Multiple-View Structure and Motion From Line Correspondences

Adrien Bartoli, Peter Sturm

ICCV'03 — IN PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION,
NICE, FRANCE, OCTOBER 2003.

Multiple-View Structure and Motion From Line Correspondences

Adrien Bartoli **Peter Sturm**
INRIA Rhône-Alpes, 655, avenue de l'Europe
38334 Saint Ismier cedex, France. *First.Last@inria.fr*

Abstract

We address the problem of camera motion and structure reconstruction from line correspondences across multiple views, from initialization to final bundle adjustment. One of the main difficulties when dealing with line features is their algebraic representation.

First, we consider the triangulation problem. Based on Plücker coordinates to represent the lines, we propose a maximum likelihood algorithm, relying on linearising the Plücker constraint, and on a Plücker correction procedure to compute the closest Plücker coordinates to a given 6-vector.

Second, we consider the bundle adjustment problem. Previous overparameterizations of 3D lines induce gauge freedoms and/or internal consistency constraints. We propose the orthonormal representation, which allows handy non-linear optimization of 3D lines using the minimum 4 parameters, within an unconstrained non-linear optimizer.

We compare our algorithms to existing ones on simulated and real data.

1. Introduction

The goal of this paper is to give methods for reconstruction of line features from image correspondences over multiple views, from initialization to final bundle adjustment. Reconstruction of line features is an important topic since it is used in various areas. Bundle adjustment is the computation of an optimal visual reconstruction of camera motion and 3D scene structure, where optimal means of maximum likelihood in terms of reprojected image error. We make no assumption about the calibration of the cameras. We assume that line correspondences over at least three views are available[1].

While the multiple-view geometry of lines is well-understood, see e.g. [3, 5], there is still a need for practical structure and motion algorithms. The factorization algorithms, e.g. [7], yield reliable results but requires that all lines are visible in all views. We focus on the common three-stage approach, see e.g. [5, §17.5], consisting in (i) computing camera motion using inter-image matching tensors, (ii) triangulating the features and (iii) running bundle adjustment.

There exist reliable algorithms for step (i). In particular, it can be solved by computing trifocal tensors for triplets of consecutive images, using e.g. the automatic computation algorithm described in [5, §15.6], and registering the triplets in a manner similar to [4]. Other integrated motion estimation systems are [8], based on Kalman filtering techniques, and [11] based on registering each view in turn.

In steps (ii) and (iii), one of the main difficulties when dealing with line features arises: The algebraic representation. Indeed, there is no minimal, complete and globally non-singular parameterization of the 4-dimensional set of 3D lines, see e.g. [5, §2.2]. Hence, they are often over-parameterized. The algorithm in [5, §15.2] shows that the 'two image lines' representation is well-adapted to the computation of the trifocal tensor, while the sequential algorithm of [8] is based on Plücker coordinates.

Concerning step (ii), many of the previous works assume calibrated cameras, e.g. [6, 9, 10, 12], and use specific Euclidean representations. The linear three view algorithm of [12] and the algorithm of [10] utilize a 'closest point+direction' representation, while [9] uses the projections of the line on the $x = 0$ and the $y = 0$ planes, which has obvious singularities. These algorithms yield sub-optimal results in that none of them maximizes the individual likelihood of the reconstructed lines.

Bundle adjustment, step (iii), is a non-linear procedure involving camera and line parameters, which maximizes the likelihood of the reconstruction, corresponding to minimizing the reprojection error when the noise on measured features is assumed to have an identical and independent normal distribution. Previously-mentioned overparameterizations are not well-adapted to standard non-linear optimizers. An appropriate representation would note involve internal constraint or gauge freedom.

To summarize, there is a need for an efficient optimal triangulation algorithm, and a representation of 3D lines well-adapted to non-linear optimization. We address both of these problems. In §3, we propose triangulation methods, and in

[1] line correspondences over two views do not constrain the camera motion.

§4, we propose a non-linear representation of 3D lines that we call the *orthonormal representation*, meeting the above-mentioned efficiency constraints. Finally, §5 validates our algorithms and compares them to existing ones.

2. Preliminaries and Notations

We make no formal distinction between coordinate vectors and physical entities. Everything is represented in homogeneous coordinates. Equality up to scale is denoted by $\sim$, transposition and transposed inverse by $^\mathsf{T}$ and $^{-\mathsf{T}}$. Vectors are typeset using bold fonts ($\mathbf{L}$, $\mathbf{l}$), matrices using sans-serif fonts (S, A, R) and scalars in italics. Bars represent inhomogeneous leading parts of vectors or matrices, e.g. $\mathbf{M}^\mathsf{T} \sim \left(\bar{\mathbf{M}}^\mathsf{T} \mid m\right)$. The $\mathcal{L}_2$-norm of vector $\mathbf{v}$ is denoted $\|\mathbf{v}\|$. The identity matrix is denoted I. $SO(2)$ and $SO(3)$ denote the 2D and 3D rotation groups. The 2D Euclidean distance between point $\mathbf{q}$ and line $\mathbf{l}$, weighted by q_3, is:

$$d_\perp^2(\mathbf{q},\mathbf{l}) = (\mathbf{q}^\mathsf{T}\mathbf{l})^2/(l_1^2+l_2^2). \tag{1}$$

Given two 3D points $\mathbf{M}^\mathsf{T} \sim \left(\bar{\mathbf{M}}^\mathsf{T} \mid m\right)$ and $\mathbf{N}^\mathsf{T} \sim \left(\bar{\mathbf{N}}^\mathsf{T} \mid n\right)$, one can represent the line joining them by a homogeneous 'Plücker' 6-vector $\mathbf{L}^\mathsf{T} \sim \left(\mathbf{a}^\mathsf{T} \mid \mathbf{b}^\mathsf{T}\right)$, where $\mathbf{a} = \bar{\mathbf{M}} \times \bar{\mathbf{N}}$ and $\mathbf{b} = m\bar{\mathbf{N}} - n\bar{\mathbf{M}}$. This vector must satisfy the following *Plücker constraint* :

$$\mathcal{C}(\mathbf{L}) = 0 \text{ where } \mathcal{C}(\mathbf{L}) = \mathbf{a}^\mathsf{T}\mathbf{b}. \tag{2}$$

Given a standard (3×4) perspective projection matrix $\mathsf{P} \sim (\bar{\mathsf{P}} \mid \mathbf{p})$, a (3×6) matrix projecting Plücker line coordinates [2, 3] is given by:

$$\widetilde{\mathsf{P}} \sim (\det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\mathsf{T}} \mid [\mathbf{p}]_\times \bar{\mathsf{P}}). \tag{3}$$

As noted in [5, §15.7.2], no matter how many points are used to represent an image line $\mathbf{l}$, the quadratic error function on it can be expressed in the form $d_\perp^2(\mathbf{x},\mathbf{l}) + d_\perp^2(\mathbf{y},\mathbf{l})$ for two weighted points $\mathbf{x}$, $\mathbf{y}$ on $\mathbf{l}$. We will use this representation for simplicity. If we have 3D lines $\mathcal{S} = \{\mathbf{L}^1, \ldots, \mathbf{L}^m\}$ and cameras $\mathcal{M} = \{\mathsf{P}^1, \ldots, \mathsf{P}^n\}$, the negative log likelihood function $\mathcal{E}(\mathcal{S},\mathcal{M})$ for the reconstruction, corresponding to the reprojection error, can be written in terms of individual reprojection errors $\mathcal{E}(\mathbf{L}^j,\mathcal{M})$ for each line j:

$$\mathcal{E}(\mathcal{S},\mathcal{M}) = \sum_{j=1}^{m} \mathcal{E}(\mathbf{L}^j,\mathcal{M}) \tag{4}$$

$$\mathcal{E}(\mathbf{L}^j,\mathcal{M}) = \sum_{i=1}^{n} \left(d_\perp^2(\mathbf{x}^{ij},\mathbf{l}^{ij}) + d_\perp^2(\mathbf{y}^{ij},\mathbf{l}^{ij})\right). \tag{5}$$

3. Triangulation

This section discusses computation of structure given camera motion. We propose direct linear and iterative non-linear methods to recover Plücker line coordinates.

First, we describe a somehow trivial linear algorithm where a biased error function (compared to the reprojection error) is minimized. This algorithm is subject to the same kind of drawback as the 8 point algorithm for computing the fundamental matrix: due to possible noise in the data, the resulting 6-vectors do not generally satisfy the bilinear Plücker constraint (2), similarly to the matrix computed by the 8 point algorithm not being rank deficient [5, §10.2]. We propose what we call a *Plücker correction* procedure, which allows to compute the closest Plücker coordinates to a 6-vector.

Second, we propose an algorithm where the reprojection error of the line is minimized. The cornerstone of this algorithm is the linearization of the Plücker constraint.

Since the reconstruction of each line is independent from the others, we drop the j index in this section.

3.1. Linear Algorithm

We describe a linear algorithm, 'LIN'. In the reprojection error (5), each term is based on the square of the 2D point-to-line orthogonal distance $d_\perp$, defined by equation (1). The denominator of this distance is the cause of the non-linearity. Ignoring this denominator leads to an algebraic distance denoted by d_a, biased compared to the orthogonal distance. d_a is linear in the predicted line $\mathbf{l}$ and defined by $d_a^2(\mathbf{q},\mathbf{l}) = d_\perp^2(\mathbf{q},\mathbf{l})\, w^2 = (\mathbf{q}^\mathsf{T}\mathbf{l})^2$, where the scalar factor w encapsulates the bias as $w^2 = l_1^2 + l_2^2$:

$$(w^i)^2 = \left((\widetilde{\mathsf{P}}^i\mathbf{L})_1\right)^2 + \left((\widetilde{\mathsf{P}}^i\mathbf{L})_2\right)^2. \tag{6}$$

We define the biased linear least squares error function:

$$\mathcal{B}(\mathbf{L},\mathcal{M}) = \sum_{i=1}^{n} \left((\mathbf{x}^{i\mathsf{T}}\widetilde{\mathsf{P}}^i\mathbf{L})^2 + (\mathbf{y}^{i\mathsf{T}}\widetilde{\mathsf{P}}^i\mathbf{L})^2\right) = \|\mathsf{A}_{(2n\times 6)}\mathbf{L}\|^2 \tag{7}$$

$$\mathsf{A}^\mathsf{T} = \begin{pmatrix} \ldots & \widetilde{\mathsf{P}}^{i\mathsf{T}}\mathbf{x}^i & \widetilde{\mathsf{P}}^{i\mathsf{T}}\mathbf{y}^i & \ldots \end{pmatrix}. \tag{8}$$

Since $\mathbf{L}$ is an homogeneous vector, we add the constraint $\|\mathbf{L}\|^2 = 1$. The $\mathbf{L}$ that minimizes $\mathcal{B}(\mathbf{L},\mathcal{M})$ is then given by the singular vector of A associated to its smallest singular value. Due to noise, the recovered 6-vector does not in general satisfy the Plücker constraint (2).

3.2. Plücker Correction

Let $\mathbf{L}^\mathsf{T} \sim (\mathbf{a}^\mathsf{T} \mid \mathbf{b}^\mathsf{T})$ be a 6-vector that does not necessarily satisfy the Plücker constraint (2), i.e. $\mathbf{a}^\mathsf{T}\mathbf{b}$ might be non-zero. We seek $\widehat{\mathbf{L}}^\mathsf{T} \sim (\mathbf{u}^\mathsf{T} \mid \mathbf{v}^\mathsf{T})$, defined by $\min_{\widehat{\mathbf{L}},\mathbf{u}^\mathsf{T}\mathbf{v}=0} \|\widehat{\mathbf{L}} - \mathbf{L}\|^2$. Although this problem has a clear and concise formulation, it is *not* trivial.

We propose the following solution, summarized in a practical manner in table 1. Due to lack of space, we provide the proof of this algorithm in the extended version of this paper only. We transform the original 3D problem to an equivalent 2D problem, and solve the 2D transformed problem.

- Compute the singular value decomposition $(\mathbf{a}\ \mathbf{b}) = \bar{\mathsf{U}}\bar{\Sigma}\bar{\mathsf{V}}^{\mathsf{T}}$.
- Let $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^{\mathsf{T}}$, form matrix $\mathsf{T} = \left(\begin{smallmatrix} z_{21} & z_{22} \\ z_{12} & -z_{11} \end{smallmatrix}\right)$.
- Compute the singular vector $\widehat{\mathbf{v}}$ associated to the smallest singular value of matrix T.
- Let $\bar{\mathsf{V}} = \left(\begin{smallmatrix} \hat{v}_1 & -\hat{v}_2 \\ \hat{v}_2 & \hat{v}_1 \end{smallmatrix}\right)$, we obtain: $(\mathbf{u}\ \mathbf{v}) \sim \bar{\mathsf{U}}\,\widehat{\mathsf{V}}\,\mathrm{diag}\left(\widehat{\mathsf{V}}^{\mathsf{T}}\bar{\Sigma}\bar{\mathsf{V}}^{\mathsf{T}}\right)$.

Table 1. The *Plücker correction* algorithm. Given a 6-vector $\mathbf{L}^{\mathsf{T}} \sim (\mathbf{a}^{\mathsf{T}} \mid \mathbf{b}^{\mathsf{T}})$, this algorithm computes the closest Plücker coordinates $\widehat{\mathbf{L}}^{\mathsf{T}} \sim (\mathbf{u}^{\mathsf{T}} \mid \mathbf{v}^{\mathsf{T}})$, i.e. $\mathbf{u}^{\mathsf{T}}\mathbf{v} = 0$, in the sense of the $\mathcal{L}_2$-norm, i.e. $\|\widehat{\mathbf{L}} - \mathbf{L}\|^2$ is minimized.

3.3. Quasi-Linear Algorithms

We describe algorithms 'QLIN1' and 'QLIN2', that consider the reprojection error (5). They are based on an iterative bias-correction, through reweighting of the biased error function (7). Such algorithms are coined quasi-linear.

We showed previously that the orthogonal and the algebraic distances are related by a scalar factor, given by equation (6), depending on the 3D line. The fact that these factors depend on the unknown 3D line suggests an iterative reweighting scheme.

The first approach that comes to mind is 'QLIN1'. The linear system considered for method LIN is formed and solved. The resulting 6-vector $\mathbf{L}_0$ is corrected to be valid Plücker coordinates. This yields a biased estimate of the 3D line. Using this estimate, weight factors that contain the bias of the linear least squares error function are computed, and used to reweight the equations. The process is iterated to compute successive refined estimates $\mathbf{L}_k$ until convergence, where k is the iteration counter. Convergence is determined by thresholding the difference between two consecutive errors. It is typically reached in 3 or 4 iterations.

Experimental results show that this naive approach performs very badly, see §5. This is due to the fact that the Plücker constraint is enforced afterhand and is not taken into account while solving the linear least squares system.

To remedy to this problem, we propose 'QLIN2', that linearizes and enforces the Plücker constraint (2), as follows. The algorithm is summarized in table 2. Rewrite the constraint as $\mathcal{C}(\mathbf{L}) = \mathbf{L}^{\mathsf{T}}\mathsf{G}\mathbf{L}$ where $\mathsf{G}_{(6\times6)} = \left(\begin{smallmatrix} 0 & \mathrm{I} \\ \mathrm{I} & 0 \end{smallmatrix}\right)$. By expanding this expression to first order around the estimate $\mathbf{L}_k$, and after some minor algebraic manipulations, we obtain the following linear constraint on $\mathbf{L}_{k+1}$:

$$\mathcal{C}_k(\mathbf{L}_{k+1}) = \mathbf{L}_k^{\mathsf{T}}\mathsf{G}\mathbf{L}_{k+1} = 0.$$

We follow the constrained linear least squares optimization method summarized in [5, §A3.4.3] to enforce this linearized constraint, as well as $\|\mathbf{L}_{k+1}\| = 1$, while minimizing $\mathcal{B}(\mathbf{L}_{k+1}, \mathcal{M})$.

1. *Initialization:* Form the linear least squares system A from equation (8), compute $\mathbf{L}_0$ by minimizing $\|\mathsf{A}\mathbf{L}_0\|^2$, see §3.1, and by applying the Plücker correction procedure described in §3.2. Set $k = 0$.
2. *Constraint linearization:* Compute the SVD $\mathbf{L}_k^{\mathsf{T}}\mathsf{G} \sim \mathbf{u}^{\mathsf{T}}\mathrm{diag}(1,0,0,0,0,0)(\mathbf{v}_{(6\times1)} \mid \bar{\mathsf{V}}_{(6\times5)})^{\mathsf{T}}$.
3. *Estimation:* Compute $\min_{\boldsymbol{\gamma},\|\boldsymbol{\gamma}\|^2=1} \|\mathsf{A}\bar{\mathsf{V}}\boldsymbol{\gamma}\|^2$ and set $\mathbf{L}_{k+1} = \bar{\mathsf{V}}\boldsymbol{\gamma}$.
4. *Bias-correction:* Reweight the linear system A by computing the weights according to equation (6).
5. *Iteration:* Iterate steps 2, 3 and 4 until convergence.

Table 2. Quasi-linear triangulation algorithm 'QLIN2'.

4. Bundle Adjustment

Bundle adjustment is the non-linear minimization of the reprojection error (4), over camera and line parameters. We focus on the parameterization of 3D lines. Parameterizing the camera motion has been addressed in e.g. [1, 5, §A4.6].

4.1. Problem Statement

As said in the introduction, there are various possibilities to overparameterize the 4-dimensional set of 3D lines, which, in the context of non-linear optimization, may induce several problems.

This motivates the need for a representation of 3D lines allowing non-linear optimization with the minimum 4 parameters. In that case, there is no free scale induced by homogeneity or internal consistency constraints, and an unconstrained non-linear optimizer can be used.

4.2. The Orthonormal Representation

The orthonormal representation has been introduced in [1] for the non-linear optimization of the fundamental matrix with the minimum 7 parameters. It consists in finding a representation involving elements of $SO(n)$ and scalars (hence the term 'orthonormal representation'). In particular, no other algebraic constraints should be necessary, such as the rank-two constraint of fundamental matrices or the bilinear Plücker constraint. Using orthonormal matrices implies that the representation is well-conditioned. Based on such a representation, local update using the minimum number of parameters is possible. We derive a closed-form of the Jacobian matrix of the Plücker coordinates with respect to these parameters.

Example: representing $\mathbb{P}^1$. We derive the orthonormal representation of the 1-dimensional projective space $\mathbb{P}^1$. This is used in §4.3 to derive the orthonormal representation of 3D lines. Let $\boldsymbol{\sigma} \in \mathbb{P}^1$. Such a 2-vector is defined up to scale and has therefore only 1 degree of freedom. We represent it by an $SO(2)$ matrix W defined by:

$$\mathsf{W} = \frac{1}{\|\boldsymbol{\sigma}\|} \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}. \tag{9}$$

The first column of this matrix is $\boldsymbol{\sigma}$ itself, normalized to unit-norm. Let θ be the update parameter. A local update step is $\mathsf{W} \leftarrow \mathsf{W}\mathsf{R}(\theta)$ where $\mathsf{R}(\theta)$ is the 2D rotation matrix of angle θ. The Jacobian matrix $\frac{\partial \boldsymbol{\sigma}}{\partial \theta}$ evaluated at $\theta_0 = 0$ (the update is with respect to a base rotation) is given by:

$$\left.\frac{\partial \boldsymbol{\sigma}}{\partial \theta}\right|_{\theta_0} = \left.\frac{\partial \mathbf{w}_1}{\partial \theta}\right|_{\theta_0} = \begin{pmatrix} -\sigma_2 \\ \sigma_1 \end{pmatrix} = \mathbf{w}_2, \tag{10}$$

where $\mathbf{w}_i$ is the i-th column of W.

Updating $SO(3)$. A matrix $\mathsf{U} \in SO(3)$ can be locally updated using 3 parameters by any locally non-singular representation, such as 3 Euler angles $\boldsymbol{\theta}^\mathsf{T} = (\theta_1 \mid \theta_2 \mid \theta_3)$ as:

$$\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\boldsymbol{\theta}) \text{ with } \mathsf{R}(\boldsymbol{\theta}) = \mathsf{R}_\mathsf{x}(\theta_1)\mathsf{R}_\mathsf{y}(\theta_2)\mathsf{R}_\mathsf{z}(\theta_3), \tag{11}$$

where $\mathsf{R}_\mathsf{x}(\theta_1)$, $\mathsf{R}_\mathsf{y}(\theta_2)$ and $\mathsf{R}_\mathsf{z}(\theta_3)$ are $SO(3)$ matrices representating 3D rotations around the x-, y- and z-axis of angle θ_1, θ_2 and θ_3 respectively. The Jacobian matrix, evaluated at $\boldsymbol{\theta}_0 = \mathbf{0}_{(3\times1)}$, is given by :

$$\partial \text{vect}(\mathsf{U}) = -\begin{pmatrix} [\mathbf{l}_1]_\times & [\mathbf{l}_2]_\times & [\mathbf{l}_3]_\times \end{pmatrix}^\mathsf{T} \partial\boldsymbol{\theta}, \tag{12}$$

where $\mathbf{l}_i$ is the i-th row of U.

4.3. The Case of 3D Lines

The case of 3D lines is strongly linked with the cases of $SO(2)$ and $SO(3)$, as shown by the following result:

Any (projective) 3D line $\mathbf{L}$ can be represented by $(\mathsf{U}, \mathsf{W}) \in SO(3) \times SO(2)$, which is the orthonormal representation of the 3D line $\mathbf{L}$.

The proof of this result is obtained by showing that any 3D line has an orthonormal representation $(\mathsf{U}, \mathsf{W}) \in SO(3) \times SO(2)$, while any $(\mathsf{U}, \mathsf{W}) \in SO(3) \times SO(2)$ corresponds to a unique 3D line. The next paragraph illustrates this by means of Plücker coordinates. These results are summarized in table 3 in a practical manner.

Relating Plücker coordinates and the orthonormal representation. The orthonormal representation of a 3D line can be computed from its Plücker coordinates $\mathbf{L}^\mathsf{T} \sim (\mathbf{a}^\mathsf{T} \mid \mathbf{b}^\mathsf{T})$, by factorizing $\bar{\mathsf{C}}_{(3\times2)} \sim (\mathbf{a}\ \mathbf{b})$ as :

$$\bar{\mathsf{C}} \sim \underbrace{\begin{pmatrix} \frac{\mathbf{a}}{\|\mathbf{a}\|} & \frac{\mathbf{b}}{\|\mathbf{b}\|} & \frac{\mathbf{a}\times\mathbf{b}}{\|\mathbf{a}\times\mathbf{b}\|} \end{pmatrix}}_{SO(3)} \underbrace{\begin{pmatrix} \|\mathbf{a}\| & \\ & \|\mathbf{b}\| \end{pmatrix}}_{(\|\mathbf{a}\|\ \|\mathbf{b}\|)^\mathsf{T} \in \mathbb{P}^1}.$$

In practice, we use QR decomposition, $\bar{\mathsf{C}}_{(3\times2)} = \mathsf{U}_{(3\times3)}\Sigma_{(3\times2)}$. As already mentioned, the special form of matrix Σ is due to the Plücker constraint. While $\mathsf{U} \in SO(3)$, the two non-zero entries of Σ defined up to scale can be represented by an $SO(2)$ matrix W, as shown in §4.2.

Going back from the orthonormal representation to Plücker coordinates is trivial. The Plücker coordinates of the line are obtained from its orthonormal representation (U, W) as:

$$\mathbf{L}^\mathsf{T} \sim (w_{11}\mathbf{u}_1^\mathsf{T} \mid w_{21}\mathbf{u}_2^\mathsf{T}), \tag{13}$$

where $\mathbf{u}_i$ is the i-th column of U.

A 4-parameter update. Since $\mathsf{U} \in SO(3)$, as reviewed in §4.2, it can not be minimally parameterized but can be locally updated using equation (11), as $\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\boldsymbol{\theta})$ where $\boldsymbol{\theta} \in \mathbb{R}^3$. Matrix $\mathsf{W} \in SO(2)$ can be updated as $\mathsf{W} \leftarrow \mathsf{W}\mathsf{R}(\theta)$, where $\theta \in \mathbb{R}$. We define the update parameters by the 4-vector $\mathbf{p}^\mathsf{T} \sim (\boldsymbol{\theta}^\mathsf{T} \mid \theta)$. We denote J the (6×4)

Initialization. The initial guess is given by the Plücker coordinates $\mathbf{L}_0^\mathsf{T} \sim (\mathbf{a}_0^\mathsf{T} \mid \mathbf{b}_0^\mathsf{T})$.

- Compute the orthonormal representation $(\mathsf{U}, \mathsf{W}) \in SO(3) \times SO(2)$ of $\mathbf{L}_0$ by QR decomposition : $(\mathbf{a}_0\ \mathbf{b}_0) = \mathsf{U}\begin{pmatrix} \sigma_1 & \\ & \sigma_2 \end{pmatrix}$ and set $\mathsf{W} = \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}$.
- The 4 optimization parameters are $\mathbf{p}^\mathsf{T} = (\boldsymbol{\theta}^\mathsf{T} \mid \theta)$ where the 3-vector $\boldsymbol{\theta}$ and the scalar θ are used to update U and W respectively.

Update. (i.e. one optimization step)

- Current line is $\mathbf{L}^\mathsf{T} \sim (w_{11}\mathbf{u}_1^\mathsf{T} \mid w_{21}\mathbf{u}_2{}^\mathsf{T})$ and $\partial\mathbf{L}/\partial\mathbf{p}$ is given by equation (14).
- Compute $\mathbf{p}$ by minimizing some criterion.
- Update U and W: $\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\boldsymbol{\theta})$ and $\mathsf{W} \leftarrow \mathsf{W}\mathsf{R}(\theta)$.

Table 3. Elements for 3D line optimization through the orthonormal representation.

Jacobian matrix of the Plücker coordinates, with respect to the orthonormal representation. Matrix J must be evaluated at $\mathbf{p}_0 = \mathbf{0}_{(4\times1)}$. By using the orthonormal representation to Plücker coordinates equation (13) and equations (10,12) :

$$\mathsf{J}_{(6\times4)} = \begin{pmatrix} \mathbf{0}_{(3\times1)} & -\sigma_1\mathbf{u}_3 & \sigma_1\mathbf{u}_2 & -\sigma_2\mathbf{u}_1 \\ \sigma_2\mathbf{u}_3 & \mathbf{0}_{(3\times1)} & -\sigma_2\mathbf{u}_1 & \sigma_1\mathbf{u}_2 \end{pmatrix}. \tag{14}$$

Geometric interpretation. Each of the 4 above-defined update parameters $\mathbf{p}$ has a geometric interpretation. Since matrix W encapsulates the distance d from the origin $\mathbf{O}$ to $\mathbf{L}$, parameter θ acts on d. Matrix U is related to a 3D coordinate frame attached to $\mathbf{L}$. Parameter θ_1 rotates $\mathbf{L}$ around a circle with radius d, centered on $\mathbf{O}$, and lying on the plane defined by $\mathbf{O}$ and $\mathbf{L}$. Parameter θ_2 rotates $\mathbf{L}$ around a circle with radius d, centered on $\mathbf{O}$, and lying in a plane containing $\mathbf{O}$, the closest point $\mathbf{Q}$ of $\mathbf{L}$ to $\mathbf{O}$, and perpendicular to $\mathbf{L}$. Parameter θ_3 rotates $\mathbf{L}$ around the axis defined by $\mathbf{O}$ and $\mathbf{Q}$. For the last three cases, the angles of rotation are the parameters themselves. This interpretation allows to easily incorporate a priori knowledge while estimating a line. For example, to leave the direction of the line invariant, one may use the 2 update parameters θ_2 and θ, while to leave the distance of the line to the origin invariant, one may use the 3 update parameters $\boldsymbol{\theta}$.

5. Experimental Results

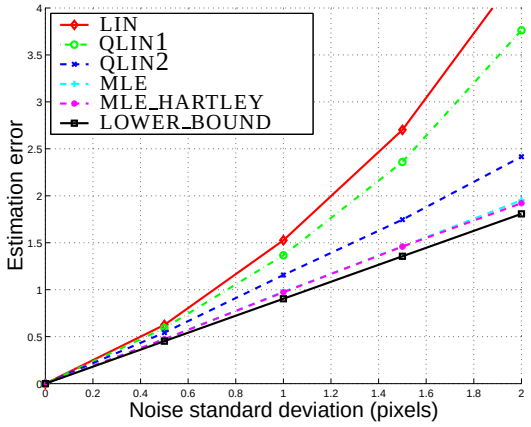
5.1. Simulated Data

Our simulated experimental setup consists of a set of cameras looking inwards at 3D lines randomly chosen in a sphere. Cameras are spread widely around the sphere. We fix the focal length of the cameras to 1000 (in number of pixels). Note that this information is not used in the rest of the experiments. The end-points of all lines are projected in all views, where their positions are corrupted by an additive Gaussian noise.

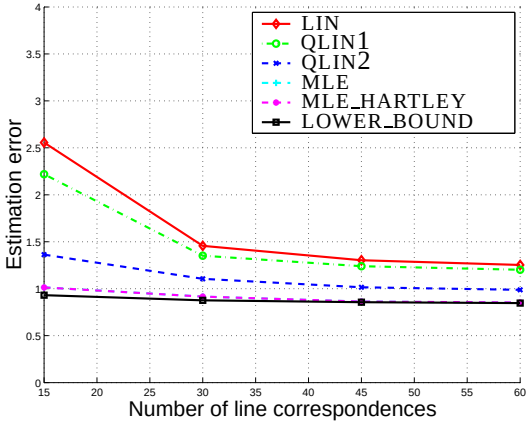
We compare the 4 methods given in this paper: LIN, QLIN1, QLIN2 and MLE (bundle adjustment based on our orthonormal representation of 3D lines), as well as the method given in [5, §15.4.1], denoted by 'MLE_HARTLEY'. This method consists in non-linearly computing the trifocal tensor as well as reconstructed lines by minimizing the reprojection error (4) and parameterizing the 3D lines by two of their three images. We also compare QLIN2 to a direct Levenberg-Marquardt-based minimization of the reprojection error: These two methods give undistinguishable results in all our experiments. Note that most existing methods, e.g. [6, 9, 10, 12] can be applied only when camera calibration is available.

We vary the added noise level from 0 to 2 pixels, while considering 20 lines and 3 views. The result is shown on figure 1 (a). One observes that, beyond 1 pixel noise, methods LIN and QLIN1 behave very badly. This is mainly due to the bias introduced by the Plücker correction procedure. Methods QLIN2, MLE and MLE_HARTLEY degrade gracefully as the noise level increases. Method QLIN2 gives reasonable results. Methods MLE and MLE_HARTLEY give undistinguishable results, very close to the theoretical lower bound.

We vary the number of lines from 15 to 60, while considering a 1 pixel noise and 3 views. The result is shown on figure 1 (b). Similar conclusions as for the previous experiment can be drawn, except for the fact, that when more than 30 lines are considered, methods LIN and QLIN1 give reasonable results. Also, methods MLE and MLE_HARTLEY give results undistinguishable from the theoretical lower bound when more than 45 lines are considered.



(a)



(b)

Figure 1. Estimation error for different methods.

Another experiment, not shown here due to lack of space, shows that when the number of views increases, the estimation error decreases for all compared methods. Note that method MLE_HARTLEY can not be used with more than three views and is therefore not concerned with these conclusions. We observe that beyond 10 views, the result of method MLE is undistinguishable from the theoretical lower bound. The results given by methods LIN and QLIN1 are reasonable when more than 15 views are considered.

The quasi-linear methods always converged within 5 iterations.

5.2. Real Data

We tested our algorithms on several image sequences. For one of them, see figure 2 (a), we show results. We pro-

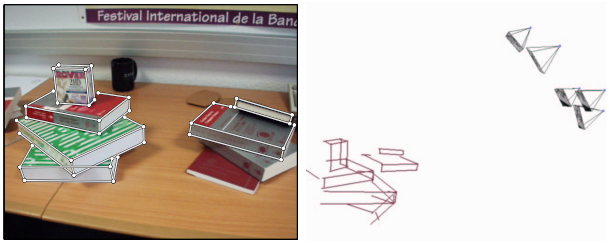
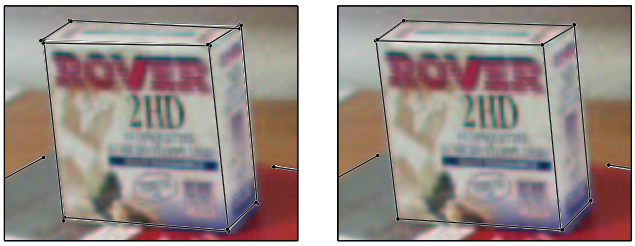


Figure 2. Sample image out of a 5-frame indoor sequence overlaid with manually-provided lines (a), and snapshot of the reconstruction given by MLE (b).

vide 45 line correspondences by hand. Note that some of them are visible in two views only. We use these line correspondences to compute the trifocal tensor corresponding to each subsequence formed by a triplet of consecutive images, using the linear method described in e.g. [5, §15.2]. We use method QLIN2 to reconstruct the lines associated with each triplet. We registered these subsequences by using the method given in [2]. At this point, we have a suboptimal guess of metric structure and motion. We further refine it using our structure from motion algorithms, to reconstruct each line by taking into account all of its images. The corresponding estimation errors are, respectively for LIN, QLIN1 and QLIN2, 2.3, 1.9 and 1.4 pixels. We used the result of QLIN2 to initialize our maximum likelihood estimator for structure and motion based on the proposed orthonormal representation together with a metric parameterization of the camera motion, which ends up with a 0.9 pixel estimation error.



QLIN2 MLE

Figure 3. Zoom on some original (white) and reprojected lines (black).

For each estimation, we reconstruct the end-points corresponding to the first view (shown on the left of figure 2). The maximum likelihood end-points are given by orthogonally projecting their images onto the image of the corresponding line. These results are visible on figure 3. Figure 2 (b) shows the cameras and lines reconstructed by MLE. There is visually no difference with the reconstruction provided by algorithm QLIN2, but that reconstructions provided by LIN and QLIN1 appear distorted.

6. Conclusion

We addressed the problem of structure and motion recovery from line correspondences across multiple views.

First, we proposed an optimal triangulation algorithm, minimizing the reprojection error, based on an iteratively reweighted least squares scheme, a linearized Plücker constraint, and a Plücker correction procedure.

Second, we proposed the orthonormal representation of 3D lines, which allows non-linear optimization with the minimal 4 parameters and analytic differentiation.

Experimental results on simulated and real data show that the linear method and its naive bias-corrected extension perform very badly. Our bias-corrected algorithm performs as well as direct Levenberg-Marquardt-based triangulation. Based on our orthonormal representation, bundle adjustment gives results close to the theoretical lower bound and undistinguishable from the three-view maximum likelihood estimator of [5, §15.4.1], while being usable with any number of views.

References

[1] A. Bartoli. On the non-linear optimization of projective motion using minimal parameters. In *ECCV*, May 2002.

[2] A. Bartoli and P. Sturm. The 3D line motion matrix and alignment of line reconstructions. In *CVPR*, December 2001.

[3] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *ICCV*, June 1995.

[4] A. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *ECCV*, June 1998.

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

[6] Y. Liu and T. Huang. A linear algorithm for motion estimation using straight line correspondences. *Computer Vision, Graphics and Image Processing*, 44(1):35–57, 1988.

[7] D. Martinec and T. Pajdla. Line reconstruction from many perspective images by factorization. In *CVPR*, June 2003.

[8] Y. Seo and K. S. Hong. Sequential reconstruction of lines in projective space. In *ICPR*, August 1996.

[9] M. Spetsakis and J. Aloimonos. Structure from motion using line correspondences. *IJCV*, 4:171–183, 1990.

[10] C. Taylor and D. Kriegman. Structure and motion from line segments in multiple images. *PAMI*, 17(11), 1995.

[11] T. Viéville, Q. Luong, and O. Faugeras. Motion of points and lines in the uncalibrated case. *IJCV*, 17(1), 1995.

[12] J. Weng, T. Huang, and N. Ahuja. Motion and structure from line correspondences: Closed-form solution, uniqueness, and optimization. *PAMI*, 14(3):318–336, 1992.