibody trifocal tensor $\mathcal{T}$ from a rank constraint on the embedded correspondences. In Section 3.2, we show how to estimate the epipolar lines in the second and third views, ℓ_x' and ℓ_x'', respectively, associated with each point x in the first view by solving for the common root of a set of univariate polynomials. In Section 3.3, we show how to estimate the epipoles in the second and third views, $\{e_i'\}_{i=1}^n$ and $\{e_i''\}_{i=1}^n$, respectively, by solving a plane clustering problem. Given epipolar lines and epipoles, one may immediately cluster the correspondences into n groups and then estimate individual trifocal tensors and camera matrices from the data associated with each group. Alternatively, one may recover the individual trifocal tensors directly from the second-order derivatives of the multibody trilinear constraint, as we show in Section 3.4. Once the trifocal tensors have been computed, one can easily obtain the camera and fundamental matrices, as shown in Section 3.5. In Section 3.6, we show how to refine the estimates of the linear algorithm by extending the K-Means algorithm [4] to a mixture of trifocal tensors.

[2] Note that polynomial factorization may fail when $\mathcal{T}$ is computed from noisy correspondences, because the polynomial $q_n(x)$ will not perfectly factor as a product of linear forms.

3.1 Computing the Multibody Trifocal Tensor $\mathcal{T}$ and the Number of Independent Motions n

Recall from [14] that the trifocal tensor T associated with a single rigid-body motion can be computed linearly from the trilinear constraint (4) given at least 26 point-line-line correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{\ell}' \leftrightarrow \boldsymbol{\ell}''$ in general configuration. If instead we are given point-point-point correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x}' \leftrightarrow \boldsymbol{x}''$, then for each point $\boldsymbol{x}'$ in the second view, we can obtain two lines $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ passing through $\boldsymbol{x}'$ and, similarly, for the third view. Therefore, each correspondence gives in general four independent equations on T, and we only need seven point-point-point correspondences to linearly estimate T [14].

In the case of n rigid-body motions, the multibody trilinear constraint (7) is also linear in the multibody trifocal tensor $\mathcal{T}$. In fact, we may rewrite it as

$$(\widetilde{\boldsymbol{x}} \otimes \widetilde{\boldsymbol{\ell}'} \otimes \widetilde{\boldsymbol{\ell}''})^\top \boldsymbol{t} = 0, \tag{12}$$

where $\boldsymbol{t} \in \mathbb{R}^{M_n^3}$ is the stack of all the entries in $\mathcal{T}$, and $\otimes$ is the Kronecker product. Thus, if we are given $N \geq M_n^3 - 1$ point-line-line correspondences $\{\boldsymbol{x}_j \leftrightarrow \boldsymbol{\ell}'_j \leftrightarrow \boldsymbol{\ell}''_j\}_{j=1}^N$, we can solve for $\mathcal{T}$ linearly from (12), provided that the number of motions n is known. However, this requires a rather large number of correspondences: 26 correspondences for one motion, 215 for two motions, 999 for three motions, and so forth.

Fortunately, as in the case of a single rigid-body motion, we can significantly reduce the data requirements by working with point-point-point correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x}' \leftrightarrow \boldsymbol{x}''$. Since each point in the second view $\boldsymbol{x}'$ gives two lines $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ and each point in the third view $\boldsymbol{x}''$ gives two lines $\boldsymbol{\ell}''_1$ and $\boldsymbol{\ell}''_2$, a naive calculation would give $2^2 = 4$ linear equations in $\mathcal{T}$ per correspondence. However, due to the algebraic properties of the Veronese map, each correspondence provides in general $(n+1)^2$ independent constraints on $\mathcal{T}$.

To see this, remember that the multibody trilinear constraint is satisfied by *all* lines $\boldsymbol{\ell}' = \alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2$ and $\boldsymbol{\ell}'' = \beta\boldsymbol{\ell}''_1 + \boldsymbol{\ell}''_2$ passing through $\boldsymbol{x}'$ and $\boldsymbol{x}''$, respectively. Therefore, for all $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$, we must have

$$\left(\widetilde{\boldsymbol{x}} \otimes (\widetilde{\alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2}) \otimes (\widetilde{\beta\boldsymbol{\ell}''_1 + \boldsymbol{\ell}''_2})\right)^\top \boldsymbol{t} = 0. \tag{13}$$

This equation, viewed as a function of α, is a polynomial of degree n; hence, its $n+1$ coefficients must be zero. Each one of its coefficients is in turn a polynomial of degree n in β, whose $n+1$ coefficients must be zero. Therefore, each point-point-point correspondence gives $(n+1)^2$ constraints on the multibody trifocal tensor $\mathcal{T}$. We do not present an analytical proof that these $(n+1)^2$ constraints are in fact linearly independent. However, numerical examples show that this is indeed true for generic data. Exhibiting a single example for which the constraints are indeed linearly independent is enough to show that this is generically the case (that is, for almost all sets of input data).

In order to compute the multibody trifocal tensor, notice that after expanding $(\widetilde{\alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2})$ as $\sum_{j=0}^n \alpha^j C_j(\boldsymbol{\ell}'_1, \boldsymbol{\ell}'_2)$ and $(\widetilde{\beta\boldsymbol{\ell}''_1 + \boldsymbol{\ell}''_2})$ as $\sum_{k=0}^n \beta^k C_k(\boldsymbol{\ell}''_1, \boldsymbol{\ell}''_2)$, where $C_j(\boldsymbol{\ell}_1, \boldsymbol{\ell}_2) \in \mathbb{R}^{M_n}$, and substituting these expressions in (13), the $(n+1)^2$ constraints can be written explicitly as

$$(\widetilde{\boldsymbol{x}} \otimes C_j(\boldsymbol{\ell}'_1, \boldsymbol{\ell}'_2) \otimes C_k(\boldsymbol{\ell}''_1, \boldsymbol{\ell}''_2))^\top \boldsymbol{t} = 0 \; j, k = 0, \ldots, n. \tag{14}$$

Therefore, if we are given a set of $N \geq (M_n^3 - 1)/(n+1)^2$ point-point-point correspondences $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}'_i \leftrightarrow \boldsymbol{x}''_i\}_{i=1}^N$, we may generate a set of N line pairs $(\boldsymbol{\ell}'_{i1}, \boldsymbol{\ell}'_{i2})$ and $(\boldsymbol{\ell}''_{i1}, \boldsymbol{\ell}''_{i2})$ passing through $\boldsymbol{x}'_i$ and $\boldsymbol{x}''_i$, respectively, and solve for the multibody trifocal tensor $\mathcal{T}$ from the system of linear equations:

$$\boldsymbol{V}_n \boldsymbol{t} \doteq \begin{bmatrix} (\widetilde{\boldsymbol{x}}_1 \otimes C_0(\boldsymbol{\ell}'_{11}, \boldsymbol{\ell}'_{12}) \otimes C_0(\boldsymbol{\ell}''_{11}, \boldsymbol{\ell}''_{12}))^\top \\ \vdots \\ (\widetilde{\boldsymbol{x}}_1 \otimes C_n(\boldsymbol{\ell}'_{11}, \boldsymbol{\ell}'_{12}) \otimes C_n(\boldsymbol{\ell}''_{11}, \boldsymbol{\ell}''_{12}))^\top \\ \vdots \\ (\widetilde{\boldsymbol{x}_N} \otimes C_0(\boldsymbol{\ell}'_{N1}, \boldsymbol{\ell}'_{N2}) \otimes C_0(\boldsymbol{\ell}''_{N1}, \boldsymbol{\ell}''_{N2}))^\top \\ \vdots \\ (\widetilde{\boldsymbol{x}_N} \otimes C_n(\boldsymbol{\ell}'_{N1}, \boldsymbol{\ell}'_{N2}) \otimes C_n(\boldsymbol{\ell}''_{N1}, \boldsymbol{\ell}''_{N2}))^\top \end{bmatrix} \boldsymbol{t} = 0. \tag{15}$$

Note that we need only $(M_n^3 - 1)/(n+1)^2$ point-point-point correspondences to linearly estimate $\mathcal{T}$. That is, 7, 24, and 63 correspondences for one, two, and three motions, respectively. This represents a significant improvement not only with respect to the case of point-line-line correspondences but also with respect to the case of two perspective views, which requires $M_n^2 - 1$ point-point correspondences to linearly estimate the multibody fundamental matrix [32], that is, 8, 35, and 99 correspondences for one, two, and three motions, respectively. Table 1 gives the minimum number of point correspondences as a function of the number of motions. Of course, with noisy data, it is better to use many more correspondences than the minimum required.

TABLE 1
Minimum Number of Point-Point-Point Correspondences as a Function of the Number of Motion Models

n	1	2	3	4	5
Multibody fundamental matrix	8	35	99	224	440
Multibody trifocal tensor	7	24	63	135	258

Although the multibody trilinear constraint is linear in the multibody trifocal tensor $\mathcal{T}$, we cannot solve for $\mathcal{T}$ without knowing the number of motions in advance, because (15) depends explicitly on n. However, in order for this linear system to have a unique solution, we must have $\operatorname{rank}(\boldsymbol{V}_n) = M_n^3 - 1$. As it turns out, this rank constraint provides us with a method for computing the number of independent motions. This is because the multibody trilinear constraint of degree n is the polynomial of minimal degree that fits the given data. This implies that (15) has no solution if n is less than the true number of motions, a unique solution if n equals the true number of motions, and more than one solution otherwise, that is

$$\operatorname{rank}(\boldsymbol{V}_i) \begin{cases} > M_i^3 - 1, & \text{if } i < n \\ = M_i^3 - 1, & \text{if } i = n \\ < M_i^3 - 1, & \text{if } i > n. \end{cases} \tag{16}$$

Therefore, the number of independent motions is given by

$$n \doteq \min\{i : \operatorname{rank}(\boldsymbol{V}_i) = M_i^3 - 1\}. \tag{17}$$

Clearly, this formula for the number of motions is useful only if the correspondences are noiseless, because with noisy image measurements the matrix $\boldsymbol{V}_i$ may be full rank for all i. An extremely simple way for computing the number of motions n from a noisy matrix $\boldsymbol{V}_i$ is

$$n = \arg\min_{i\geq 1} \frac{\sigma^2_{M_i^3}(\boldsymbol{V}_i)}{\sum_{k=1}^{M_i^3-1} \sigma_k^2(\boldsymbol{V}_i)} + \mu M_i^3, \tag{18}$$

where $\sigma_k(\boldsymbol{V}_i)$ is the kth singular value of $\boldsymbol{V}_i$, and μ is a parameter. The formula in (18) for estimating n is motivated by model selection techniques [16] in which one minimizes a cost function that consists of a data fitting term and a model complexity term. The data fitting term measures how well the data is approximated by the model—in this case, how close the matrix $\boldsymbol{V}_i$ is to dropping rank by one. The model complexity term penalizes choosing models of high complexity—in this case, choosing a large rank.

We summarize the results so far with the following linear algorithm for estimating the multibody trifocal tensor.

Algorithm 1. (Computing the multibody trifocal tensor $\mathcal{T}$) Given $N \geq (M_n^3 - 1)/(n+1)^2$ point-point-point correspondences $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}'_i \leftrightarrow \boldsymbol{x}''_i\}_{i=1}^N$ in general configuration, compute $\mathcal{T}$ as follows:

1. For $i = 1, \dots, N$, generate line pairs $(\boldsymbol{\ell}'_{i1}, \boldsymbol{\ell}'_{i2})$ and $(\boldsymbol{\ell}''_{i1}, \boldsymbol{\ell}''_{i2})$ passing through $\boldsymbol{x}'_i$ and $\boldsymbol{x}''_i$, respectively.
2. For $n = 1, 2, \dots,$ form the matrix $\boldsymbol{V}_n \in \mathbb{R}^{N(n+1)^2 \times M_n^3}$ whose rows are $\widetilde{\boldsymbol{x}}_i \otimes C_j(\boldsymbol{\ell}'_{i1}, \boldsymbol{\ell}'_{i2}) \otimes C_k(\boldsymbol{\ell}''_{i1}, \boldsymbol{\ell}''_{i2}) \in \mathbb{R}^{M_n^3}$ for all $i = 1, \dots, N$ and $j, k = 0, \dots, n$ and determine the number of motions as in (18).
3. Compute $\mathcal{T}$, interpreted as a vector in $\mathbb{R}^{M_n^3}$, as the singular vector of $\boldsymbol{V}_n$ associated with its smallest singular value.

Notice that Algorithm 1 is the same as the well-known linear *seven-point algorithm* for estimating the trifocal tensor T [14]. The only difference is that we need to generate more than two equations per point in the second and third views $\boldsymbol{x}'$ and $\boldsymbol{x}''$ in order to build the data matrix, whose null space is the multibody trifocal tensor.

3.2 Computing Epipolar Lines

Given the trifocal tensor T, it is well known how to compute the epipolar lines in the second and third views of a point $\boldsymbol{x}$ in the first view [14]. Specifically, notice from (3) that the matrix

$$M_{\boldsymbol{x}} = (\boldsymbol{x}T) = (R'\boldsymbol{x}\boldsymbol{e}''^{\top} - \boldsymbol{e}'\boldsymbol{x}^{\top}R''^{\top}) \in \mathbb{R}^{3\times 3} \tag{19}$$

has rank 2. In fact, its left null space is $\boldsymbol{\ell}'_{\boldsymbol{x}} = \boldsymbol{e}' \times (R'\boldsymbol{x})$ and its right null space is $\boldsymbol{\ell}''_{\boldsymbol{x}} = \boldsymbol{e}'' \times (R''\boldsymbol{x})$, that is, the epipolar lines of $\boldsymbol{x}$ in the second and third views, respectively. In brief

Lemma 1. *The epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$ in the second view corresponding to a point $\boldsymbol{x}$ in the first view is the line such that $\boldsymbol{x}\boldsymbol{\ell}'_{\boldsymbol{x}}T = \mathbf{0}$. Similarly, the epipolar line $\boldsymbol{\ell}''_{\boldsymbol{x}}$ in the third view is the line satisfying $\boldsymbol{x}\boldsymbol{\ell}''_{\boldsymbol{x}}T = \mathbf{0}$. Therefore,* $\operatorname{rank}(\boldsymbol{x}T) = 2$.

In the case of multiple motions, we are faced with the more challenging problem of computing the epipolar lines $\boldsymbol{\ell}'_{\boldsymbol{x}}$ and $\boldsymbol{\ell}''_{\boldsymbol{x}}$ without knowing the individual trifocal tensors $\{T_i\}_{i=1}^n$ or the clustering of the correspondences. The question is then how to compute such epipolar lines from the multibody trifocal tensor $\mathcal{T}$. To this end, we notice that with each point in the first view $\boldsymbol{x}$, we can associate n epipolar lines $\{\boldsymbol{\ell}'_{i\boldsymbol{x}}\}_{i=1}^n$, each one of them corresponding to one of the n motions between the first and second views (see Fig. 4). We thus have $\boldsymbol{x}\boldsymbol{\ell}'_{i\boldsymbol{x}}T_i = \mathbf{0}$, which implies that for *any* line $\boldsymbol{\ell}''$ in the third view, $\boldsymbol{x}\boldsymbol{\ell}'_{i\boldsymbol{x}}\boldsymbol{\ell}''T_i = 0$. Now, since the span of $\widetilde{\boldsymbol{\ell}''}$ for all $\boldsymbol{\ell}'' \in \mathbb{R}^3$ is $\mathbb{R}^{M_n}$, we have that for all $i = 1, \dots, n$

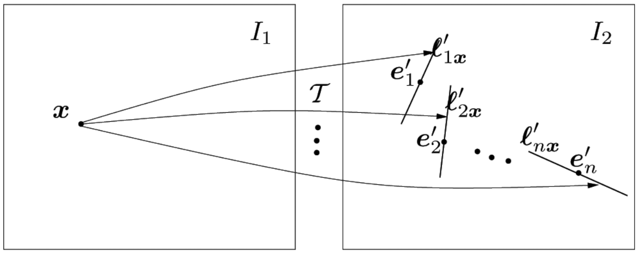

I_1
I_2
$\boldsymbol{x}$
$\mathcal{T}$
$\boldsymbol{\ell}'_{1\boldsymbol{x}}$
$\boldsymbol{e}'_1$
$\boldsymbol{\ell}'_{2\boldsymbol{x}}$
$\boldsymbol{e}'_2$
$\boldsymbol{\ell}'_{n\boldsymbol{x}}$
$\boldsymbol{e}'_n$

Fig. 4. The multibody trifocal tensor $\mathcal{T}$ maps each point x in the first image to n epipolar lines $\ell_{1x}, \dots, \ell_{nx}$ that pass through the n epipoles $e'_1, \dots, e'_n$, respectively. One of these epipolar lines passes through x'.

$$\forall \boldsymbol{\ell}'' \left[\prod_{k=1}^n (\boldsymbol{x}\boldsymbol{\ell}'_{i\boldsymbol{x}}\boldsymbol{\ell}''T_k) = (\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}'_{i\boldsymbol{x}}}\widetilde{\boldsymbol{\ell}''}\mathcal{T}) = 0\right] \iff \left(\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}'_{i\boldsymbol{x}}}\mathcal{T} = \mathbf{0}\right).$$

Since the vectors $\widetilde{\boldsymbol{\ell}'_{i\boldsymbol{x}}}$ are linearly independent when $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ are pairwise different in $\mathbb{P}^2$ (see [32]), the matrix $\widetilde{\boldsymbol{x}}\mathcal{T}$ has in general at least n vectors in its left null space. Therefore,

Theorem 3. *If $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ and $\boldsymbol{\ell}''_{i\boldsymbol{x}}$ are the epipolar lines in the second and third views corresponding under the ith motion to a point $\boldsymbol{x}$ in the first view, then $\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}'_{i\boldsymbol{x}}}\mathcal{T} = \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}''_{i\boldsymbol{x}}}\mathcal{T} = \mathbf{0} \in \mathbb{R}^{M_n}$. Thus,* $\operatorname{rank}(\widetilde{\boldsymbol{x}}\mathcal{T}) \leq M_n - n$ *if the epipolar lines are different.*

This result alone does not help us to find $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ according to a given motion, since any one of the n epipolar lines $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ will satisfy the conditions of Theorem 3. In fact, this question of determining the epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$ corresponding to a point $\boldsymbol{x}$ is not well posed as such, since the epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$ depends on which of the n motions the point $\boldsymbol{x}$ belongs to, which cannot be determined without additional information. We therefore pose the question a little differently and suppose that we know the point $\boldsymbol{x}'$ in the second view corresponding to $\boldsymbol{x}$ and wish to find the epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$ also in the second view. This epipolar line must of course pass through $\boldsymbol{x}'$.

To solve for $\boldsymbol{\ell}'_{\boldsymbol{x}}$, notice that $\boldsymbol{\ell}'_{\boldsymbol{x}}$ can be parameterized as

$$\boldsymbol{\ell}'_{\boldsymbol{x}} = \alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2, \tag{20}$$

where, as before, $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ are two different lines passing through $\boldsymbol{x}'$. From Theorem 3, we have that for some $\alpha \in \mathbb{R}$

$$\widetilde{\boldsymbol{x}}\widetilde{(\alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2)}\mathcal{T} = \mathbf{0}. \tag{21}$$

Each of the M_n components of this vector is a polynomial of degree n in α. These polynomials must have a common root α^* for which all the polynomials (and, hence, the vector) vanish. The epipolar line of $\boldsymbol{x}$ in the second view is then $\boldsymbol{\ell}'_{\boldsymbol{x}} = \alpha^*\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2$. We thus have the following algorithm for computing epipolar lines from the multibody fundamental tensor.

Algorithm 2. (Computing epipolar lines from $\mathcal{T}$) Given a point-point-point correspondence $\boldsymbol{x} \leftrightarrow \boldsymbol{x}' \leftrightarrow \boldsymbol{x}''$

1. Choose two different lines $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ passing through $\boldsymbol{x}'$. Build the polynomial vector $q'(\alpha) = \widetilde{\boldsymbol{x}}\widetilde{(\alpha\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2)}\mathcal{T}$. Compute the common root α^* of these M_n polynomials as the real root of the derivative of $f'(\alpha) = \sum_{k=1}^{M_n} q'_k(\alpha)^2$ that minimizes $f'(\alpha)$. The epipolar line of $\boldsymbol{x}$ in the second view is given by $\boldsymbol{\ell}'_{\boldsymbol{x}} = \alpha^*\boldsymbol{\ell}'_1 + \boldsymbol{\ell}'_2$.
2. Given a correspondence $\boldsymbol{x} \leftrightarrow \boldsymbol{x}''$, determine the epipolar line of $\boldsymbol{x}$ in the third view, $\boldsymbol{\ell}''_{\boldsymbol{x}}$, in an analogous way.

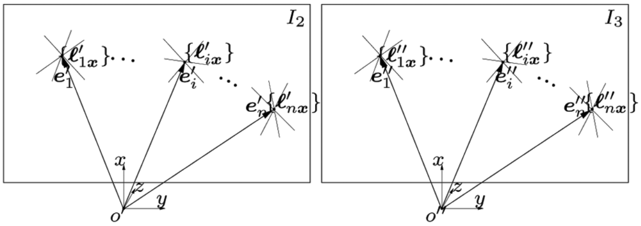

I_2
$\{\boldsymbol{\ell}'_{1\boldsymbol{x}}\}\cdots$
$\{\boldsymbol{\ell}'_{i\boldsymbol{x}}\}$
$\boldsymbol{e}'_1$
$\boldsymbol{e}'_i$
$\boldsymbol{e}'_n\{\boldsymbol{\ell}'_{n\boldsymbol{x}}\}$
x
z
y
o
I_3
$\{\boldsymbol{\ell}''_{1\boldsymbol{x}}\}\cdots$
$\{\boldsymbol{\ell}''_{i\boldsymbol{x}}\}$
$\boldsymbol{e}''_1$
$\boldsymbol{e}''_i$
$\boldsymbol{e}''_n\{\boldsymbol{\ell}''_{n\boldsymbol{x}}\}$
x
z
y
o''

Fig. 5. When n objects move independently, the epipolar lines in the second and third views associated with each image point in the first view form n groups and intersect, respectively, at n epipoles in the second and third views.

We may apply Algorithm 2 to all point correspondences $\{\boldsymbol{x}_j \leftrightarrow \boldsymbol{x}'_j \leftrightarrow \boldsymbol{x}''_j\}_{j=1}^N$ and obtain the set of all N epipolar lines in the second and third views according to the motion associated with each correspondence. Notice again that these epipolar lines are obtained from the multibody trifocal tensor only, without yet knowing the individual trifocal tensors or the clustering of the given correspondences.

It is also useful to note that the only property of $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ that we used in Algorithm 2 was that the desired epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$ could be expressed as a linear combination of $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$. If instead we knew the epipole corresponding to the required motion, then we could choose $\boldsymbol{\ell}'_1$ and $\boldsymbol{\ell}'_2$ to be any two lines passing through the epipole and apply Algorithm 2 to determine the epipolar line $\boldsymbol{\ell}'_{\boldsymbol{x}}$.

Once the epipoles and epipolar lines are known, we may segment the data into multiple groups and subsequently determine the fundamental matrices and trifocal tensors for each group (see Section 3.4). Before proceeding, we need to show how to determine the epipoles, which we do in the next section.

3.3 Computing Epipoles

In the case of one rigid-body motion, the epipoles in the second and third views $\boldsymbol{e}'$ and $\boldsymbol{e}''$ must lie in the epipolar lines in the second and third views, $\{\boldsymbol{\ell}'_{\boldsymbol{x}_j}\}_{j=1}^N$ and $\{\boldsymbol{\ell}''_{\boldsymbol{x}_j}\}_{j=1}^N$, respectively. Thus, we can obtain the epipoles from

$$\boldsymbol{e}'^\top\left[\boldsymbol{\ell}'_{\boldsymbol{x}_1},\ldots,\boldsymbol{\ell}'_{\boldsymbol{x}_N}\right] = \boldsymbol{0}^\top \text{ and } \boldsymbol{e}''^\top\left[\boldsymbol{\ell}''_{\boldsymbol{x}_1},\ldots,\boldsymbol{\ell}''_{\boldsymbol{x}_N}\right] = \boldsymbol{0}^\top. \tag{22}$$

Clearly, we only need two epipolar lines to determine the epipoles; hence, we do not need to compute the epipolar lines for all points in the first view. However, it is better to use more than two lines in the presence of noise.

In the case of n motions, there exist n epipole pairs $\{(\boldsymbol{e}'_i, \boldsymbol{e}''_i)\}_{i=1}^n$, where $\boldsymbol{e}'_i$ and $\boldsymbol{e}''_i$ are epipoles in the second and third views corresponding to the ith motion. Given a set of point correspondences, we may compute the multibody trifocal tensor $\mathcal{T}$ and determine the epipolar lines $\boldsymbol{\ell}'_{\boldsymbol{x}_j}$ and $\boldsymbol{\ell}''_{\boldsymbol{x}_j}$ associated with each correspondence $\{\boldsymbol{x}_j \leftrightarrow \boldsymbol{x}'_j \leftrightarrow \boldsymbol{x}''_j\}$ using Algorithms 1 and 2. Then, for each pair of epipolar lines $(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})$, there exists an epipole pair $(\boldsymbol{e}'_i, \boldsymbol{e}''_i)$ such that

$$\boldsymbol{e}'^\top_i\boldsymbol{\ell}'_{\boldsymbol{x}_j} = 0 \quad \text{and} \quad \boldsymbol{e}''^\top_i\boldsymbol{\ell}''_{\boldsymbol{x}_j} = 0 \tag{23}$$

as illustrated in Fig. 5. Our task is twofold: First, we need to find the set of epipole pairs $\{(\boldsymbol{e}'_i, \boldsymbol{e}''_i)\}_{i=1}^n$. Second, we need to determine which pair of epipoles lie in the epipolar lines $(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})$ derived from a given point correspondence.

If two point correspondences $\boldsymbol{x}_j \leftrightarrow \boldsymbol{x}'_j \leftrightarrow \boldsymbol{x}''_j$ and $\boldsymbol{x}_k \leftrightarrow \boldsymbol{x}'_k \leftrightarrow \boldsymbol{x}''_k$ both belong to the same motion, then the pair of epipoles can be determined easily by intersecting the epipolar lines. If the two motions are different, then the intersection points of the epipolar lines will have no geometric meaning and will be essentially arbitrary. This suggests an approach to determining the epipoles based on RANSAC [7] in which we intersect pairs of epipolar lines to find candidate epipoles and determine their degree of support among the other point correspondences. This method is expected to be effective with small numbers of motions.

In reality, we used a different method based on the idea of *multibody epipoles* proposed in [32] for the case of two views, which we now extend and modify for the case of three views. Notice from (23) that regardless of the motion associated with each pair of epipolar lines, we must have

$$\prod_{i=1}^n(\boldsymbol{e}'^\top_i\boldsymbol{\ell}'_{\boldsymbol{x}}) = \boldsymbol{c}'^\top\widetilde{\boldsymbol{\ell}'_{\boldsymbol{x}}} = 0 \wedge \prod_{i=1}^n(\boldsymbol{e}''^\top_i\boldsymbol{\ell}''_{\boldsymbol{x}}) = \boldsymbol{c}''^\top\widetilde{\boldsymbol{\ell}''_{\boldsymbol{x}}} = 0, \tag{24}$$

where the *multibody epipoles* $\boldsymbol{c}' \in \mathbb{R}^{M_n}$ and $\boldsymbol{c}'' \in \mathbb{R}^{M_n}$ are the coefficients of the homogeneous polynomials of degree n

$$p'(\boldsymbol{\ell}'_{\boldsymbol{x}}) = \boldsymbol{c}'^\top\widetilde{\boldsymbol{\ell}'_{\boldsymbol{x}}} \quad \text{and} \quad p''(\boldsymbol{\ell}''_{\boldsymbol{x}}) = \boldsymbol{c}''^\top\widetilde{\boldsymbol{\ell}''_{\boldsymbol{x}}}, \tag{25}$$

respectively.[3] Similar to (22), we may obtain the multibody epipoles from

$$\boldsymbol{c}'^\top[\widetilde{\boldsymbol{\ell}'_{\boldsymbol{x}_1}},\ldots,\widetilde{\boldsymbol{\ell}'_{\boldsymbol{x}_N}}] = \boldsymbol{0}^\top \text{ and } \boldsymbol{c}''^\top[\widetilde{\boldsymbol{\ell}''_{\boldsymbol{x}_1}},\ldots,\widetilde{\boldsymbol{\ell}''_{\boldsymbol{x}_N}}] = \boldsymbol{0}^\top. \tag{26}$$

Clearly, we only need $M_n - 1$ epipolar lines to determine the multibody epipoles. However, it is better to use more than $M_n - 1$ epipolar lines in the presence of noise.

In order to estimate the epipoles, we notice that if a pair of epipolar lines $(\boldsymbol{\ell}'_{\boldsymbol{x}}, \boldsymbol{\ell}''_{\boldsymbol{x}})$ corresponds to the ith motion, then the derivatives of p' and p'' at the pair $(\boldsymbol{\ell}'_{\boldsymbol{x}}, \boldsymbol{\ell}''_{\boldsymbol{x}})$ give the epipoles $\boldsymbol{e}'_i$ and $\boldsymbol{e}''_i$, that is,

$$\frac{\partial}{\partial\boldsymbol{\ell}'_{\boldsymbol{x}}}(\boldsymbol{c}'^\top\widetilde{\boldsymbol{\ell}'_{\boldsymbol{x}}}) \sim \boldsymbol{e}'_i \quad \text{and} \quad \frac{\partial}{\partial\boldsymbol{\ell}''_{\boldsymbol{x}}}(\boldsymbol{c}''^\top\widetilde{\boldsymbol{\ell}''_{\boldsymbol{x}}}) \sim \boldsymbol{e}''_i. \tag{27}$$

Therefore, in order to estimate the n epipole pairs from the multibody epipoles, we only need to find n pairs of epipolar lines, one per motion class, and then evaluate the derivatives of p' and p'' at those pairs of epipolar lines. The first pair can be chosen to minimize the sum of squared distances to the epipoles. From [31], we know that the distance from a point $\boldsymbol{x}$ to a line $\boldsymbol{b}_i^\top\boldsymbol{x} = 0$ is given by $|p(\boldsymbol{x})|/\|\nabla p(\boldsymbol{x})\| + O(\|\boldsymbol{x}\|^2)$, where $p(\boldsymbol{x}) = (\boldsymbol{b}_1^\top\boldsymbol{x})\cdots(\boldsymbol{b}_n^\top\boldsymbol{x})$. Thus, we choose the pair of lines $(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})$ that minimizes

$$\frac{|p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})|^2}{\|\nabla p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})\|^2} + \frac{|p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})|^2}{\|\nabla p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})\|^2}, \quad j = 1,\ldots,N. \tag{28}$$

The remaining $n-1$ pairs of epipolar lines can be chosen in an analogous fashion, except that we need to penalize choosing pairs from the motion groups that have already been chosen. For $i = n-1 : 1$, this can be done by choosing the pair of epipolar lines $(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})$, $j = 1,\ldots,N$, that minimizes

[3] More specifically, $\boldsymbol{c}'$ and $\boldsymbol{c}''$ are a vector representation of the symmetric tensor product of the epipoles $\{\boldsymbol{e}'_i\}_{i=1}^n$ and $\{\boldsymbol{e}''_i\}_{i=1}^n$, as shown in [31].

$$\frac{\frac{|p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})|^2}{\|\nabla p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})\|^2}}{\prod_{k=i+1}^{n} |\boldsymbol{e}'^{\top}_k \boldsymbol{\ell}'_{\boldsymbol{x}_j}|^2} + \frac{\frac{|p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})|^2}{\|\nabla p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})\|^2}}{\prod_{k=i+1}^{n} |\boldsymbol{e}''^{\top}_k \boldsymbol{\ell}''_{\boldsymbol{x}_j}|^2}, \qquad (29)$$

where the epipoles pairs $(\boldsymbol{e}'_k, \boldsymbol{e}''_k)$ are computed as in (27).

We therefore have the following algorithm for computing the epipoles from a set of epipolar lines.

Algorithm 3. (Computing epipoles from $\mathcal{T}$)
Given a set of epipolar lines $\{(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})\}_{j=1}^N$

1. Compute the multibody epipoles $\boldsymbol{c}'$ and $\boldsymbol{c}''$ from (26).
2. Compute the epipole pairs $\{(\boldsymbol{e}'_i, \boldsymbol{e}''_i)\}_{i=1}^n$ as

$$\boldsymbol{e}'_i \sim \nabla p'(\boldsymbol{\ell}'_{\boldsymbol{x}_{j_i}}) \qquad \text{and} \qquad \boldsymbol{e}''_i \sim \nabla p''(\boldsymbol{\ell}''_{\boldsymbol{x}_{j_i}}),$$

where $p'(\boldsymbol{\ell}') = \boldsymbol{c}'^{\top}\widetilde{\boldsymbol{\ell}}'$, $p''(\boldsymbol{\ell}'') = \boldsymbol{c}''^{\top}\widetilde{\boldsymbol{\ell}}''$, and for $i = n : 1$

$$j_i = \arg\min_{j=1,\dots,N} \frac{\frac{|p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})|^2}{\|\nabla p'(\boldsymbol{\ell}'_{\boldsymbol{x}_j})\|^2}}{\prod_{k=i+1}^{n} |\boldsymbol{e}'^{\top}_k \boldsymbol{\ell}'_{\boldsymbol{x}_j}|^2} + \frac{\frac{|p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})|^2}{\|\nabla p''(\boldsymbol{\ell}''_{\boldsymbol{x}_j})\|^2}}{\prod_{k=i+1}^{n} |\boldsymbol{e}''^{\top}_k \boldsymbol{\ell}''_{\boldsymbol{x}_j}|^2}.$$

Remark 1 (Computing derivatives). Note that given $\boldsymbol{c}$, the computation of the derivatives of $p(\boldsymbol{\ell}) = \boldsymbol{c}^{\top}\widetilde{\boldsymbol{\ell}}$ can be done algebraically, that is, it does not involve taking derivatives of the (possibly noisy) data. For instance, one may compute $\frac{\partial p(\boldsymbol{\ell})}{\partial \ell_k}$ as $\boldsymbol{c}^{\top} E_{nk} \widetilde{\boldsymbol{\ell}}^{n-1}$, where $E_{nk} \in \mathbb{R}^{M_n \times M_{n-1}}$ is a constant matrix that depends on the exponents of $\widetilde{\boldsymbol{\ell}}$, and $\widetilde{\boldsymbol{\ell}}^{n-1} \in \mathbb{R}^{M_{n-1}}$ is a vector containing all monomials of degree $n-1$ in $\boldsymbol{\ell}$.

3.4 Computing Individual Trifocal Tensors

Given epipolar lines and epipoles, we can immediately segment the point correspondences $\{\boldsymbol{x}_j \leftrightarrow \boldsymbol{x}'_j \leftrightarrow \boldsymbol{x}''_j\}_{j=1}^N$ into n groups using the distance from epipoles to epipolar lines as a criterion. We simply assign point j to motion i if

$$i = \arg\min_{\ell=1,\dots n} (\boldsymbol{e}'^{\top}_{\ell} \boldsymbol{\ell}'_{\boldsymbol{x}_j})^2 + (\boldsymbol{e}''^{\top}_{\ell} \boldsymbol{\ell}''_{\boldsymbol{x}_j})^2. \qquad (30)$$

Once the correspondences have been segmented, we can compute the individual trifocal tensors $\{T_i\}_{i=1}^n$ by applying the seven-point algorithm to the point correspondences associated with each one of the n motion groups.

In this section, we demonstrate that one can estimate the individual trifocal tensors *without* first clustering the image correspondences. The key is to look at the second-order derivatives of the multibody trilinear constraint. Therefore, we contend that *all* the geometric information about the multiple motions is already encoded in the multibody trifocal tensor.

Let $\boldsymbol{x}$ be an arbitrary point in $\mathbb{P}^2$ (not necessarily a point in the first view). Since the ith epipole $\boldsymbol{e}'_i$ is known, we can compute two lines $\boldsymbol{\ell}'_{i1}$ and $\boldsymbol{\ell}'_{i2}$ passing through $\boldsymbol{e}'_i$ and apply Algorithm 2 to compute the epipolar line of $\boldsymbol{x}$ in the second view $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ according to the ith motion. In a completely analogous fashion, we can compute the epipolar line of $\boldsymbol{x}$ in the third view $\boldsymbol{\ell}''_{i\boldsymbol{x}}$ from two lines passing through $\boldsymbol{e}''_i$. Given $(\boldsymbol{\ell}'_{i\boldsymbol{x}}, \boldsymbol{\ell}''_{i\boldsymbol{x}})$, the slices of the trifocal tensor T_i can be expressed in terms of the second derivative of the multibody epipolar constraint, as stated by the following theorem.

Theorem 4 (Slices of the trifocal tensors from the second-order derivatives of the multibody trilinear constraint). *The second-order derivative of the multibody trilinear constraint with respect to the second and third arguments evaluated at $(\boldsymbol{x}, \boldsymbol{\ell}'_{i\boldsymbol{x}}, \boldsymbol{\ell}''_{i\boldsymbol{x}})$ gives the matrix $M_{i\boldsymbol{x}} \sim \boldsymbol{x}T_i \in \mathbb{R}^{3\times 3}$, that is,*

$$\left.\frac{\partial^2(\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}}'\widetilde{\boldsymbol{\ell}}''\mathcal{T})}{\partial\boldsymbol{\ell}'\partial\boldsymbol{\ell}''}\right|_{(\boldsymbol{x},\boldsymbol{\ell}'_{i\boldsymbol{x}},\boldsymbol{\ell}''_{i\boldsymbol{x}})} = M_{i\boldsymbol{x}} \sim \boldsymbol{x}T_i \in \mathbb{R}^{3\times 3}. \qquad (31)$$

Proof. A simple calculation shows that

$$\frac{\partial^2(\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}}'\widetilde{\boldsymbol{\ell}}''\mathcal{T})}{\partial\boldsymbol{\ell}'\partial\boldsymbol{\ell}''} = \sum_{j=1}^{n}(\boldsymbol{x}T_j)\prod_{k\neq j}(\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T_k) + \sum_{j=1}^{n}(\boldsymbol{x}\boldsymbol{\ell}'T_j)\sum_{k\neq j}(\boldsymbol{x}\boldsymbol{\ell}''T_k)\prod_{\ell\neq k}(\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T_{\ell}).$$

Since $\boldsymbol{\ell}'_{i\boldsymbol{x}}$ and $\boldsymbol{\ell}''_{i\boldsymbol{x}}$ are epipolar lines associated with the ith motion, then $\boldsymbol{x}\boldsymbol{\ell}'_{i\boldsymbol{x}}T_i = \boldsymbol{x}\boldsymbol{\ell}''_{i\boldsymbol{x}}T_i = 0$. Therefore,

$$\left.\frac{\partial^2(\widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{\ell}}'\widetilde{\boldsymbol{\ell}}''\mathcal{T})}{\partial\boldsymbol{\ell}'\partial\boldsymbol{\ell}''}\right|_{(\boldsymbol{x},\boldsymbol{\ell}'_{i\boldsymbol{x}},\boldsymbol{\ell}''_{i\boldsymbol{x}})} = (\boldsymbol{x}T_i)\prod_{j\neq i}(\boldsymbol{x}\boldsymbol{\ell}'\boldsymbol{\ell}''T_j) \sim (\boldsymbol{x}T_i).$$

□

Thanks to (31), we can immediately outline an algorithm for computing the individual trifocal tensors.

Algorithm 4. (Computing trifocal tensors from $\mathcal{T}$)
Let $\{\boldsymbol{e}'_i, \boldsymbol{e}''_i\}_{i=1}^n$ be the set of epipoles in the second and third views. Also, let $\{\boldsymbol{x}_j\}_{j=1}^N$ be a set of $N \geq 4$ randomly chosen points. For $i = 1, \dots, n$, do the following:

1. Use Algorithm 2 to obtain the epipolar lines of $\boldsymbol{x}_j$ in the second and third views $\boldsymbol{\ell}'_{i\boldsymbol{x}_j}$ and $\boldsymbol{\ell}'_{i\boldsymbol{x}_j}$ from the epipoles $\boldsymbol{e}'_i$ and $\boldsymbol{e}''_i$, respectively.
2. Use (31) to obtain $M_{i\boldsymbol{x}_j}$, the slice of T_i along $\boldsymbol{x}_j$.
3. Solve for T_i for $i = 1, \dots, n$ from the set of linear equations

$$M_{i\boldsymbol{x}_j} \sim \boldsymbol{x}_j T_i \quad j = 1, \dots, N.$$

Once the individual trifocal tensors have been computed, one may cluster the correspondences by assigning point j to the trifocal tensor T_i that minimizes the algebraic error, that is,

$$i = \arg\min_{\ell=1,,n} \sum_{k=1}^{2}\sum_{m=1}^{2}(\boldsymbol{x}_j \boldsymbol{l}'_{jk} \boldsymbol{l}''_{jm} T_{\ell})^2. \qquad (32)$$

3.5 Computing Fundamental and Camera Matrices

Given the epipoles $\boldsymbol{e}'_i$ and $\boldsymbol{e}''_i$ and the trifocal tensor T_i, the computation of fundamental and camera matrices proceeds as follows [14]. We notice that for all $\boldsymbol{x} \in \mathbb{P}^2$:

$$[\boldsymbol{e}'_i]_{\times} M_{i\boldsymbol{x}} \boldsymbol{e}''_i = [\boldsymbol{e}'_i]_{\times}(R'_i \boldsymbol{x} \boldsymbol{e}''^{\top}_i - \boldsymbol{e}'_i \boldsymbol{x}^{\top} R''^{\top}_i)\boldsymbol{e}''_i \qquad (33)$$

$$\sim [\boldsymbol{e}'_i]_{\times} R'_i \boldsymbol{x} = F_{i21}\boldsymbol{x}. \qquad (34)$$

Therefore, we can obtain the fundamental matrices as

$$F_{i21} = [\boldsymbol{e}'_i]_{\times}[M_{i\boldsymbol{e}_1}\boldsymbol{e}''_i \; M_{i\boldsymbol{e}_2}\boldsymbol{e}''_i \; M_{i\boldsymbol{e}_3}\boldsymbol{e}''_i], \qquad (35)$$

$$F_{i31} = [\boldsymbol{e}''_i]_\times \left[M_{ie_1}^\top \boldsymbol{e}'_i\ M_{ie_2}^\top \boldsymbol{e}'_i\ M_{ie_3}^\top \boldsymbol{e}'_i\right], \tag{36}$$

where e_k for $k = 1, 2, 3$ are the standard basis for $\mathbb{R}^3$. Then, one can obtain the camera matrices up to a common projective transformation of the 3D space as

$$\mathtt{P}'_i = [M_{ie_1} \boldsymbol{e}''_i\ M_{ie_2} \boldsymbol{e}''_i\ M_{ie_3} \boldsymbol{e}''_i], \tag{37}$$

$$\mathtt{P}''_i = [\boldsymbol{e}''_i]^2_\times \left[M_{ie_1}^\top \boldsymbol{e}'\ M_{ie_2}^\top \boldsymbol{e}'_i\ M_{ie_3}^\top \boldsymbol{e}'_i\right]. \tag{38}$$

Once the camera matrices have been computed, one may cluster the correspondences using more sophisticated errors than the sum of distances to epipolar lines (30) or the algebraic distance to a trifocal tensor (32). For example, one may first reconstruct the 3D point $\boldsymbol{X}_j \in \mathbb{P}^2$ associated with $\boldsymbol{x}_j \leftrightarrow \boldsymbol{x}'_j \leftrightarrow \boldsymbol{x}''_j$ by triangulation [14], project $\boldsymbol{X}_j$ onto the three views using the camera matrices $\mathtt{P}_i$, $\mathtt{P}'_i$, and P''_i, and then assign point j to the motion i that minimizes the reprojection error

$$\|\boldsymbol{x}_j - \frac{\mathtt{P}_i \boldsymbol{X}_j}{e_3^\top \mathtt{P}_i \boldsymbol{X}_j}\|^2 + \|\boldsymbol{x}'_j - \frac{\mathtt{P}'_i \boldsymbol{X}_j}{e_3^\top \mathtt{P}'_i \boldsymbol{X}_j}\|^2 + \|\boldsymbol{x}''_j - \frac{\mathtt{P}''_i \boldsymbol{X}_j}{e_3^\top \mathtt{P}_i \boldsymbol{X}_j}\|^2. \tag{39}$$

3.6 Iterative Refinement

The motion segmentation algorithm we have proposed so far is purely geometric and provably correct in the absence of noise. Since most of the steps of the algorithm involve solving linear systems, the algorithm will also work with a moderate level of noise (as we will show in the experiments) provided that one solves each step in a least squares fashion. However, the results may be improved significantly by following the algebraic approach with an iterative refinement stage. In this section, we propose to refine the estimates of the trifocal tensors and the clustering of the correspondences by extending the classical K-Means algorithm [4] to a mixture of trifocal tensors model. We call the new algorithm *K-trifocal*.

Let $w_{ij} \in \{0, 1\}$ represent the assignment of the jth correspondence to the ith motion model, that is, $w_{ij} = 1$ if the jth point belongs to the ith motion and $w_{ij} = 0$ otherwise. We can solve for the trifocal tensors T_i and the segmentation of the correspondences $\{w_{ij}\}$ by minimizing the cost function

$$\sum_{j=1}^{N} \sum_{i=1}^{n} w_{ij} \epsilon_{ij}, \tag{40}$$

where ϵ_{ij} measures the error of point j to motion model i.

One possible choice for the ϵ_{ij} is the algebraic error $\epsilon_{ij} = \sum_{k=1}^{2} \sum_{m=1}^{2} (\boldsymbol{x}_j \boldsymbol{l}'_{jk} \boldsymbol{l}''_{jm} T_i)^2$. In this case, we can minimize (40) following a coordinate descent algorithm that alternates between computing (linearly) the trifocal tensors for each motion class and clustering the correspondences. More specifically, given an initial estimate for the segmentation of the correspondences, we alternate between the following two steps:

1. Given the segmentation of the correspondences $\{w_{ij}\}$, we compute the optimal solution for the trifocal tensors as the least squares solution of the set of linear equations:

$$\boldsymbol{x}_j \boldsymbol{l}'_{jk} \boldsymbol{l}''_{jm} T_i = 0 \qquad j \in \{j : w_{ij} = 1\}, \qquad k, m = 1, 2. \tag{41}$$

2. Given the trifocal tensors $\{T_i\}$, we compute the optimal solution for the segmentation of the correspondences as

$$w_{ij} = \begin{cases} 1 & \text{if } i = \arg\min\limits_{k=1,\dots,n} \epsilon_{kj} \\ 0 & \text{otherwise.} \end{cases} \tag{42}$$

As the iterations proceed, the cost function does not increase; hence, the algorithm converges to a local minimum of (40).

An alternative choice for ϵ_{ij} is the reprojection error (39). Given $\{T_i\}$, the computation of the optimal segmentation is as in (42), except that the calculation of ϵ_{ij} requires computing the camera matrices and triangulating the correspondences. However, given $\{w_{ij}\}$, the computation of the optimal trifocal tensors requires nonlinear optimization. For ease of computation, we still compute the trifocal tensors linearly as in (41).

3.7 Three-View Multibody Structure from Motion Algorithm

Algorithm 5 summarizes the main steps of the algorithm for segmenting trifocal tensors described in this section.

Algorithm 5. (Segmentation of trifocal tensors)
Given a set of points $\{(\boldsymbol{x}_j, \boldsymbol{x}'_j, \boldsymbol{x}''_j)\}_{j=1}^N$ corresponding to N 3D points undergoing n different rigid-body motions relative to a moving perspective camera, recover the number of independent motions n, the trifocal tensors $\{T_i\}_{i=1}^n$ associated with each motion, and the motion associated with each correspondence as follows:

1. **Number of motions.** Compute two lines $(\boldsymbol{\ell}'_{j1}, \boldsymbol{\ell}'_{j2})$ passing through $\boldsymbol{x}'_j$ and two lines $(\boldsymbol{\ell}''_{j1}, \boldsymbol{\ell}''_{j2})$ passing through $\boldsymbol{x}''_j$. Form the embedded data matrix of degree $i = 1, \dots, n$, $\boldsymbol{V}_i \in \mathbb{R}^{N(i+1)^2 \times M_i^3}$, as defined in (15). Compute the number of independent motions n from a rank constraint on $\boldsymbol{V}_i$ as in (18).
2. **Multibody trifocal tensor.** Compute the multibody trifocal tensor $\mathcal{T}$ from the null space of $\boldsymbol{V}_n$ using least squares.
3. **Epipolar lines.** For all $i = 1, \dots, N$, compute the epipolar lines of $\boldsymbol{x}_j$ in the second and third views, $\boldsymbol{\ell}'_{\boldsymbol{x}_j}$ and $\boldsymbol{\ell}''_{\boldsymbol{x}_j}$, from the common root of a set of univariate polynomials, as described in Algorithm 2.
4. **Epipoles.** Use the epipolar lines $\{(\boldsymbol{\ell}'_{\boldsymbol{x}_j}, \boldsymbol{\ell}''_{\boldsymbol{x}_j})\}_{j=1}^N$ to compute the multibody epipoles $\boldsymbol{c}'$ and $\boldsymbol{c}''$ from the linear systems in (26). Compute the epipole pairs $\{(e'_i, e''_i)\}_{i=1}^n$ from the gradients of $p'(\boldsymbol{\ell}') = \boldsymbol{c}'^\top \widetilde{\boldsymbol{\ell}'}$ and $p''(\boldsymbol{\ell}'') = \boldsymbol{c}''^\top \widetilde{\boldsymbol{\ell}''}$ as shown in Section 3.3.
5. **Feature clustering from epipoles and epipolar lines.** Assign point correspondence $(\boldsymbol{x}_j, \boldsymbol{x}'_j, \boldsymbol{x}''_j)$ to motion i according to (30).
6. **Individual trifocal tensors and camera matrices.** Compute the individual trifocal tensors $\{T_i\}_{i=1}^n$ by applying the seven-point algorithm to the groups obtained in Step 5. Compute the camera matrices $\mathtt{P}_i$, $\mathtt{P}'_i$, and $\mathtt{P}''_i$ from each T_i, as shown in Section 3.5.
7. **Feature clustering from trifocal tensors.** Use triangulation to find the 3D point $\boldsymbol{X}_j$ associated with each point $(\boldsymbol{x}_j, \boldsymbol{x}'_j, \boldsymbol{x}''_j)$. Assign $(\boldsymbol{x}_j, \boldsymbol{x}'_j, \boldsymbol{x}''_j)$ to motion i according to (39).

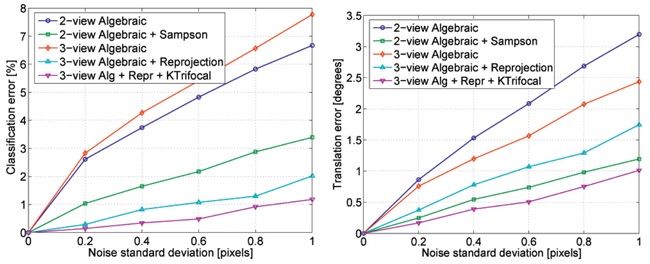


Fig. 6. Motion segmentation and motion estimation errors as a function of noise for $\tau = 100$ u.f.l. and $\theta = 0$ degree.

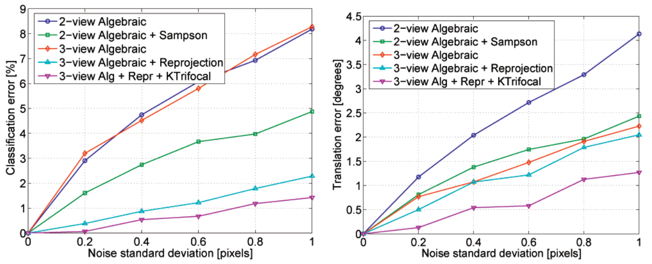


Fig. 7. Motion segmentation and motion estimation errors as a function of noise for $\tau = 100$ u.f.l. and $\theta = 5$ degrees.

8. **Iterative refinement.** Starting from the segmentation in Step 7, alternate between computing (linearly) the trifocal tensors for each motion class (41) and clustering the correspondences according to (30), as described in Section 3.6.

4 Experiments

In this section, we present several experiments on various synthetic and real sequences. The experiments compare our three-view algorithm and variations of it against two-view perspective and multiview affine algorithms. More specifically, we compare the following methods:

1. *Two-view perspective algorithms*
   a. **Algebraic (2-A).** This algorithm parallels Steps 2-5 of Algorithm 5 but uses fundamental matrices instead of trifocal tensors. See [32] for details.
   b. **Algebraic+Sampson (2-A+S).** This algorithm parallels Steps 2-7 of Algorithm 5 but uses fundamental matrices instead of trifocal tensors and the Sampson error instead of the reprojection error in Step 7. See [32] for details.
2. *Three-view perspective algorithms*
   a. **Algebraic (3-A).** This algorithm follows Steps 2-5 of Algorithm 5.
   b. **Algebraic+Reprojection (3-A+R).** This algorithm follows Steps 2-7 of Algorithm 5.
   c. **Algebraic+Reprojection+Ktrifocal (3-A+R+T).** This algorithm follows Steps 2-8 of Algorithm 5.
3. *Multiview affine algorithms*
   a. **Algebraic (3-Aff).** This algorithm assumes an affine projection model. Under this model, the trajectories of a point in F views live in a subspace of $\mathbb{R}^{2F}$ of dimension $d \leq 4$. Motion segmentation is then equivalent to clustering the motion subspaces. The algorithm uses GPCA to segment the motion subspaces. See [27] for details. In our experiments, we use $F = 3$ views to make the comparison fair.

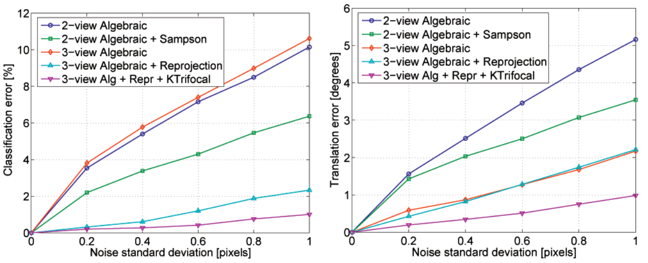


Fig. 8. Motion segmentation and motion estimation errors as a function of noise for $\tau = 70$ u.f.l. and $\theta = 5$ degrees.

4.1 Experiments on Synthetic Sequences

We first test our algorithm on synthetic data. We randomly generate two groups of 100 3D points each with a depth variation of 100-400 units of focal length (u.f.l.). These points are rotated and translated according to two rigid-body motions with a random axis of rotation and a random direction of translation. The interframe rotation is $\theta \in \{0, 5\}$ degrees, and the interframe translation is $\tau \in \{70, 100\}$ u.f.l. The third configuration of points is obtained by applying another pair of rigid-body motions to the same 3D points. The three views are obtained by perspective projection using an image size of $1,000 \times 1,000$ pixels. Zero-mean Gaussian noise with a standard deviation of $\sigma \in [0, 1]$ pixels is added to the so-obtained point correspondences in three views.

Figs. 6, 7, and 8 show the performance of perspective motion segmentation algorithms for $(\tau, \theta) = (100, 0)$, $(\tau, \theta) = (100, 5)$, and $(\tau, \theta) = (70, 5)$, respectively, as a function of the level of noise σ. The performance measures are the percentage of misclassified correspondences and the error in the estimation of the translation direction in degrees, averaged over 1,000 trials. Two-view algorithms are applied to views 1-2 and 1-3, and the errors are averaged. We notice the following:

1. The performance of all five algorithms deteriorates with the amount of noise and with the amount of rotation.
2. The algebraic algorithms 2-A and 3-A have a comparable performance in terms of segmentation error, but 3-A consistently gives better estimates of the translation. Notice also that as the amount of rotation increases, the relative performance of 3-A versus 2-A improves.
3. Using the Sampson error (2-A+S) or the reprojection error (3-A+R) instead of the algebraic distance from epipoles to epipolar lines (2-A and 3-A) improves the performance of the algebraic algorithms both in terms of motion segmentation and motion estimation errors.
4. Algorithm 3-A+R outperforms 2-A+S in terms of segmentation error. As per the error in translation, 2-A+S outperforms 3-A+R for zero rotation, but

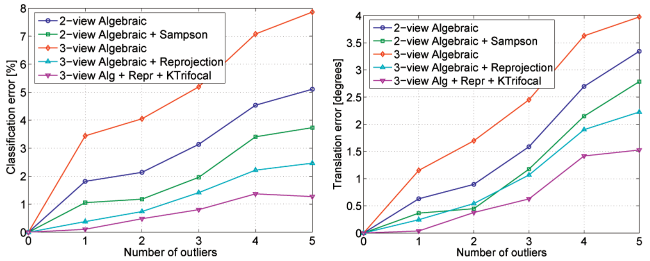


Fig. 9. Motion segmentation and motion estimation errors as a function of the number of outliers for $\tau = 100$ u.f.l. and $\theta = 0$ degree.

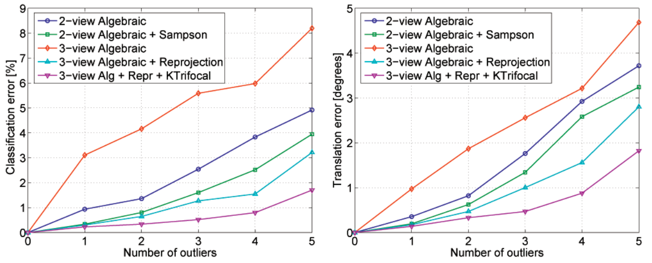


Fig. 10. Motion segmentation and motion estimation errors as a function of the number of outliers for $\tau = 100$ u.f.l. and $\theta = 5$ degrees.

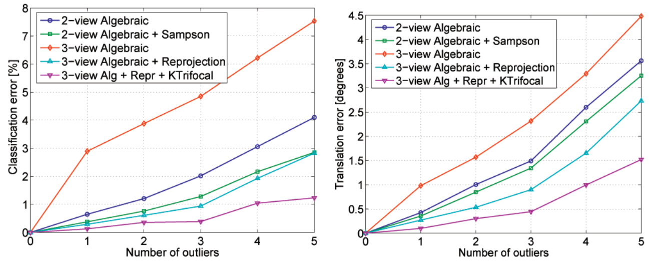
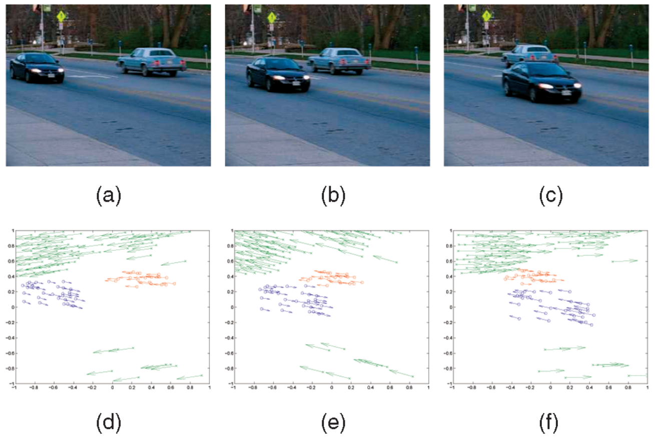


Fig. 11. Motion segmentation and motion estimation errors as a function of the number of outliers for $\tau = 100$ u.f.l. and $\theta = 10$ degrees.

3-A+R outperforms 2-A+S as the amount of rotation increases.

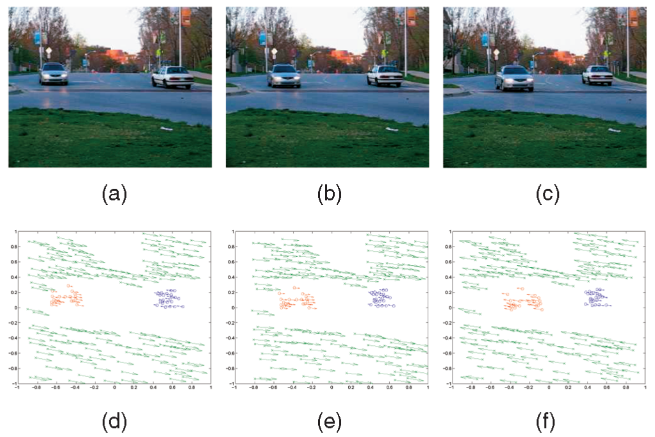
5. Iterative refinement using the K-trifocal algorithm gives the best performance, with a misclassification ratio of less than 1.5 percent and a translation error of less than 1.5 degrees.
6. The performance of all algorithms deteriorates as the baseline τ reduces. However, three-view algorithms are in general less sensitive to the reduction of the baseline.
7. It is reported in [13] that if one randomly chooses two lines passing through each image point in Algorithms 1 and 2, then algorithms 3-A, 3-A+R, and 3-A+R+T give misclassification errors of 0-20 percent, 0-9 percent, and 0-3 percent, respectively, and translation errors of 0-22 degrees, 0-12 degrees, and 0-3.7 degrees, respectively. Therefore, choosing two canonical lines passing through each image point in Algorithms 1 and 2, as we have done in this paper, gives significantly better results than choosing the lines at random.

Figs. 9, 10, and 11 show the performance of perspective motion segmentation algorithms as a function of the number of outliers. For each one of the three views, outliers are drawn uniformly on the image domain. As expected, the most robust algorithm is 3-A+R+T, followed by 3-A+R and 2-A+S, because they use a more robust error for clustering the correspondences. Notice, however, that 2-A is more robust than 3-A, even though 3-A uses more views. At a high level, the main difference between algorithms 2-A and 3-A is the computation of epipolar lines (Step 3 of Algorithm 5): 2-A computes the epipolar lines directly from the derivatives of the multibody epipolar constraint [32], whereas 3-A needs to solve for the common root of a set of univariate polynomials. The robustness of the 3-A algorithm would likely benefit by using a more robust algorithm for computing the common root of a set of polynomials.


(a) (b) (c)

(d) (e) (f)

Fig. 12. Top: frames 1, 6, and 12 of the cars2-06 sequence. Bottom: 2D displacements of the 123 correspondences from the current view (“o”) to the next (“→ ”). (a) Frame 1. (b) Frame 6. (c) Frame 12. (d) Two-dimensional displacements from frames 1 to 6. (e) Two-dimensional displacements from frames 6 to 12. (f) Two-dimensional displacements from frames 12 to 1.


(a) (b) (c)

(d) (e) (f)

Fig. 13. Top: frames 5, 9, and 13 of the cars2-07 sequence. Bottom: 2D displacements of the 212 correspondences from the current view (“o”) to the next (“→ ”). (a) Frame 5. (b) Frame 9. (c) Frame 13. (d) Two-dimensional displacements from frames 5 to 9. (e) Two-dimensional displacements from frames 9 to 13. (f) Two-dimensional displacements from frames 13 to 5.

4.2 Experiments on Real Sequences

We now test the performance of all algorithms on several frame triples from two traffic sequences: *cars2-06* (Fig. 12) and *cars2-07* (Fig. 13). These sequences contain two cars translating and rotating (groups 1 and 2) and are taken with a moving hand held camera (group 3). Point correspondences are

TABLE 2
Misclassification Rates for Sequence *Cars2-06-g12*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-3-6	0.2778	0.2917	0.2708	0.0208	0.0208	0.0000
1-6-12	0.1528	0.2222	0.2917	0.0208	0.0208	0.0000
2-3-5	0.2083	0.2847	0.1667	0.0625	0.0625	0.0000
2-4-5	0.2639	0.3333	0.1667	0.1667	0.1458	0.0000
2-5-8	0.2778	0.2569	0.3125	0.0208	0.0208	0.0000
3-6-11	0.3403	0.3403	0.0833	0.0417	0.0417	0.0000
3-10-11	0.2500	0.2292	0.4167	0.3542	0.1250	0.0000
5-8-12	0.3333	0.3611	0.1667	0.0208	0.0208	0.0000
6-7-10	0.2292	0.3403	0.2500	0.1250	0.0625	0.0000
8-9-10	0.4167	0.4583	0.0833	0.0208	0.0208	0.0000

TABLE 3
Misclassification Rates for Sequence *Cars2-06-g23*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-9-14	0.3814	0.2788	0.2981	0.0673	0.0673	0.0481
2-11-12	0.3462	0.3173	0.2596	0.1538	0.0769	0.0096
5-13-15	0.2051	0.1795	0.2788	0.2788	0.0288	0.0385
11-13-15	0.3173	0.2821	0.2404	0.3558	0.0385	0.0673

TABLE 4
Misclassification Rates for Sequence *Cars2-06-g13*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-3-15	0.2943	0.3440	0.3298	0.0426	0.0426	0.1170
1-7-14	0.3156	0.2943	0.3298	0.0106	0.0106	0.0000
2-3-6	0.2057	0.2624	0.2766	0.0745	0.0745	0.0106
2-5-7	0.2447	0.3546	0.1064	0.0106	0.0106	0.0532
7-9-12	0.2801	0.2801	0.3191	0.0106	0.0106	0.0106

TABLE 5
Misclassification Rates for Sequence *Cars2-06-g123*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-7-11	0.5339	0.5176	0.4228	0.2358	0.0325	0.0000
2-12-15	0.4688	0.4851	0.5935	0.3171	0.0650	0.1951
2-3-13	0.4851	0.3740	0.5610	0.3659	0.1870	0.0000
3-13-14	0.4715	0.4092	0.5447	0.2114	0.0488	0.0163
4-9-13	0.4309	0.3930	0.4309	0.1220	0.0244	0.0732
5-6-8	0.4444	0.3740	0.2520	0.0813	0.0813	0.1870
6-9-12	0.4201	0.3875	0.3821	0.2114	0.0000	0.0488
7-9-12	0.4228	0.4092	0.5366	0.1220	0.0000	0.2358
8-10-15	0.4580	0.4661	0.4228	0.3171	0.0569	0.2114
9-12-15	0.4526	0.4878	0.4959	0.3984	0.0488	0.1951

TABLE 6
Misclassification Rates for Sequence *Cars2-07-g12*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-5-20	0.3496	0.3496	0.1707	0.1707	0.0732	0.0732
1-6-19	0.3415	0.3415	0.1951	0.1463	0.0976	0.0000
1-10-12	0.2195	0.2846	0.2439	0.0244	0.0244	0.1463
2-7-15	0.2927	0.3496	0.1951	0.0244	0.0244	0.0000
3-8-17	0.2358	0.2195	0.2927	0.0732	0.0244	0.0244
3-10-14	0.0325	0.0650	0.1220	0.0488	0.0488	0.0488
7-10-20	0.1463	0.1463	0.2439	0.0976	0.0244	0.0488

TABLE 7
Misclassification Rates for Sequence *Cars2-07-g23*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-11-15	0.3886	0.3264	0.0881	0.0777	0.0777	0.0363
2-9-12	0.4197	0.3541	0.0881	0.0052	0.0052	0.0725
2-10-14	0.4007	0.3834	0.0933	0.0052	0.0052	0.1295
2-11-15	0.4473	0.3541	0.0725	0.0052	0.0052	0.0415
5-9-13	0.4162	0.4093	0.0881	0.0155	0.0155	0.0933
10-17-18	0.3454	0.3143	0.0725	0.1088	0.1088	0.0259
12-15-19	0.4542	0.4646	0.1865	0.0466	0.0466	0.0415

TABLE 8
Misclassification Rates for Sequence *Cars2-07-g13*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
1-2-9	0.3772	0.3140	0.3000	0.0947	0.0789	0.1526
2-3-19	0.3737	0.3491	0.1737	0.1105	0.0421	0.0053
3-6-8	0.4070	0.3053	0.3684	0.0842	0.0842	0.0684
4-9-10	0.3895	0.4246	0.2579	0.1368	0.1368	0.1895
5-6-19	0.4018	0.3982	0.2789	0.1000	0.0947	0.0053
6-10-20	0.4158	0.4368	0.2316	0.1053	0.1000	0.0000
7-9-17	0.4140	0.3825	0.1421	0.0737	0.0526	0.0158
8-12-20	0.4526	0.4123	0.1579	0.1000	0.0632	0.0000

extracted automatically using OpenCV, which is available at http://sourceforge.net/projects/opencvlibrary/. For the purposes of ground truth comparison, the point trajectories are manually segmented according to the three independent motions in the scene. We use point correspondences from groups 1-2, 1-3, and 2-3 for two-body motion segmentation and 1-2-3 for three-body motion segmentation.

4.2.1 Results on the Cars-06 Sequence

Tables 2, 3, and 4 show segmentation results for point correspondences associated with groups 1-2, 2-3, and 1-3, respectively, of the *cars2-06* sequence. With few exceptions, 2-A and 2-A+S give an error of about 20-40 percent, 3-A gives an error of about 15-30 percent, 3-A+R gives an error of about 0-15 percent, 3-A+R+T gives an error of about 0-10 percent, and 3-Aff gives an error of about 0-5 percent. However, there are triplets of frames for which the misclassification error of perspective algorithms is high (frames 2-4-5 in Table 2), 2-A and 2-A+S perform better than 3-A and 3-A+R (frames 3-10-11 in Table 2), or 3-A+R initialized with 3-A works worse than 3-A (frames 11-13-15 in Table 3). Table 5 show segmentation results for point correspondences associated with groups 1-2-3. In general, algorithms 2-A, 2-A+S, 3-A, and 3-A+R have high errors, whereas 3-A+R+T and 3-Aff give an error of about 0-19 percent.

Among the perspective algorithms, the one with the best performance on this sequence is 3-A+R+T. However, the affine algorithm 3-Aff performs well in most cases, particularly for groups 1-2 where it gives perfect segmentation. This is because groups 1-2 contain degenerate motions (two cars moving on a straight line on the same plane) for which an affine model may be more appropriate than a fundamental matrix or a trifocal tensor. In fact, when group 3 (camera) is present, the scene has more perspective effects, and 3-A+R+T and 3-Aff perform similarly.

4.2.2 Results on the Cars-07 Sequence

Tables 6, 7, and 8 show segmentation results for point correspondences associated with groups 1-2, 2-3, and 1-3, respectively, of the *cars2-07* sequence. In general, the performance of the two-view algorithms is about the same as for the *cars2-06* sequence (20-40 percent), whereas the performance of the three-view algorithms on this sequence is a bit better, particularly for groups 2-3. However, there still are triplets of frames for which the best misclassification error is high (frames 4-9-10 in Table 8), 2-A and 2-A+S perform better than 3-A (frames 7-10-20 in Table 6), or 3-A+R initialized with 3-A works worse than 3-A (frames 10-17-18 in Table 7).

TABLE 9
Misclassification Rates for Sequence *Cars2-07-g123*

Frames	2-A	2-A+S	3-A	3-A+R	3-A+R+T	3-Aff
8-12-19	0.5016	0.5283	0.4340	0.1981	0.0094	0.0519
8-15-20	0.4921	0.4450	0.5330	0.0189	0.0755	0.0896
9-16-20	0.5692	0.4308	0.5047	0.1085	0.0566	0.1085
10-14-19	0.5566	0.3758	0.4151	0.1604	0.0425	0.0896

Table 9 show segmentation results for point correspondences associated with groups 1-2-3. In general, algorithms 2-A, 2-A+S, and 3-A have high errors (40-50 percent), 3-A+R gives an error of about 10-20 percent, 3-A+R+T gives an error of about 0-8 percent and 3-Aff gives an error of about 5-10 percent.

5 Discussion

The fact that the relative performance of the algorithms is not always as expected should come at no surprise, because the quality of the segmentation of a given scene depends on whether the model (mixture of trifocal tensors) is appropriate to describe the 3D motion of the scene. Often, 3D scenes contain planar structures, two objects whose motion is similar in a few frames, objects moving approximately on a straight line or in the same plane, and so forth. All these degenerate situations affect the quality of the motion estimates and, thus, the quality of the segmentation.

The overall findings of our experiments were that the multibody trifocal tensor methods described in this paper generally outperform the comparable two-view methods based on the multibody fundamental matrix [34], [32]. Although this was true most of the time, there were occasional instances in which the fundamental matrix algorithm worked better. The advantage of using the trifocal-tensor method is that critical configurations (often fatal to success) are less likely to occur.

The iterative methods to refine the results were generally essential for acceptable results. They were not always successful, however, and would sometimes even lead to a deterioration of the results. The lesson to be learned from this is that the success in motion segmentation is very dependent on the particulars of the dynamics of the scene. It occurs not infrequently that the dynamics of the scene or the camera motion is so constrained that it is not possible to distinguish separate motions based only on multiple-motion epipolar geometry, as investigated in this paper. In such cases, results tend to be more aleatoric.

The present algorithm is mainly recommended for cases in which only three views are available or affine geometry is not a good approximation for the geometry of the scene. The advantage of using many views such as a complete video sequence is a much greater ability to distinguish different motions. We have found that the affine-motion multiple-view algorithm described in [27] gives more reliable results in cases where it may be applied. We expect (and demonstrate with our synthetic results) that the present algorithm will work more reliably when there are strong perspective effects (which make the affine approximation nonviable).

For this reason, the trifocal algorithm performs better on synthetic data. However, for real-world scenes, the advantage seems to lie with the affine multiple-view algorithm in [27]. In most real scenes, the affine approximation is reasonable, and we often have many more than three views.

6 Conclusions and Future Work

The multibody trifocal tensor is effective in the analysis of dynamic scenes involving several moving objects. The algebraic method of motion classification involves computation of the multibody tensor, computation of the epipoles for different motions, and classification of the points according to the compatibility of epipolar lines with the different epipoles. Our reported implementation of this algorithm was sufficiently good to provide an initial classification of points into different motion classes. This classification can be refined using an iterative algorithm with excellent results. It is likely that more careful methods of computing the tensor (analogous with the best methods for the single-body trifocal tensor) could give a better initialization, as could a classification method that proceeded by algebraically extracting the single-body tensors from the multibody trifocal tensor. These methods have not yet been tried. The algebraic properties of the multibody trifocal tensor are in many respects analogous to those of the single-body tensor but provide many surprises and avenues of research that we have not yet exhausted.

Acknowledgments

The authors thank Matthias Behnisch and Roberto Tron for their help with the experiments. This work has been funded with start-up funds from the Whiting School of Engineering of the Johns Hopkins University, by the US National Science Foundation Faculty Early Career Development (CAREER) Grants IIS-0447739 and US Office of Naval Research N00014-05-10836, and by National ICT Australia. National ICT Australia is funded by the Australian Government's Department of Communications, Information Technology, and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Research Centre of Excellence programs.

References

[1] T.E. Boult and L.G. Brown, "Factorization-Based Segmentation of Motions," *Proc. IEEE Workshop Motion Understanding,* pp. 179-186, 1991.

[2] J. Costeira and T. Kanade, "A Multibody Factorization Method for Independently Moving Objects," *Int'l J. Computer Vision,* vol. 29, no. 3, pp. 159-179, 1998.

[3] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc. B,* vol. 39, pp. 1-38, 1977.

[4] R. Duda, P. Hart, and D. Stork, *Pattern Classification,* second ed. John Wiley & Sons, 2000.

[5] Z. Fan, J. Zhou, and Y. Wu, "Multibody Grouping by Inference of Multiple Subspaces from High-Dimensional Data Using Oriented-Frames," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 1, pp. 91-105, Jan. 2006.

[6] X. Feng and P. Perona, "Scene Segmentation from 3D Motion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 225-231, 1998.

[7] M.A. Fischler and R.C. Bolles, "RANSAC Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM,* vol. 26, pp. 381-395, 1981.

[8] A. Fitzgibbon and A. Zisserman, "Multibody Structure and Motion: 3D Reconstruction of Independently Moving Objects," *Proc. Sixth European Conf. Computer Vision*, pp. 891-906, 2000.

[9] C.W. Gear, "Multibody Grouping from Motion Images," *Int'l J. Computer Vision*, vol. 29, no. 2, pp. 133-150, 1998.

[10] A. Gruber and Y. Weiss, "Multibody Factorization with Uncertainty and Missing Data Using the EM Algorithm," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 707-714, 2004.

[11] M. Han and T. Kanade, "Reconstruction of a Scene with Multiple Linearly Moving Objects," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 542-549, 2000.

[12] M. Han and T. Kanade, "Multiple Motion Scene Reconstruction from Uncalibrated Views," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 1, pp. 163-170, 2001.

[13] R. Hartley and R. Vidal, "The Multibody Trifocal Tensor: Motion Segmentation from 3 Perspective Views," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 769-775, 2004.

[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, second ed. Cambridge Univ. Press, 2004.

[15] K. Kanatani, "Motion Segmentation by Subspace Separation and Model Selection," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 586-591, 2001.

[16] K. Kanatani, "Evaluation and Selection of Models for Motion Segmentation," *Proc. Fifth Asian Conf. Computer Vision*, pp. 7-12, 2002.

[17] K. Kanatani and C. Matsunaga, "Estimating the Number of Independent Motions for Multibody Motion Segmentation," *Proc. Seventh European Conf. Computer Vision*, pp. 25-31, 2002.

[18] K. Kanatani and Y. Sugaya, "Multi-Stage Optimization for Multi-Body Motion Segmentation," *Proc. Australia-Japan Advanced Workshop Computer Vision*, pp. 335-349, 2003.

[19] Y. Ma, K. Huang, R. Vidal, J. Košecká, and S. Sastry, "Rank Conditions on the Multiple View Matrix," *Int'l J. Computer Vision*, vol. 59, no. 2, pp. 115-137, 2004.

[20] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3D Vision: From Images to Geometric Models.* Springer, 2003.

[21] A. Shashua and A. Levin, "Multi-Frame Infinitesimal Motion Model for the Reconstruction of (Dynamic) Scenes with Multiple Linearly Moving Objects," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 592-599, 2001.

[22] A. Shashua and L. Wolf, "Homography Tensors: On Algebraic Entities that Represent Three Views of Static or Moving Planar Points," *Proc. Sixth European Conf. Computer Vision*, vol. 1, pp. 507-521, 2000.

[23] F. Shi, J. Wang, J. Zhang, and Y. Liu, "Motion Segmentation of Multiple Translating Objects Using Line Correspondences," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 315-320, 2005.

[24] P. Sturm, "Structure and Motion for Dynamic Scenes—The Case of Points Moving in Planes," *Proc. Seventh European Conf. Computer Vision*, pp. 867-882, 2002.

[25] P. Torr, R. Szeliski, and P. Anandan, "An Integrated Bayesian Approach to Layer Extraction from Image Sequences," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 297-303, Mar. 2001.

[26] P.H.S. Torr, "Geometric Motion Segmentation and Model Selection," *Philosophical Trans. Royal Soc. of London*, vol. 356, no. 1740, pp. 1321-1340, 1998.

[27] R. Vidal and R. Hartley, "Motion Segmentation with Missing Data by PowerFactorization and Generalized PCA," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 310-316, 2004.

[28] R. Vidal and Y. Ma, "A Unified Algebraic Approach to 2-D and 3-D Motion Segmentation," *Proc. Eighth European Conf. Computer Vision*, pp. 1-15, 2004.

[29] R. Vidal, Y. Ma, and J. Piazzi, "A New GPCA Algorithm for Clustering Subspaces by Fitting, Differentiating and Dividing Polynomials," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 510-517, 2004.

[30] R. Vidal, Y. Ma, and S. Sastry, "Generalized Principal Component Analysis (GPCA)," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 621-628, 2003.

[31] R. Vidal, Y. Ma, and S. Sastry, "Generalized Principal Component Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1-15, Dec. 2005.

[32] R. Vidal, Y. Ma, S. Soatto, and S. Sastry, "Two-View Multibody Structure from Motion," *Int'l J. Computer Vision*, vol. 68, no. 1, pp. 7-25, 2006.

[33] L. Wolf and A. Shashua, "Affine 3-D Reconstruction from Two Projective Images of Independently Translating Planes," *Proc. Eighth IEEE Int'l Conf. Computer Vision*, pp. 238-244, 2001.

[34] L. Wolf and A. Shashua, "Two-Body Segmentation from Two Perspective Views," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 263-270, 2001.

[35] Y. Wu, Z. Zhang, T.S. Huang, and J.Y. Lin, "Multibody Grouping via Orthogonal Subspace Decomposition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 252-257, 2001.

[36] L. Zelnik-Manor and M. Irani, "Degeneracies, Dependencies and Their Implications in Multi-Body and Multi-Sequence Factorization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 287-293, 2003.

René Vidal received the BS degree (highest honors) in electrical engineering from the Universidad Católica de Chile in 1997 and the MS and PhD degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 2000 and 2003, respectively. In 2004, he joined the Johns Hopkins University as an assistant professor in the Department of Biomedical Engineering and the Center for Imaging Science. He coedited (with Anders Heyden and Yi Ma) the book *Dynamical Vision* and authored more than 80 articles in biomedical imaging, computer vision, machine learning, hybrid systems, robotics, and vision-based control. He is the recipient of the 2005 US National Science Foundation (NFS) Faculty Early Career Development (CAREER) Award, the 2004 Best Paper Award Honorable Mention at the European Conference on Computer Vision, the 2004 Sakrison Memorial Prize, the 2003 Eli Jury Award, and the 1997 Award of the School of Engineering of the Universidad Católica de Chile to the best graduating student of the school. He is a member of the IEEE.

Richard Hartley received the BSc degree from the Australian National University (ANU) in 1971, the MSc degree in computer science from Stanford University in 1972, and the PhD degree in mathematics from the University of Toronto, Canada, in 1976. He is currently a professor and member of the computer vision group in the Department of Information Engineering at ANU. He also belongs to the Vision Science Technology and Applications Program in National ICT Australia, a government-funded research institute. He did his PhD thesis in knot theory and worked in this area for several years before joining the General Electric (GE) Research and Development Center, where he worked from 1985 to 2001. During the period 1985-1988, he was involved in the design and implementation of computer-aided design tools for electronic design and created a very successful design system called the Parsifal Silicon Compiler, described in his book *Digit Serial Computation*. In 1991, he was awarded GE's Dushman Award for this work. Around 1990, he developed an interest in computer vision, and in 2000, he coauthored (with Andrew Zisserman) a book on multiple-view geometry. He has authored more than 100 papers in knot theory, geometric voting theory, computational geometry, computer-aided design, and computer vision and holds 32 US patents. He is a member of the IEEE.