COMMUNICATIONS IN INFORMATION AND SYSTEMS
Vol. 1, No. 1, pp. 51-74, January 2001

004

OPTIMAL MOTION ESTIMATION FROM MULTIPLE IMAGES BY NORMALIZED EPIPOLAR CONSTRAINT*

Y. MA†, R. VIDAL†, S. HSU†, AND S. SASTRY†

Abstract. In this paper, we study the structure from motion problem as a constrained nonlinear least squares problem which minimizes the so called reprojection error subject to all constraints among multiple images. By converting this constrained optimization problem to an unconstrained one, we contend that multilinear constraints, when used for motion and structure estimation, need to be properly normalized, which makes them no longer tensors. We demonstrate this by using the bilinear epipolar constraints and show how they give rise to a multiview version of the (crossed) normalized epipolar constraint of two views [6]. Such a (crossed) normalized epipolar constraint serves as an optimal objective function for motion (and structure) estimation. This objective function further reveals certain statistical relationship between bilinear and trilinear constraints: Even rectilinear motion can be correctly estimated by the normalized epipolar constraint as a limit of generic cases, hence trilinear constraints are not really necessary. Since the so obtained objective function is defined naturally on a product of Stiefel manifolds, we show how to use geometric optimization techniques [2] to minimize such a function. Simulation and experimental results are presented to evaluate the proposed algorithm and verify our claims.

1. Introduction.

Over the past years, computer vision has been widely used in robotics and control applications for many purposes: autonomous navigation, obstacle avoidance, object recognition and manipulation, 3D map building, telepresence, etc. In all these areas, an important question is how to recover geometric and dynamic information from the scene being observed.

In this paper, we revisit a classic problem in computer vision: *Given a camera undergoing an unknown rigid body motion and observing a cloud of points with unknown 3D positions, recover* **camera motion** *and (Euclidean)* **scene structure** *(3D position of the points) from their* **correspondences** *in multiple images (position of each point projected in each one of the images)*

With such a vast body of literature studying almost every aspect of this problem (see, for example, reviews of batch methods [15], recursive methods [8, 14], orthographic case [16] and projective reconstruction [19]), it is quite reasonable to ask what, if anything, can still be new in this topic.

First of all, we do not yet have a clear picture about the relationship between multilinear constraints and the (statistical) optimality of motion and structure estimates. Although we have understood very well the geometric (or algebraic) relationship among multilinear constraints [5, 7, 12, 18] (which will be briefly reviewed in Section 3), when it comes to using them for designing motion or structure recovery algorithms, they are usually used as *objectives*, rather than *constraints*. Many researchers believe that multilinear tensors should be recovered first and, from them, motion and structure could be further retrieved [4]. Algebraically, this is true. Nevertheless, when a noise model is considered and the direct objective is to minimize certain statistics, such as the *reprojection error* (also called *nonlinear least squares*

*Invited paper; received May 26, 2000; accepted for publication June 13, 2000.
† Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720-1774, E-mail: {mayi, rvidal, shawn, sastry}@eecs.berkeley.edu

error [15]), it becomes quite unclear how to incorporate these multilinear constraints into the objective. More specifically, we want to answer the questions:

> **(i)** *Can we convert such a constrained estimation (or optimization) problem to an unconstrained one? If so, what weight should be assigned to each constraint?*

Secondly, we have every reason to believe that, for such a constrained estimation problem, its *a posteriori* likelihood function (or some variation of it) still needs to be found. From an estimation theoretic viewpoint, such a function should indeed capture some peculiar statistical nature of the multiview structure from motion problem. Other than the well known algebraic and geometric relationship between bilinear and trilinear constraints, we may ask:

> **(ii)** *What is the* **statistical relationship** *between bilinear and trilinear constraints? Are trilinear constraints really needed for motion (or structure) estimation in the degenerate rectilinear motion case?*

On the other hand, from an optimization theoretic viewpoint, with such a function we can further understand:

> **(iii)** *What is the exact nature of the* **optimization** *associated with the original problem? What geometric space does the optimization take place on? Is there any generic optimization technique available for minimizing such a function?*

Finally, in applications which require high accuracy, noise sensitivity becomes the primary concern [1, 6, 20]. Although a specific sensitivity study is needed for every algorithm, it is still possible to study the *intrinsic sensitivity* inherent in the initial problem. From statistics, we know that the Hessian of the *a posteriori* likelihood function, evaluated at the maximum, closely approximates the covariance matrix of the estimates. Hence, an *explicit* expression for the likelihood function is absolutely necessary for a systematic study of the intrinsic sensitivity issue. As we will soon see, the normalized epipolar constraint to be derived is such a function and we will show how to compute its Hessian, even though the sensitivity issue is not a main subject of this paper (see Section 5).

In this paper, we will give clear answers to the above questions through the development of a solution to the constrained nonlinear least squares optimization problem which minimizes the reprojection error subject to all constraints among multiple images. Question set **(i)** will be answered in Section 4. The answers will become evident from the derivation and the form of the normalized epipolar constraint for multiple images. For Question set **(ii)**, the statistical relationship between bilinear and trilinear constraints will be revealed by Simulation 3 in Section 6 and some further explanation will be given in Comment 5. Question set **(iii)** will be answered in Section 5 where a generic optimization algorithm is explicitly laid out for minimizing the normalized epipolar constraint for multiple images. Although our results, including the algorithm, can be easily generalized to trilinear constraints or even to an uncalibrated framework, we choose to present the calibrated case using bilinear (epipolar) constraints so as to clearly convey the main ideas.

Relations to Previous Work: Our algorithm belongs to the so called *batch methods* for motion and structure recovery from multiple views, like those in [15, 16, 19], and is a necessary extension of the unconstrained nonlinear least squares method in [15]. We here emphasize again that our focus is *not* on an algorithm for computing motion or structure faster than the ones in [10, 20], although we will mention briefly how to speed up our algorithm. Instead, we are using our algorithm as a means of revealing the interesting *geometry* in multiview structure from motion. In doing so, one will be able to see what roles multilinear constraints play in the design of optimal algorithms. In addition, the revelation of the statistical relationship between bilinear and trilinear constraints is an important complement to the well known algebraic or geometric results [5, 7, 12, 18]. Our results, especially the normalized epipolar constraint, may also help improve existing *recursive methods* such as those in [8, 14] if the filter objective function is modified to the one given by us. Moreover, studying the Hessian of such an objective will allow to extend existing sensitivity studies [1, 6] to the multiview case.

2. Notation and Problem Statement. We first introduce some notation which will be frequently used in this paper (the notation is consistent with that in [9]). Given a vector $p = [p_1, p_2, p_3]^T \in \mathbb{R}^3$, we define $\widehat{p} \in so(3)$ (the space of skew symmetric matrices in $\mathbb{R}^{3\times 3}$) by:

$$\widehat{p} = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{bmatrix}. \tag{2.1}$$

It then follows from the definition of cross-product of vectors that, for any two vectors $p, q \in \mathbb{R}^3$ we have $p \times q = \widehat{p}q$. Also, we represent a point $q = [q_1, q_2, q_3]^T \in \mathbb{R}^3$ in **homogeneous coordinates** as $\underline{q} = [q_1, q_2, q_3, 1]^T \in \mathbb{R}^4$. The set of all such points can also be identified as the subset of $\mathbb{RP}^3$ excluding the plane at infinity, *i.e.,* the plane consisting of all points with coordinates $[q_1, q_2, q_3, 0]^T$.

The camera motion is modeled as a rigid body motion in $\mathbb{R}^3$. The displacement of the camera belongs to the special Euclidean group $SE(3)$, represented in homogeneous coordinates as:

$$SE(3) = \left\{ g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \Big| \, p \in \mathbb{R}^3, R \in SO(3) \right\} \tag{2.2}$$

where $SO(3)$ is the space of 3×3 rotation matrices (orthogonal matrices with determinant $+1$). Clearly, a transformation g is uniquely determined by its rotational part $R \in SO(3)$ and translational part $p \in \mathbb{R}^3$, therefore, we can express $g \in SE(3)$ by (R, p) as a shorthand. Let $q(t), t \in \mathbb{R}$ be the coordinates of q with respect to the camera coordinate frame at time t. Then the coordinate transformation between $q(t)$ and $q(t_0)$ is given by:

$$\underline{q}(t) = g(t)\underline{q}(t_0). \tag{2.3}$$

Without loss of generality, we may assume that $q(t_0)$ are the coordinates of q with respect to a pre-defined inertial frame. In $\mathbb{R}^3$, the above coordinate transformation is equivalent to:

$$q(t) = R(t)q(t_0) + p(t). \tag{2.4}$$

Define the **projection matrix** $P \in \mathbb{R}^{3\times 4}$ to be $P = [I_{3\times 3}, 0]$ and the imaging surface $N \subset \mathbb{R}^3$. In this paper we always use bold letters to denote image points. Then, the image $\mathbf{x} = (x, y, z)^T \in N$ of a point $q \in \mathbb{R}^3$ is in general assumed to satisfy the following equation:

$$\lambda \mathbf{x} = P\underline{q}. \tag{2.5}$$

where $\lambda > 0$ encodes the (unknown positive) depth information, defined to be the **scale** of the point q with respect to its image $\mathbf{x}$. For instance, $\lambda = q_3$ for perspective projection and $\lambda = \|q\|$ for spherical projection. If the imaging surface has variable curvature, λ can be more involved. Combining (2.3) and (2.5), we have the imaging model for a moving camera:

$$\lambda(t)\mathbf{x}(t) = Pg(t)\underline{q} \tag{2.6}$$

where $g(t)$ is the (unknown) coordinate transformation between the camera frames at times t and t_0.

In the above equation, the image point $\mathbf{x}$ is expressed in coordinates with the same metric as a pre-chosen inertial coordinate frame. In practice, however, the physical location of the image point $\mathbf{x}$ is usually expressed in the so called image coordinates $\mathbf{x}_{im}$ which depend on many physical parameters of the camera such as focal length, skew, etc. In general, we can think of $\mathbf{x}_{im}$ as $\mathbf{x}$ distorted by some linear map: $\mathbf{x}_{im} = K\mathbf{x}$. The matrix $K \in \mathbb{R}^{3\times 3}$ is of the general form:

$$K = \begin{bmatrix} k_x & s & t_x \\ 0 & k_y & t_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.7}$$

Determining the matrix K is the so called camera calibration problem. It is not the goal of this paper to study how to calibrate a camera. Hence, throughout this paper, we will assume that our camera is pre-calibrated, *i.e.,* we know K in advance. In Section 7.1, we will briefly review how to determine K using one of the many methods in the computer vision literature. Once K is known, we can always normalize an image point $\mathbf{x}_{im}$ by pre-multiplying it with K^{-1} to obtain the coordinates $\mathbf{x} = K^{-1}\mathbf{x}_{im}$ with respect to the pre-chosen metric.

Finally, we need to obtain the coordinates $\mathbf{x}_{im}$ of each image point and establish its correspondence among multiple images. That is, we need to find image points from a sequence of images that correspond to the same *physical* point in 3D. These problems are called **feature tracking** and **correspondence** problems, respectively. We will briefly review them in Section 7.1.

Problem Statement: Given a set of corresponding image points $\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_m^i \in N$ of a 3D point $q^i, i = 1, \ldots, n$, with respect to m camera frames (at m unknown locations or time instances), recover the relative motions (transformations) among the m camera frames and then the 3D locations of the n points with respect to the m camera frames.

To be consistent with the notation, we always use the superscript to enumerate the n different points. The subscript is always used to enumerate the m different camera frames. According to the problem statement, in Eq. (2.6), except for the fact that $\mathbf{x}$ is measured and P is a constant matrix, everything else, *i.e.,* λ, q and g, are all unknown and are entitled to be recovered from the measured $\mathbf{x}$. As we will soon see, due to some constraints that multiple images of a 3D point must satisfy, the problem of recovering the camera motion g and that of recovering the 3D location of the point q can be very much decoupled. Furthermore, once the camera motion is known, determining the 3D locations of all the feature points is a much simpler problem. Hence, in this paper, we will focus on the problem of recovering camera motion. In Experiment 1 Section 7.2, we will show that once the motion is well estimated, a good reconstruction of 3D structure can also be obtained.

3. Geometric Interpretation of Multilinear Constraints. Denote the relative motion (transformation) between the k^{th} and j^{th} frames as $g_{kj} = (R_{kj}, p_{kj}) \in SE(3), 1 \leq j,k \leq m$. For $i = 1, \ldots, n$, let λ_j^i be the scale of the point q^i with respect to its j^{th} image $\mathbf{x}_j^i$. Then from (2.3) and (2.5) we have:

$$\begin{bmatrix} \mathbf{x}_1^i & 0 & \cdots & 0 \\ 0 & \mathbf{x}_2^i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{x}_m^i \end{bmatrix} \begin{bmatrix} \lambda_1^i \\ \lambda_2^i \\ \vdots \\ \lambda_m^i \end{bmatrix} = \begin{bmatrix} Pg_{11} \\ Pg_{21} \\ \vdots \\ Pg_{m1} \end{bmatrix} \underline{q}^i \tag{3.1}$$

which we rewrite in a more compact notation as:

$$\mathbf{X}^i \vec{\lambda}^i = A\underline{q}^i. \tag{3.2}$$

We call $A \in \mathbb{R}^{3m \times 4}$ the **motion matrix**. Notice that the motion matrix $A = [a_1, a_2, a_3, a_4]$ has four column vectors $a_l \in \mathbb{R}^{3m}$, $1 \leq l \leq 4$. A only depends on the relative motions between camera frames and can be viewed as a natural generalization of the two view case [6].

Now for the j^{th} image $\mathbf{x}_j \in \mathbb{R}^3$ of a point q, we define the vector $\vec{\mathbf{x}}_j \in \mathbb{R}^{3m}$ associated with $\mathbf{x}_j$ to be the j^{th} column of the matrix $\mathbf{X}$:

$$\vec{\mathbf{x}}_j = [0, \ldots, 0, \mathbf{x}_j^T, 0, \ldots, 0]^T \in \mathbb{R}^{3m}, \quad 1 \leq j \leq m. \tag{3.3}$$

We then have the well-known results:

PROPOSITION 3.1 (Multilinear Constraint). *Given m images $\{\mathbf{x}_j \in \mathbb{R}^3\}_{j=1}^m$ of a point q, and the motion matrix $A = [a_1, a_2, a_3, a_4] \in \mathbb{R}^{3m \times 4}$ of relative motions between camera frames, the associated vectors $\{\vec{\mathbf{x}}_j \in \mathbb{R}^{3m}\}_{j=1}^m$ satisfy the following wedge product equation:*

$$a_1 \wedge a_2 \wedge a_3 \wedge a_4 \wedge \vec{\mathbf{x}}_1 \wedge \ldots \wedge \vec{\mathbf{x}}_m = 0. \tag{3.4}$$

For given camera motions, this equation gives multilinear constraints in the m images $\mathbf{x}_j$ of a single 3D point. Among all the constraints given by this wedge product equation, those involving only four images are called **quadrilinear**, those involving only three images are called **trilinear**, and those involving only two images are called either **bilinear**, **fundamental** or **epipolar**. It has been shown that constraints involving more than four images are (algebraically) dependent on the trilinear and bilinear ones [5].

For the problem of motion and structure reconstruction, we are more interested in recovering the motion matrix A from measured images $\mathbf{x}_j$'s. In general, coefficients of all the multilinear constraints are minors of the motion matrix A. As for relationships among these coefficients, it is also known that the following statement is true [7]:

PROPOSITION 3.2 (Multilinear Constraint Dependency). *The coefficients of trilinear or quadrilinear constraints are functions of those of all bilinear (epipolar) constraints (or equivalently the corresponding fundamental matrices) given that the locations of the camera centers do not lie on a straight line.*

This proposition states a very important fact: Information about the camera motion is already fully contained in the bilinear constraints unless the camera center moves in a straight line – such a motion is also called **rectilinear motion**. This degenerate case is illustrated geometrically in Figure 3.1. In fact, a set of points $\{\mathbf{x}_j\}_{j=1}^m$ on m image planes satisfy all multilinear constraints **if and only if** "rays" extending from camera centers along these image points intersect at a *unique* point in 3D. As a consequence of this geometric interpretation of multilinear constraints, in order for an extra image to satisfy all multilinear constraints, it only needs to satisfy two (bilinear) coplanar constraints given that the new camera center is not collinear with the previous ones. For example, in Figure 3.2, in order for the fourth image to satisfy all multilinear constraints, it is sufficient for the ray (o_4, q) to be coplanar with the ray (o_2, q) and the ray (o_3, q). The coplanar condition between the ray (o_4, q) and the ray (o_1, q) is redundant.

4. Normalized Epipolar Constraint for Multiple Images. Multilinear constraints have conventionally been used to formulate various objective functions for motion recovery. However, if we do use them as constraints, we only need to pick a minimal set of independent ones. The minimal requirement is needed for Lagrangian

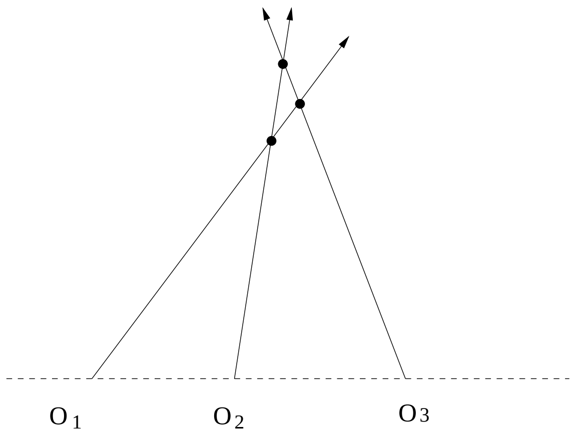

FIG. 3.1. *Degeneracy: Centers of camera lie on a straight line. Coplanar constraints are not sufficient to uniquely determine the intersection hence trilinear constraints are needed.*

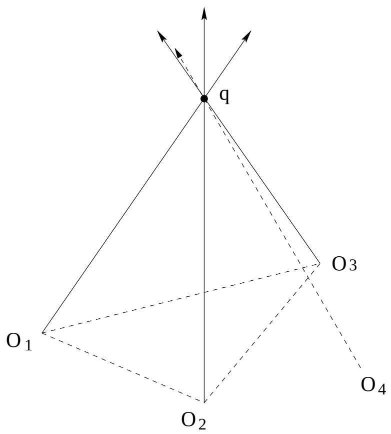

FIG. 3.2. *Sufficiency: Centers of camera and the point are not coplanar. Three (bilinear) coplanar constraints are sufficient to uniquely determine the intersection.*

multipliers to have a unique solution for each critical point of the objective function. The dependency among multilinear constraints suggests that if the centers of the camera do not lie on a straight line, pairwise epipolar constraints already provide a sufficient set of constraints. In this paper we will assume this condition is satisfied unless otherwise stated – Comments 2 and 5 will discuss the degenerate case. Furthermore, the (pairwise) epipolar constraints among three consecutive images naturally give a minimal set of independent constraints. In this section, we show how to use these constraints to derive a clean form of an optimal objective function for motion (and structure) recovery. In the next section, we will show how to use geometric optimization techniques to find the *optimal* solution which minimizes the objective function derived here.

The rigid body motion between the k^{th} and j^{th} camera frames is $g_{kj} = (R_{kj}, p_{kj}) \in SE(3)$, $1 \leq k, j \leq m$. Thus the coordinates of a 3D point $q \in \mathbb{R}^3$ with respect to frames j and k are related by:

$$q_k = R_{kj} q_j + p_{kj}. \tag{4.1}$$

Let us denote by $E_{jk} = R_{kj}^T \widehat{p}_{kj} \in \mathbb{R}^{3\times 3}$ the essential matrix associated with the camera motion between the k^{th} and j^{th} frames, then in absence of noise, image points $\mathbf{x}_j^i$ satisfy the epipolar constraints:

$$\mathbf{x}_j^{iT} E_{jk} \mathbf{x}_k^i = 0. \tag{4.2}$$

In presence of *isotropic* noises, we seek for points $\tilde{\mathbf{x}} = \{\tilde{\mathbf{x}}_j^i\}$ on the image plane and a configuration of m camera frames $\mathcal{G} = \{g_{kj}\}$ such that they minimize the total **reprojection error**. That is, we want to minimize the objective:

$$F(\mathcal{G}, \tilde{\mathbf{x}}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \|\tilde{\mathbf{x}}_j^i - \mathbf{x}_j^i\|^2 \tag{4.3}$$

subject to the constraints:

$$\tilde{\mathbf{x}}_j^{iT} E_{j,j+1} \tilde{\mathbf{x}}_{j+1}^i = 0, \quad \tilde{\mathbf{x}}_k^{iT} E_{k,k+2} \tilde{\mathbf{x}}_{k+2}^i = 0, \quad \tilde{\mathbf{x}}_l^{iT} e_3 = 1 \tag{4.4}$$

where $e_3 = (0,0,1)^T \in \mathbb{R}^3, 1 \leq j \leq m-1, 1 \leq k \leq m-2, 1 \leq l \leq m$ and $1 \leq i \leq n$. The first two constraints are epipolar constraints among three consecutive images. From the previous section, we know that they form a minimal (but sufficient) set of constraints among multiview images for a generic configuration. The last constraint is for the imaging model of perspective projection.[1] Using **Lagrangian multipliers**, the above constrained optimization problem is equivalent to minimizing:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \Big(\|\tilde{\mathbf{x}}_j^i - \mathbf{x}_j^i\|^2 + \sum_{k=j+1}^{j+2} \alpha_{jk}^i \tilde{\mathbf{x}}_j^{iT} E_{jk} \tilde{\mathbf{x}}_k^i 1_{k\leq m} + \beta_j^i (\tilde{\mathbf{x}}_j^{iT} e_3 - 1) \Big) \tag{4.5}$$

for some $\alpha_{jk}^i, \beta_j^i \in \mathbb{R}$. From the necessary condition $\nabla F = 0$ for local minima,

$$2(\tilde{\mathbf{x}}_j^i - \mathbf{x}_j^i) + \sum_{k=j+1}^{j+2} \alpha_{jk}^i E_{jk} \tilde{\mathbf{x}}_k^i 1_{k\leq m} + \sum_{k=j-2}^{j-1} \alpha_{kj}^i E_{kj}^T \tilde{\mathbf{x}}_k^i 1_{k\geq 1} + \beta_j^i e_3 = 0 \tag{4.6}$$

[1] Without loss of generality, we will only discuss the perspective projection case. The spherical projection case is similar and hence omitted for simplicity.

for all $i = 1, \ldots, n,\ j = 1, \ldots, m$. Multiplying the above equation by $\widehat{e}_3^T\widehat{e}_3$ to eliminate β_j^i, we obtain:

$$2(\mathbf{x}_j^i - \tilde{\mathbf{x}}_j^i) = \widehat{e}_3^T\widehat{e}_3\Big(\sum_{k=j+1}^{j+2} \alpha_{jk}^i E_{jk}\tilde{\mathbf{x}}_k^i 1_{k\le m} + \sum_{k=j-2}^{j-1} \alpha_{kj}^i E_{kj}^T\tilde{\mathbf{x}}_k^i 1_{k\ge 1}\Big) \tag{4.7}$$

for all $i = 1, \ldots, n,\ j = 1, \ldots, m$. It is readily seen that, in order to convert the above constrained optimization to an unconstrained one, we need to solve for α_{kj}^i and α_{jk}^i's. For this purpose, we define vectors $\tilde{\mathbf{x}}^i, \mathbf{x}^i, \Delta\mathbf{x}^i \in \mathbb{R}^{3m}$ associated with the i^{th} point as $\tilde{\mathbf{x}}^i = \left[\tilde{\mathbf{x}}_1^{iT}, \ldots, \tilde{\mathbf{x}}_m^{iT}\right]^T$, $\mathbf{x}^i = \left[\mathbf{x}_1^{iT}, \ldots, \mathbf{x}_m^{iT}\right]^T$, $\Delta\mathbf{x}^i = \mathbf{x}^i - \tilde{\mathbf{x}}^i$, the vector of all Lagrangian multipliers as:

$$\alpha^i = [\alpha_{12}^i, \alpha_{13}^i, \alpha_{23}^i, \alpha_{24}^i, \alpha_{34}^i, \ldots, \alpha_{m-2,m}^i, \alpha_{m-1,m}^i]^T \in \mathbb{R}^{2m-3}, \tag{4.8}$$

and matrix $D \in \mathbb{R}^{3m\times 3m}$ with $\widehat{e}_3^T\widehat{e}_3$ as diagonal blocks:

$$D = \begin{bmatrix} \widehat{e}_3^T\widehat{e}_3 & \cdots & 0_{3\times 3} \\ \vdots & \ddots & \vdots \\ 0_{3\times 3} & \cdots & \widehat{e}_3^T\widehat{e}_3 \end{bmatrix}. \tag{4.9}$$

We define, for $m \ge 3$, matrices $E = E(m) \in \mathbb{R}^{3m\times 3(2m-3)}$ and $\tilde{X}^i = \tilde{X}^i(m) \in \mathbb{R}^{3m\times(2m-3)}$ recursively as:

$$E(m) = \left[\begin{array}{c|c} E(m-1) & 0_{(3m-9)\times 6} \\ 0_{3\times 3(2m-5)} & E_m \end{array}\right] \text{ and } \tilde{X}^i(m) = \left[\begin{array}{c|c} \tilde{X}^i(m-1) & 0_{(3m-9)\times 2} \\ 0_{3\times(2m-5)} & \tilde{X}_m^i \end{array}\right],$$

with

$$E(2) = \begin{bmatrix} E_{12} \\ E_{12}^T \end{bmatrix},\ E_m = \begin{bmatrix} E_{m-2,m} & 0_{3\times 3} \\ 0_{3\times 3} & E_{m-1,m} \\ E_{m-2,m}^T & E_{m-1,m}^T \end{bmatrix},$$

$$\tilde{X}^i(2) = \begin{bmatrix} \tilde{\mathbf{x}}_2^i \\ \tilde{\mathbf{x}}_1^i \end{bmatrix},\ \tilde{X}_m^i = \begin{bmatrix} \tilde{\mathbf{x}}_m^i & 0_{3\times 1} \\ 0_{3\times 1} & \tilde{\mathbf{x}}_m^i \\ \tilde{\mathbf{x}}_{m-2}^i & \tilde{\mathbf{x}}_{m-1}^i \end{bmatrix}.$$

We define the **pseudo-array multiplication** $E \cdot \tilde{X}^i$ recursively as:

$$E(m)\cdot\tilde{X}^i(m) = \left[\begin{array}{c|c} E(m-1)\cdot\tilde{X}^i(m-1) & 0_{(3m-9)\times 2} \\ 0_{3\times(2m-5)} & E_m\cdot\tilde{X}_m^i \end{array}\right] \tag{4.10}$$

with

$$E(2)\cdot\tilde{X}^i(2) = \begin{bmatrix} E_{12}\tilde{\mathbf{x}}_2^i \\ E_{12}^T\tilde{\mathbf{x}}_1^i \end{bmatrix}, \quad E_m\cdot\tilde{X}_m^i = \begin{bmatrix} E_{m-2,m}\tilde{\mathbf{x}}_m^i & 0_{3\times 1} \\ 0_{3\times 1} & E_{m-1,m}\tilde{\mathbf{x}}_m^i \\ E_{m-2,m}^T\tilde{\mathbf{x}}_{m-2}^i & E_{m-1,m}^T\tilde{\mathbf{x}}_{m-1}^i \end{bmatrix}. \tag{4.11}$$

Using this notation, (4.7) can be rewritten as:

$$2\Delta\mathbf{x}^i = DE\cdot\tilde{X}^i\alpha^i. \tag{4.12}$$

Note that D is a projection matrix, *i.e.*, $D^2 = D$. All the constraints in (4.4) can then be rewritten compactly as two matrix equations:

$$\tilde{\mathbf{x}}^{iT} E \cdot \tilde{X}^i = 0, \quad D\Delta\mathbf{x}^i = \Delta\mathbf{x}^i. \tag{4.13}$$

The first equation is simply a matrix expression for all the epipolar constraints. Thus we can solve (4.12) for α^i:

$$\alpha^i = 2\left((E \cdot \tilde{X}^i)^T D E \cdot \tilde{X}^i\right)^{-1} (E \cdot \tilde{X}^i)^T \mathbf{x}^i \tag{4.14}$$

given that the matrix $G = (E \cdot \tilde{X}^i)^T D E \cdot \tilde{X}^i$ is invertible. We call matrix G the **observability Grammian**.

COMMENT 1 (Observability Grammian). *In general, the observability Grammian is invertible even in cases that the algorithm is not designed for, i.e., the camera motions are such that optical centers lie on a straight line, except for points on the line. In fact, 3D points which make the Grammian degenerate, i.e.,* $\det(G) = 0$, *are very rare. Geometrically, it means that, given a sequence of camera motions, the 3D location of a point whose images make the Grammian degenerate is not* **observable**. *For example, for camera translating in a straight line, points on the line itself satisfy* $\det(G) = 0$ *hence their images contain no information about either their 3D location or the camera motion on the line. In this sense, G can be thought of as the* **observability matrix** *in control theory.*

Substituting the expression for α^i (4.14) into (4.12), we then obtain the expression for $\Delta\mathbf{x}^i$ and we have:

$$\|\Delta\mathbf{x}^i\|^2 = \mathbf{x}^{iT} E \cdot \tilde{X}^i \left((E \cdot \tilde{X}^i)^T D E \cdot \tilde{X}^i\right)^{-1} (E \cdot \tilde{X}^i)^T \mathbf{x}^i. \tag{4.15}$$

Substituting this expression into the objective function $F(\mathcal{G}, \tilde{\mathbf{x}})$ we obtain:

$$F(\mathcal{G}, \tilde{\mathbf{x}}) = \sum_{i=1}^{n} \mathbf{x}^{iT} E \cdot \tilde{X}^i \left((E \cdot \tilde{X}^i)^T D E \cdot \tilde{X}^i\right)^{-1} (E \cdot \tilde{X}^i)^T \mathbf{x}^i. \tag{4.16}$$

For $m = 2$ the objective function reduces to:

$$F(\mathcal{G}, \tilde{\mathbf{x}}) = \sum_{i=1}^{n} \frac{(\mathbf{x}_1^{iT} E_{12} \tilde{\mathbf{x}}_2^i + \tilde{\mathbf{x}}_1^{iT} E_{12} \mathbf{x}_2^i)^2}{||\widehat{e}_3 E_{12} \tilde{\mathbf{x}}_2^i||^2 + ||\widehat{e}_3 E_{12}^T \tilde{\mathbf{x}}_1^i||^2}. \tag{4.17}$$

Hence, the terms on the right hand side of (4.16) are exactly multiview versions of the **crossed normalized epipolar constraints**, but *by no means* a trivial sum of the pairwise crossed normalized epipolar constraints [6]. In order to minimize $F(\mathcal{G}, \tilde{\mathbf{x}})$, we need to iterate between the camera motion $\mathcal{G}$ and triangulated structure $\tilde{\mathbf{x}}$, which would be essentially a multiview version of the **optimal triangulation** procedure proposed in [6]. In this paper, however, we will only demonstrate how to obtain optimal motion estimates. Note that, in the expression for $F(\mathcal{G}, \tilde{\mathbf{x}})$, the matrix $\tilde{X}^i$ is a function of $\tilde{\mathbf{x}}^i$ instead of the measured $\mathbf{x}^i$. In general, the difference between $\tilde{\mathbf{x}}^i$ and $\mathbf{x}^i$ is small. Therefore, we may approximate $\tilde{X}^i$ by replacing $\tilde{\mathbf{x}}_j^i$ in $\tilde{X}^i$ by the known

$\mathbf{x}_j^i$. We call the resulting matrix X^i. We then obtain a new function of the camera motion only, $F_n(\mathcal{G}) = F(\mathcal{G}, \mathbf{x})$:

$$F_n(\mathcal{G}) = \sum_{i=1}^{n} \mathbf{x}^{iT} E \cdot X^i \left((E \cdot X^i)^T D E \cdot X^i\right)^{-1} (E \cdot X^i)^T \mathbf{x}^i. \tag{4.18}$$

In absence of noise, each term of $F_n(\mathcal{G})$ should be:

$$\mathbf{x}^{iT} E \cdot X^i \left((E \cdot X^i)^T D E \cdot X^i\right)^{-1} (E \cdot X^i)^T \mathbf{x}^i = 0. \tag{4.19}$$

We call this the **normalized epipolar constraint** for multiple images. This is a natural generalization of the normalized epipolar constraint in the two view case [6]. Thus, as in the two view case, $F_n(\mathcal{G})$ can be regarded as a statistically adjusted objective function for directly estimating the camera motions.

COMMENT 2 (Bilinear vs. Trilinear Constraints). *It is true that one can also use a set of independent trilinear constraints to replace those in (4.4) and, with a similar exercise, derive its normalized version for motion (and structure) estimation. However, trilinear tensors (as functions of camera motions) do not have as good of a geometric structure as the bilinear ones. This makes the associated optimization problem harder to describe, even though it is essentially an equivalent optimization problem. One must also be aware that, in the rectilinear motion case, the normalized epipolar constraint objective F_n is not supposed to have a unique minimum (as we will soon see in Simulation 3, in presence of noise, this is not completely true. We will discuss further the new meaning of the minimum in Comment 5) while the corresponding normalized trilinear one always gives a unique solution.*

COMMENT 3 (Calibrated vs. Uncalibrated Camera). *In the case of an uncalibrated camera, nothing substantial will change in the above derivation except that the essential matrices need to be replaced by fundamental matrices and that the camera intrinsic parameters will introduce 5 new unknowns.*

5. Geometric Optimization Techniques. F_n in the previous section is a function defined on the space of configurations of m camera frames, which is not a regular Euclidean space. Thus conventional optimization techniques cannot be directly applied to minimize F_n. In this section, we show how to apply newly developed geometric optimization techniques [2, 13] to solve this problem. Here we will adopt the Newton's method, although it may not be the fastest, because it allows us to compute the Hessian of the objective function which is potentially useful for sensitivity analysis.

The configuration $\mathcal{G}$ of m camera frames is determined by relative rotations and translations:

$$\mathcal{R} = [R_{21}, R_{32}, \ldots, R_{m,m-1}] \in SO(3)^{m-1}, \tag{5.1}$$

$$\mathcal{P} = \left[p_{21}^T, p_{32}^T, \ldots, p_{m,m-1}^T\right]^T \in \mathbb{R}^{3m-3}. \tag{5.2}$$

Then $F_n(\mathcal{G})$ can be denoted as $F_n(\mathcal{R}, \mathcal{P})$. It is direct to check that $F_n(\mathcal{R}, \lambda\mathcal{P}) = F_n(\mathcal{R}, \mathcal{P})$ for all $\lambda \neq 0$. Thus $F_n(\mathcal{R}, \mathcal{P})$ is a function defined on the manifold $M = SO(3)^{m-1} \times \mathbb{S}^{3m-4}$ where $\mathbb{S}^{3m-4}$ is a $3m-4$ dimensional spheroid. M is simply a product of Stiefel manifolds and it has total dimension $6m-7$. Furthermore, the (induced) Euclidean metrics on $SO(3)$ and $\mathbb{S}^{3m-4}$ are the same as their canonical metrics as Stiefel manifolds. This gives a natural Riemannian metric $\Phi(\cdot,\cdot)$ on the total manifold M. Note that any tangent vector $\mathcal{X} \in T_{(\mathcal{R},\mathcal{P})}M$ can be represented as $\mathcal{X} = (\mathcal{X}_{\mathcal{R}}, \mathcal{X}_{\mathcal{P}})$, with $\mathcal{X}_{\mathcal{R}} \in T_{\mathcal{R}}(SO(3)^{m-1})$ and $\mathcal{X}_{\mathcal{P}} \in T_{\mathcal{P}}(\mathbb{S}^{3m-4})$ defined by the expressions:

$$\mathcal{X}_{\mathcal{R}} = [R_{21}\widehat{\omega}_{21}, \ldots, R_{m,m-1}\widehat{\omega}_{m,m-1}], \tag{5.3}$$

$$\mathcal{X}_{\mathcal{P}} = \left[\mathcal{X}_{21}^T, \ldots, \mathcal{X}_{m,m-1}^T\right]^T \tag{5.4}$$

where $\omega_{i+1,i} \in \mathbb{R}^3, \mathcal{X}_{i+1,i} \in \mathbb{R}^3, i = 1, \ldots, m-1$ and ${\mathcal{X}_{\mathcal{P}}}^T\mathcal{P} = 0$. Then the Riemannian metric $\Phi(\cdot,\cdot)$ on the manifold M is explicitly given by:

$$\Phi(\mathcal{X}, \mathcal{X}) = \sum_{i=1}^{m-1} \omega_{i+1,i}^T \omega_{i+1,i} + \mathcal{X}_{\mathcal{P}}^T \mathcal{X}_{\mathcal{P}}. \tag{5.5}$$

Similar to the two view case [6], we can directly apply the Riemannian optimization schemes developed in [2, 13] for minimizing the function $F_n(\mathcal{R},\mathcal{P})$.

Riemannian Newton's Algorithm for Minimizing $F_n(\mathcal{R},\mathcal{P})$:

1. *Pick an orthonormal basis $\{\mathcal{B}^i\}_{i=1}^{6m-7}$ on $T_{(\mathcal{R},\mathcal{P})}M$. Compute the vector $\mathbf{g} \in \mathbb{R}^{6m-7}$ with its i^{th} entry given by $(\mathbf{g})_i = dF_n(\mathcal{B}^i)$. Compute the matrix $\mathbf{H} \in \mathbb{R}^{(6m-7)\times(6m-7)}$ with its $(i,j)^{th}$ entry given by $(\mathbf{H})_{i,j} = HessF_n(\mathcal{B}^i, \mathcal{B}^j)$. Compute the vector $\delta = -\mathbf{H}^{-1}\mathbf{g} \in \mathbb{R}^{6m-7}$.*
2. *Recover the vector $\Delta \in T_{(\mathcal{R},\mathcal{P})}M$ whose coordinates with respect to the orthonormal basis $\mathcal{B}^i$'s are exactly δ. Update the point $(\mathcal{R},\mathcal{P})$ along the geodesic to* $\exp(\Delta)$.
3. *Repeat step 1 if $\|\mathbf{g}\| \geq \epsilon$ for some pre-specified tolerance $\epsilon > 0$.*

In the above algorithm, we still need to know: how to pick an orthonormal basis on TM, how to compute geodesics on the manifold M and how to compute the gradient and Hessian of F_n.

Using the Gram-Schmidt process, we can find vectors $V_{\mathcal{P}}^1, \ldots, V_{\mathcal{P}}^{3m-4} \in \mathbb{R}^{3m-3}$ such that, together with $\mathcal{P}$, they form an orthonormal basis of $\mathbb{R}^{3m-3}$. Let $e_1, e_2, e_3 \in \mathbb{R}^3$ be the standard orthonormal basis of $\mathbb{R}^3$. Then a natural orthonormal basis $\{\mathcal{B}^i\}_{i=1}^{6m-7}$ on $T_{(\mathcal{R},\mathcal{P})}M$ is given by:

$$\mathcal{B}^{3i-3+j} = \left([0, \ldots, 0, R_{i+1,i}\widehat{e}_j, 0, \ldots, 0], \mathbf{0}\right) \tag{5.6}$$

for $1 \leq i \leq m-1,\ 1 \leq j \leq 3$ and

$$\mathcal{B}^{3m-3+i} = \left(\mathbf{0}, V_{\mathcal{P}}^i\right), \quad \text{for } 1 \leq i \leq 3m-4. \tag{5.7}$$

Given a vector $\mathcal{X} = (\mathcal{X}_{\mathcal{R}}, \mathcal{X}_{\mathcal{P}}) \in T_{(\mathcal{R},\mathcal{P})}M$ with $\mathcal{X}_{\mathcal{R}}$ and $\mathcal{X}_{\mathcal{P}}$ given by (5.3) and (5.4) respectively, the geodesic $(\mathcal{R}(t), \mathcal{P}(t)) = \exp(\mathcal{X}t), t \in \mathbb{R}$ is given by:

$$\mathcal{R}(t) = (R_{21}e^{t\widehat{\omega}_{21}}, R_{32}e^{t\widehat{\omega}_{32}}, \ldots, R_{m,m-1}e^{t\widehat{\omega}_{m,m-1}}), \tag{5.8}$$

$$\mathcal{P}(t) = \mathcal{P}\cos(\sigma t) + U\sin(\sigma t),\ \ \sigma = \|\mathcal{X}_{\mathcal{P}}\|, U = \mathcal{X}_{\mathcal{P}}/\sigma. \tag{5.9}$$

The tangent of this geodesic at $t = 0$ is exactly $\mathcal{X}$.

With an orthonormal basis, the computation of the gradient and the Hessian can be reduced to directional derivatives along geodesics on M. Given a vector $\mathcal{X} \in T_{(\mathcal{R},\mathcal{P})}M$, let $(\mathcal{R}(t), \mathcal{P}(t)) = \exp(\mathcal{X}t)$. Then we have:

$$\text{(5.10)}\quad dF_n(\mathcal{X}) = \left.\frac{dF_n(\mathcal{R}(t), \mathcal{P}(t))}{dt}\right|_{t=0}, \quad \mathrm{Hess}F_n(\mathcal{X}, \mathcal{X}) = \left.\frac{d^2F_n(\mathcal{R}(t), \mathcal{P}(t))}{dt^2}\right|_{t=0}.$$

Polarizing $\mathrm{Hess}F_n(\mathcal{X}, \mathcal{X})$ we can obtain the expression of $\mathrm{Hess}F_n(\mathcal{X}, \mathcal{Y})$ for arbitrary $\mathcal{X}, \mathcal{Y} \in T_{(\mathcal{R},\mathcal{P})}M$:

$$\text{(5.11)}\quad \mathrm{Hess}F_n(\mathcal{X}, \mathcal{Y}) = \frac{1}{4}\big(\mathrm{Hess}F_n(\mathcal{X}+\mathcal{Y}, \mathcal{X}+\mathcal{Y}) - \mathrm{Hess}F_n(\mathcal{X}-\mathcal{Y}, \mathcal{X}-\mathcal{Y})\big).$$

According to the definition of gradient, $\mathrm{grad}F_n \in T_{(\mathcal{R},\mathcal{P})}M$, which is given by:

$$\text{(5.12)}\qquad dF_n(\mathcal{X}) = \Phi(\mathrm{grad}F_n, \mathcal{X}), \quad \forall \mathcal{X} \in T_{(\mathcal{R},\mathcal{P})}M,$$

is exactly equal to the 1-form dF_n with respect to an orthonormal frame. Therefore, at each point $(\mathcal{R}, \mathcal{P})$, we pick the orthonormal basis $\{\mathcal{B}^i\}_{i=1}^{6m-7}$ on $T_{(\mathcal{R},\mathcal{P})}M$ as above and compute the first and second order derivatives of F_n with respect to the corresponding geodesics of the base vectors. The gradient and Hessian of F_n are then explicitly expressed by the vector $\mathbf{g}$ and the matrix $\mathbf{H}$ as described in the above algorithm. The updating vector Δ computed in the algorithm is in fact intrinsically defined[2] and satisfies:

$$\text{(5.13)}\qquad \mathrm{Hess}F_n(\Delta, \mathcal{X}) = \Phi(-\mathrm{grad}F_n, \mathcal{X}), \quad \forall \mathcal{X} \in T_{(\mathcal{R},\mathcal{P})}M.$$

Note that F_n has a very good structure – only matrix E depends on $(\mathcal{R}, \mathcal{P})$ and it consists of blocks of essential matrices $E_{j,j+1}$ and $E_{j,j+2}$. The computation of the Hessian can then be reduced to computing derivatives of these matrices with respect to the chosen base vectors. From the definition of the essential matrix E_{jk}, we have:

$$\text{(5.14)}\quad E_{j,j+1} = R_{j+1,j}^T \widehat{p}_{j+1,j}, \quad E_{j,j+2} = E_{j,j+1}R_{j+2,j+1}^T + R_{j+1,j}^T E_{j+1,j+2}.$$

Hence the computation can be further reduced to derivatives of essential matrix $E_{j,j+1}$ only. For a vector $\mathcal{X} \in T_{(\mathcal{R},\mathcal{P})}M$ of the form given by (5.3) and (5.4), by direct computation, we have:

$$\text{(5.15)}\qquad \begin{aligned} dE_{j,j+1}(\mathcal{X}) &= \widehat{\omega}_{j+1,j}^T R_{j+1,j}^T \widehat{p}_{j+1,j} + R_{j+1,j}^T \widehat{\mathcal{X}}_{j+1,j}, \\ \mathrm{Hess}E_{j,j+1}(\mathcal{X}, \mathcal{X}) &= \widehat{\omega}_{j+1,j}^2 R_{j+1,j}^T \widehat{p}_{j+1,j} + 2\widehat{\omega}_{j+1,j}^T R_{j+1,j}^T \widehat{\mathcal{X}}_{j+1,j} \\ &\quad - \mathcal{X}_{j+1,j}^T \mathcal{X}_{j+1,j} R_{j+1,j}^T \widehat{p}_{j+1,j} \end{aligned}$$

for $j = 1, \ldots, m-1$. Note that these formulas are consistent with the corresponding ones in the two view case. Thus we now have all the necessary ingredients for implementing the proposed optimization scheme. For any given number of camera frames, we get an optimal estimate of the camera relative configuration by minimizing the normalized epipolar objective F_n.

[2]That is, the definition of Δ is independent of the choice of coordinate frame.

TABLE 6.1
Simulation parameters

Parameter	Unit	Value
Number of trials		100 - 500
Number of points		20
Number of frames		3-4
Field of view	degrees	90
Depth variation	u.f.l.	100 - 400
Image size	pixels	500 $\times$ 500

COMMENT 4 (Newton vs. Levenberg-Marquardt). *The difference between Newton and Levenberg - Marquardt (LM) methods is that in LM the Hessian is approximated by some form of the objective function's gradient. Since the gradient only involves first order derivatives, LM in general is much less costly in each step. From our implementation of the Newton's algorithm, the Hessian indeed takes more than 95% of the computing time. Nevertheless, we computed the Hessian anyway since the formula would be useful for future sensitivity analysis of motion estimation in the multiview case.*

6. Simulations on Synthetic Data. In this section, we show by simulations the performance of the normalized epipolar constraint.We will apply it to cases *with* or *without* the sufficiency of the epipolar constraint satisfied.

6.1. Setup. Table 6.1 shows the simulation parameters used. In the table, u.f.l. stands for *units of focal length.* The ratio of the magnitude of translation $||p||$ and rotation θ, or simply the *T/R ratio*, is compared at the center of the random cloud scattered in the truncated pyramid specified by the given field of view and depth variation (see Figure 6.1). For all simulations, independent Gaussian noise with std given in pixels is added to each image point. In general, the amount of rotation between consecutive frames is about 20^o and the amount of translation is then automatically given by the T/R ratio. In the following, camera motions will be specified by their translation and rotation axes. For example, between a pair of frames, the symbol XY means that the translation is along the X-axis and rotation is along the Y-axis. If n such symbols are connected by hyphens, it specifies a sequence of consecutive motions. Error measure for rotation is $\arccos\left(\frac{tr(R\tilde{R}^T)-1}{2}\right)$ in degrees where $\tilde{R}$ is an estimate of the true R. Error measure for translation is the angle between p and $\tilde{p}$ in degrees where $\tilde{p}$ is an estimate of the true p. All nonlinear (two view or multiview) algorithms are initialized by estimates from the conventional two view linear algorithm. Since the translation estimates of the linear algorithm are given up to scale only, for the multiview case an initialization of the relative scale between consecutive translations is required. This is done by triangulation since the directions of the translations are known. For example, the relative scale between p_{21} and p_{32} is $\sin(\alpha)/\sin(\gamma)$ where α is the angle between p_{31} and $R_{21}p_{21}$ and γ is the angle between p_{23} and $R_{13}p_{13}$.

6.2. Simulation 1: Comparison with Two Frame Bilinear and Normalized Epipolar Constraints. Figure 6.2 plots the errors of rotation estimates and

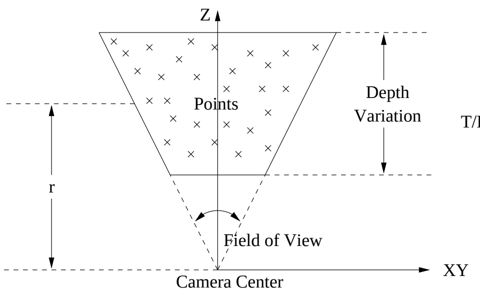

FIG. 6.1. *Simulation setup*

translation estimates compared with results from the standard 8-point linear algorithm and nonlinear algorithm for pairwise views [6]. As we see, normalization among multiple images indeed performs better than normalization among pairwise images only.

6.3. Simulation 2: Axis Dependency Profile. We run the multiview algorithm with consecutive motions along the *same* rotation and translation axes for all nine possible combinations (see Figure 6.3). Note that our multiview algorithm is not designed to work in rectilinear motion case, such as XX-XX, YY-YY and ZZ-ZZ. Nevertheless, the simulation results in the figure show that the translation estimates still converge to the correct translational direction and the error angles between estimates and the true ones are comparable to other generic cases. As we see, the estimate error is larger when translation along the Z-axis is present. This is because of a smaller signal to noise ratio in this case.

6.4. Simulation 3: A Statistically Stable Solution for Rectilinear Motion from Normalized Epipolar Constraint. ¿From the previous simulation, we notice that the algorithm indeed converges to the correct translational direction in the rectilinear motion case. Then how about the relative scales between consecutive translations? They are usually believed to be captured only by trilinear constraints but not by bilinear ones. This is *not completely true.* The rectilinear motion is indeed a degenerate case for the bilinear constraints, from which there is no unique solution for the relative scales (for example see Figure 3.1). However, statistically, the *true* relative scales must be a *stable* solution among all the possible ones. That is, if we properly normalize the epipolar constraint w.r.t. the noise model, the true relative scale should be captured by the epipolar constraints alone as a statistically stable solution. Here the noise essentially plays a positive role of "singling out" the stable solution which otherwise would be lost when degeneracy occurs. Figure 6.4 plots two histograms of relative scale estimates given by minimizing our normalized epipolar constraint: One is for rectilinear motion and the other one for generic motion. Clearly, in both cases, the histogram resembles a Gaussian distribution with the mean centered at the true scale, as a result of the proper normalization.

COMMENT 5. *(Bilinear vs. Trilinear Constraints Continued) Simulation 3 reveals a remarkable statistical relationship between bilinear and trilinear constraints:*

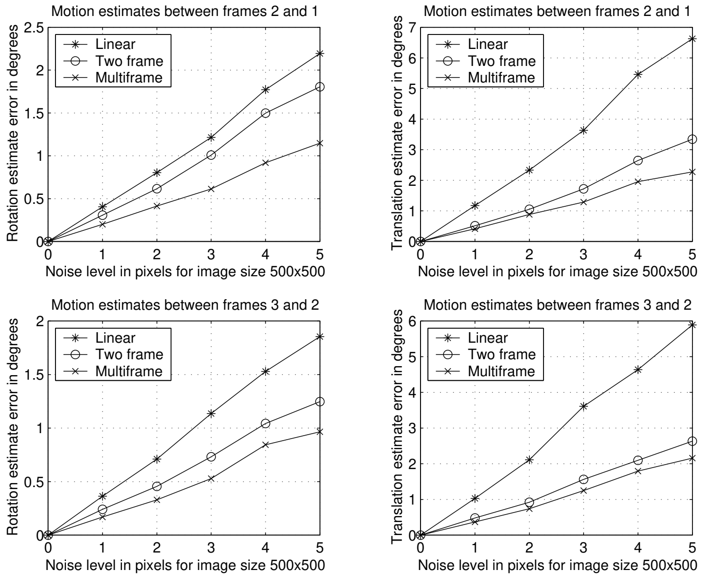

FIG. 6.2. *Motion estimate error comparison between normalized epipolar constraint of three frames, normalized epipolar constraint of two frames and (bilinear) epipolar constraint. The number of trials is 500, camera motions are XX-YY and T/R ratio is 1.*

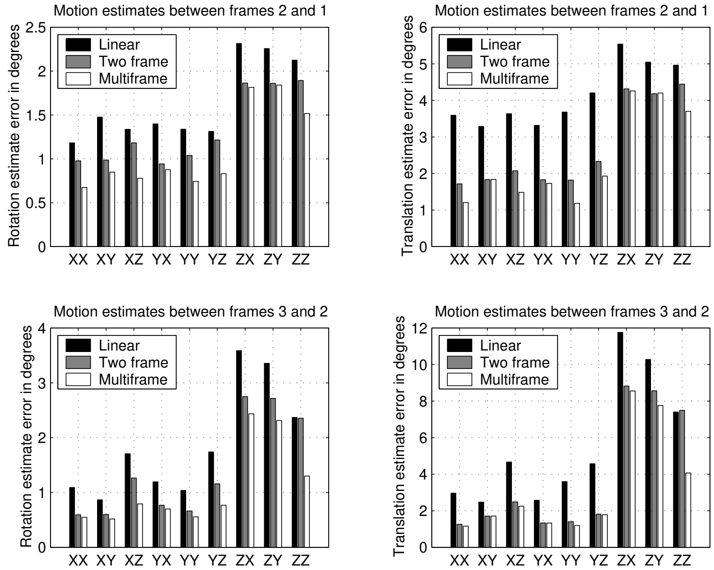

FIG. 6.3. *Axis dependency profile: The algorithms are run for all nine combinations of camera rotation and translation w.r.t. the X, Y and Z axes. The number of trials is 100, noise level is 3 pixel std and T/R ratio is 1.*

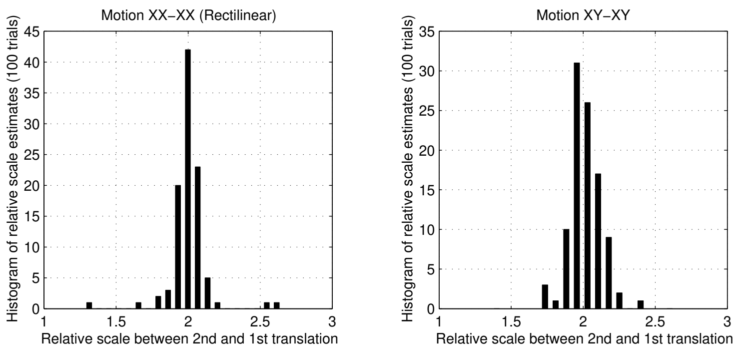

FIG. 6.4. *Histogram of relative scale estimates by normalized epipolar constraint in a rectilinear motion case and a generic motion case. The number of trial is 100, noise level is 3 pixel std and the true relative scale between consecutive translations is 2.*

If an optimal estimate is obtained for generic cases, it can still be retrieved as the stable estimate in a degenerate case – the (noise-free) deterministic constraint may be degenerate, but there is no reason for the a posteriori distribution of the estimate to be degenerate as well. Geometrically, the estimate obtained in a degenerate configuration can be interpreted as a "limit" of a sequence of estimates of generic configurations. Such an estimate may also be viewed as the so called "viscosity solution" of the normalized epipolar constraint if the Gaussian noise on the images is regarded as some kind of "diffusion". Therefore, in principle, *we do not really need trilinear constraints in order to estimate motion (including relative scales) correctly even in the rectilinear motion case, although such an estimate may be more sensitive or less robust (if the noise model changes).*

7. Experiments on Real Images. In this section we present three experiments, two with indoor image sequences and one with an outdoor image sequence. The first experiment uses an artificial pattern board to demonstrate structure reconstruction after recovering the motion of the camera using the multiview algorithm. The second experiment is done with an indoor lab sequence, with the camera undergoing rectilinear motion. The third experiment involves an outdoor scene with generic motion. These image sequences are chosen to test the algorithm under different environments and motions. The estimated motion is then compared with the ground truth data.

In order to work with real images, we need to establish correspondences across multiple frames. We adapt the algorithm from [23] for this purpose. We will briefly describe this algorithm in the following section.

7.1. Setup.

7.1.1. Feature Extraction. Corners are chosen to be the feature points. To implement feature extraction, we use the Harris corner detector [3] which is based on thresholding the operator:

$$C(\mathbf{x}) = det \begin{bmatrix} \widehat{I_x^2} & \widehat{I_x I_y} \\ \widehat{I_x I_y} & \widehat{I_y^2} \end{bmatrix} - k \cdot trace^2 \begin{bmatrix} \widehat{I_x^2} & \widehat{I_x I_y} \\ \widehat{I_x I_y} & \widehat{I_y^2} \end{bmatrix} \tag{7.1}$$

where I_x and I_y are the first order derivatives along the x and y directions, respectively. $\widehat{I_x^2}$, $\widehat{I_xI_y}$, and $\widehat{I_y^2}$ represent the Gaussian smoothing operation on the respective quantities. k is set to 0.04 to detect high contrast pixel edges.

7.1.2. Correlation Matching. To match features from one image to the next, we use a correlation window of size $(2w_x+1)\times(2w_y+1)$ centered around the feature point. Features from the first image are correlated with all the feature points from the second image within a search area of size $(2s_x+1)\times(2s_y+1)$. In our implementation, w_x and w_y are set to 7, and s_x and s_y are set to 100.

The correlation scores between each feature point in image 1 and all the feature points in image 2 within the search area are calculated. We designate the point in image 2 as a match if the correlation score is higher than a threshold of 0.8. If there are multiple matches, the point with the highest score is chosen to be the match. The above procedure is then repeated from image 2 to image 1. We use only the matches that are established from both image 1 to image 2 and from image 2 to image 1 in order to increase the accuracy of the correspondence matching.

7.1.3. Robust Estimation of the Epipolar Geometry. In order to improve the correspondences established in the previous section, the epipolar geometry of the scene is robustly estimated to compensate for possible outliers. We use the least median of squares (LMS) robust estimator as described in [11]. For comparisons of the various robust methods, please see [17].

The LMS method works by taking S randomly chosen subsets of the entire set of correspondences $\{\mathbf{x}_j^i\}$, $i=1\dots n, j=1,2$. For the l^{th} subset, the camera motion (R_{12}^l, p_{12}^l) between consecutive frames is estimated using the multiview algorithm. Then the subset giving the best motion estimate is obtained as:

$$\min_{l=1..S}\left(\operatorname*{med}_{i=1..n}\,[d^2(\mathbf{x}_2^i, E_{21}^l\mathbf{x}_1^i)+d^2(\mathbf{x}_1^i, E_{12}^l\mathbf{x}_2^i)]\right) \tag{7.2}$$

where $E_{21}^l = R_{12}^{lT}\widehat{p}_{12}^l = E_{12}^{lT}$ and

$$d(\mathbf{x}_2, E_{21}\mathbf{x}_1) = \frac{\|\mathbf{x}_2^T E_{21}\mathbf{x}_1\|}{\sqrt{(E_{21}\mathbf{x}_1)_1^2+(E_{21}\mathbf{x}_1)_2^2}} \tag{7.3}$$

is the distance between feature point $\mathbf{x}_2$ and its epipolar line $E_{21}\mathbf{x}_1$ [23]. In the formula, $(E_{21}\mathbf{x}_1)_i$ is the i^{th} component of $E_{21}\mathbf{x}_1$.

7.1.4. Correspondence Matching using the Epipolar Geometry. Once we have the epipolar geometry of the images from the previous section, we can use it to update our original correspondences found by correlation matching. We use the same methods as in Section 7.1.2, except that now the search area is limited to the epipolar lines in the two images. For any given feature point in image 1, we require that a corresponding feature point in image 2 lies within ρ pixels of the epipolar line.

7.1.5. Error in Feature Tracking. The accuracy of the feature tracking algorithm is measured by finding the distance between a matched point with its corresponding epipolar line, using equation (7.3). This distance is measured in pixels. Then the average distance for all the correspondences in all the image pairs of the sequence is calculated. From the experimental results, it appears that motion estimates with an average distance greater than 1 pixel are generally poor.

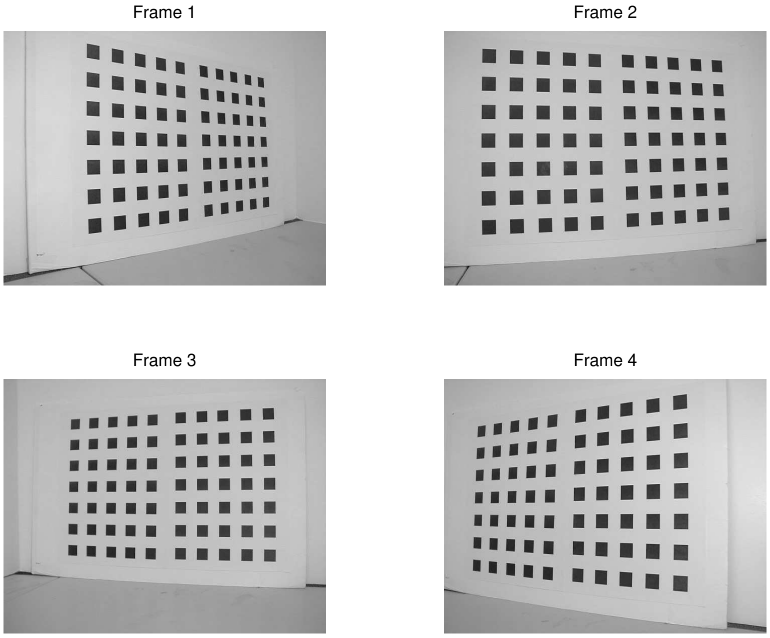

FIG. 7.1. *Four images of the calibration board.*

7.1.6. Camera Calibration. The camera is calibrated from a set of planar feature points using Zhang's technique described in [21]. The calibration board is shown in Figure 7.1. The pattern consists of 70 squares, giving a total of 280 corners. The corners are tracked through all four images. The camera is moved with unknown rotation and translation between each image. The estimated calibration values were consistent through multiple trials.

7.1.7. Initialization. The multiview algorithm is initialized by estimates from the conventional two view linear algorithm [6]. The relative scales between translations are initialized by triangulation (see section 6.1).

7.2. Experiment 1: Artificial 3D Pattern Reconstruction from Multiple Views. The purpose of this experiment is to show structure reconstruction once we recover the camera motion from multiple images of an artificial scene. We took images of the pattern board shown in Figure 7.2 from four different vantage points. The correspondences between the corners from different images are then established *by hand* for this experiment. These image points are used in the multiview algorithm to estimate for the motion of the camera. From the motion, the structure (locations) of the 3D points is reconstructed up to a general scale using linear least-squares [22]. The results are shown in Figure 7.2 from different view angles. The reconstructed points give a very accurate representation of the 3D points on the actual pattern board.

7.3. Experiment 2: Indoor Rectilinear Motion Sequence. We use 4 images of an indoor scene, with the motion of the camera in a straight line (rectilinear motion) along the Z-axis. The rotation R is set to the identity. The relative scale

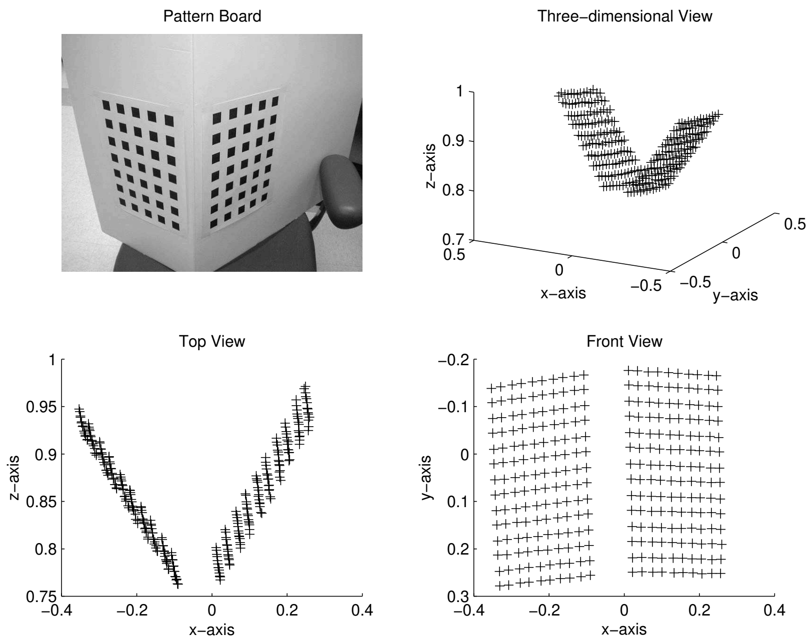

FIG. 7.2. *Four images of the pattern board*

between the first translation and the second translation is 2, and the scale between the second translation and the third translation is 0.5. The feature points are automatically extracted and the correspondences established by the robust feature tracking method described in Section 7.1. The correspondences are shown in Figure 7.3. There are 54 feature points matched through all four image frames. The average distance between the matching points and the corresponding epipolar lines for the entire sequence is 0.45 pixels. These points are used to recover the motion of the camera, using the multiview algorithm. The results are shown in Tables 7.1 and 7.2. Table 7.1 shows the error between the estimated motion and the actual motion of the camera. Table 7.2 shows the error of the relative scales between consecutive translations. The error measures for both rotation and translation follow the format used in Section 6.1.

As shown in Table 7.1, the rotation estimates tend to be more accurate than the translation estimates, which corroborates the computer simulation results of Section 6. Table 7.2 shows that the algorithm is able to recover the scale of consecutive translations in rectilinear motion, with the error below 7%. This confirms the simulation results from Section 6.4, namely, that it is possible to use only bilinear constraints to estimate motion, even in the case of rectilinear motion.

7.4. Experiment 3: Outdoor Generic Motion Sequence. This sequence consists of 4 images of an outdoor environment, with the camera undergoing motion in the YY-YX-YY (rotation-translation) axes. The relative scale between all the translations is 1:1. The correspondences are shown in Figure 7.4. There are 55 feature points matched through all four image frames. The average distance between

Table 7.1
Motion estimate errors in degrees

Frames	Rotation Errors	Translation Errors
1-2	0.78^o	9.0^o
2-3	1.94^o	2.8^o
3-4	0.91^o	1.7^o

Table 7.2
Scale estimate error

Translations	Scale Error
1-2	6.58%
2-3	1.52%

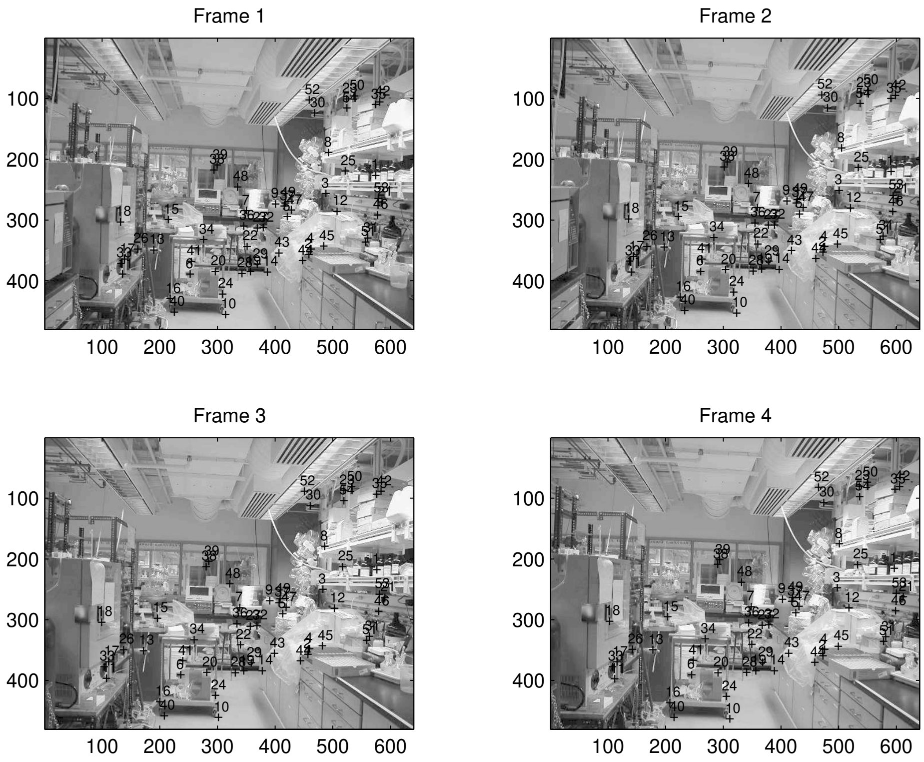

Fig. 7.3. *Indoor rectilinear motion image sequence*

the matching points and the corresponding epipolar lines for the entire sequence is 0.58 pixels. These points are used to recover the motion of the camera, using the multiview algorithm. The results are shown in Tables 7.3 and 7.4.

The estimates in general are worse than those of the indoor experiment. This is not unexpected. We will discuss the reason for this discrepancy later on in Section 7.5. For now, it is sufficient to note that the algorithm is able to recover motion in an outdoor setting, where the points are generally further away and the conditions more volatile. For example, the leaves on the trees as well as the grass on the lawn can shift positions (due to wind, shadows, etc) from image to image, independent of the camera motion. These factors can all contribute to the motion estimate error.

7.5. Analysis of Experiments. The experiments confirmed the results of the computer simulations with the algorithm tested on real images. In general, the cor-

TABLE 7.3
Motion estimate errors in degrees

Frames	Rotation Errors	Translation Errors
1-2	5.1^o	18.9^o
2-3	4.9^o	1.9^o
3-4	5.1^o	14.5^o

TABLE 7.4
Scale estimate error

Translations	Scale Error
1-2	9.0%
2-3	7.1%

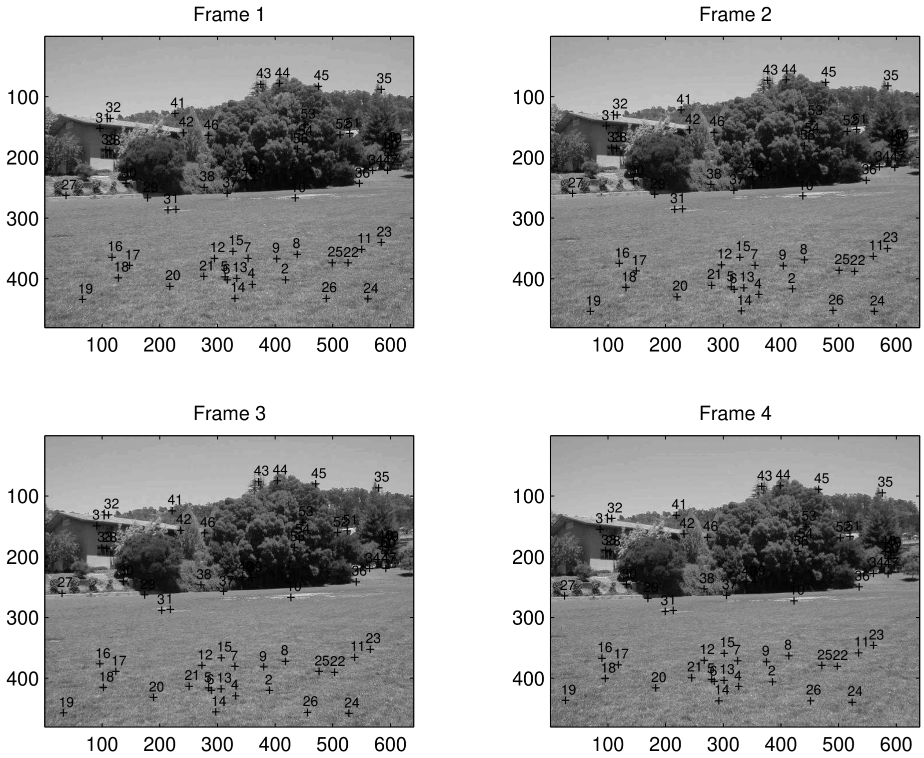

FIG. 7.4. *Outdoor generic motion image sequence*

rect motion was recovered by using the normalized epipolar constraint. The error of the motion estimates from Experiment 3 may seem high, but it should be noted that the number of feature points affects the accuracy of the estimation significantly. As we establish more correspondences in the images, the error of the estimated motion decreases. The number of correspondences used in the experiments in the previous sections should be considered as a *lower* bound on the number of correspondences needed for a good estimate using these images. The error values of the motion estimates from the experiments represent reasonable results, given that the number of correspondences is around 50 to 60 and the points are not clustered in the images.

The overall results from the indoor sequence are better than those from the outdoor sequence. That is to be expected considering that the feature points from the indoor sequence are closer to the camera in general. When the points are close, even a small amount of motion would cause a noticeable change in the position of the feature points. However, when the points are far away, even a large motion would not cause a significant change in the relative location of these points. ¿From these experiments, we found that it is very difficult to get good motion estimation when all the feature points are very far away. In order to get accurate estimates, at least some of the features must lie close to the camera. Indoor scenes tend to work better because the feature points are relatively close to the camera.

The total running time of the algorithm was around an hour to an hour and a half on a Sun UltraSparc 2 machine for a sequence of four images and around 50 to 60 feature points. About 36% of the processing time was spent on feature point extraction and matching, 58% of the time on robust estimation of the camera motion, and about 6% spent on reestablishing correspondences based on the estimated motion. However, it should be noted that our algorithm was not developed with the intent of being optimized for speed, and many improvements could be made to increase the speed of the program.

8. Conclusions and Discussions. In this paper, we contend by using (bilinear) epipolar constraints that multilinear constraints need to be properly normalized when used for motion (or structure) estimation. There are several consequences of such a normalization. First, the so obtained objective function is no longer linear hence it does not preserve the tensor structure of multilinear constraints. Second, such a normalization is a natural generalization of the well known normalized epipolar constraint between two images but by no means a trivial sum of them. Third, the normalization not only provides optimal motion (and structure) estimates but, more importantly, reveals certain statistical relationship between epipolar and trilinear constraints – as a necessary complement to the well known algebraic or geometric relationship. We now know that in principle normalized epipolar constraint alone suffices for estimating correct motion (as a statistically stable solution) even in the rectilinear motion case. However, more extensive simulation, experiments and analysis are still needed to evaluate how practical it is when applied to degenerate cases because it may be less robust to model changes. For example, in the case when the noise on the images is no longer isotropic or identically independently distributed, we do not know whether the rectilinear motion can still be well estimated. In a practical implementation, the reader is recommended to extend the idea of normalization in this paper to trilinear constraints or even to an uncalibrated camera.

In this paper, we use the generic Newton's algorithm to minimize the normalized epipolar constraint. One disadvantage is that it is slower than most gradient based algorithms, such as the commonly used Levenberg-Marquardt algorithm. For this reason, we recommend the reader to use those algorithms instead for practical implementations. We here outlined the Newton's algorithm to demonstrate how to compute all the necessary geometric entities associated with the optimization.

Acknowledgement. This work is supported by ARO under the MURI grant DAAH04-96-1-0341 and by AASERT under the grant DAAG55-97-1-0216.

REFERENCES

[1] K. Danillidis, *Visual Navigation*. Lawrence Erlbaum Associates, 1997.
[2] A. Edelman, T. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*. SIAM J. Matrix Analysis Applications, 20:2, pp. 303–353, 1999.
[3] C. Harris and M. Stephens, *A combined corner and edge detector*. In Proceedings of the Alvey Conference, pp. 189–192, 1988.
[4] R. Hartley, *Lines and points in three views - a unified approach*. In Proceedings of 1994 Image Understanding Workshop, pp. 1006–1016, Monterey, CA USA, 1994. OMNIPRESS.
[5] A. Heyden and K. Åström, *Algebraic properties of multilinear constraints*. Mathematical Methods in Applied Sciences, 20:13, pp. 1135–1162, 1997.
[6] Y. Ma, J. Košecká, and S. Sastry, *Optimization criteria, sensitivity and robustness of motion and structure estimation*. In Proceedings of ICCV workshop on Vision Theory and Algorithms, pp. 9–16, Corfu, Greece, 1999.
[7] Y. Ma, Stefano Soatto, J. Košecká, and S. Sastry, *Euclidean reconstruction and reprojection up to subgroups*. In Proceedings of 7th ICCV, pp. 773–80, Corfu, Greece, 1999.
[8] P. F. McLauchlan and D. W. Murry, *A unifying framework for structure and motion recovery from image sequences*. In Proceedings of 5th ICCV, pp. 314–20, Cambridge, MA, USA, 1995. IEEE Com. Soc. Press.
[9] R. M. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1993.
[10] J. Oliensis, *A multi-frame structure-from-motion algorithm under perspective projection*. International Journal of Computer Vision, 34:2-3, pp. 163–192, 1999.
[11] P. Rousseeuw and A. Leroy, *Robust Regression and Outlier Detection*. John Wiley and Sons, 1987.
[12] A. Shashua, *Trilinearity in visual recognition by alignment*. In Proceedings of ECCV, Volume I, pp. 479–484. Springer-Verlag, 1994.
[13] S. T. Smith, *Geometric Optimization Methods for Adaptive Filtering*. PhD thesis. Harvard University, Cambridge, Massachusetts, 1993.
[14] S. Soatto, R. Frezza, and P. Perona, *Motion estimation via dynamic vision*. IEEE Transactions on Automatic Control, 41:3, pp.393–413, 1996.
[15] R. Szeliski and S. B. Kang, *Recovering 3D shape and motion from image streams using non-linear least square*. Carnegie Mellon Research Report Series, 1993.
[16] C. Tomasi and T. Kanade, *Shape and motion from image streams: a factorization method*. Cornell TR 92-1270 and Carnegie Mellon CMU-CS-92-104, 1992.
[17] P. Torr and D. Murray, *The development and comparison of robust methods for estimating the fundamental matrix*. International Journal of Computer Vision, 24:3, pp. 271–300, 1997.
[18] B. Triggs, *Matching constraints and the joint image*. In Proceedings of 5th ICCV, pp. 338–43, Cambridge, MA, USA, 1995. IEEE Comput. Soc. Press.
[19] B. Triggs, *Factorization methods for projective structure and motion*. In Proceedings of CVPR, pp. 845–51, San Francisco, CA, USA, 1996. IEEE Comput. Soc. Press.
[20] T. Zhang and C. Tomasi, *Fast, robust and consistent camera motion estimation*. In Proceeding of CVPR, pp. 164–170, Alamitos, CA, USA, 1999.
[21] Z. Zhang, *A flexible new technique for camera calibration*. Microsoft Technical Report MSR-TR-98-71, Dec. 1998.
[22] Z. Zhang, *A new multistage approach to motion and structure estimation by gradually enforcing geometric constraints*. In Proceedings of the Third Asian Conference on Computer Vision, pp. 567–74, 1998.
[23] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, *A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry*. Artificial Intelligence, 78:1-2, pp. 87–119, 1995.