Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Vision and Image Understanding
Computer Vision and Image Understanding 100 (2005) 416–441
www.elsevier.com/locate/cviu

Structure-from-motion using lines: Representation, triangulation, and bundle adjustment

Adrien Bartoli [a,*], Peter Sturm [b]

[a] *LASMEA, 24, Avenue des Landais, 63177 Aubière cedex, France*
[b] *INRIA, 655, Avenue de l'Europe, 38334 St Ismier cedex, France*

Received 12 March 2004; accepted 17 June 2005
Available online 11 August 2005

Abstract

We address the problem of camera motion and 3D 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 3D lines, we propose a maximum likelihood algorithm, relying on linearizing the Plücker constraint and on a Plücker correction procedure, computing the closest Plücker coordinates to a given 6-vector. Second, we consider the bundle adjustment problem, which is essentially a nonlinear optimization process on camera motion and 3D line parameters. Previous overparameterizations of 3D lines induce gauge freedoms and/or internal consistency constraints. We propose the orthonormal representation, which allows handy nonlinear optimization of 3D lines using the minimum four parameters with an unconstrained optimization engine. We compare our algorithms to existing ones on simulated and real data. Results show that our triangulation algorithm outperforms standard linear and bias-corrected quasi-linear algorithms, and that bundle adjustment using our orthonormal representation yields results similar to the standard maximum likelihood trifocal tensor algorithm, while being usable for any number of views.

* Corresponding author. Fax: +33 473 407 262.
E-mail addresses: Adrien.Bartoli@gmail.com (A. Bartoli), Peter.Sturm@inria.fr (P. Sturm).

1077-3142/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2005.06.001

Keywords: Structure-from-motion; Lines; Representation; Triangulation; Bundle adjustment

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 areas such as scene modeling, augmented reality, and visual servoing. Bundle adjustment is the computation of an optimal visual reconstruction of camera motion and 3D scene structure, where optimal means 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. [5,11], there is still a need for practical structure and motion algorithms. The factorization algorithms [15,18,25] yield reliable results but requires all lines to be visible in all views. We focus on the common three-stage approach, see e.g. [11, Section 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 [11, Section 15.6], and registering the triplets in a manner similar to [6]. Other integrated motion estimation systems are [20], based on Kalman filtering techniques and [26], 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 nonsingular parameterization of the four-dimensional set of 3D lines, see, e.g. [11, Section 2.2]. Hence, they are often overparameterized, e.g., as the join of two points or as the meet of two planes (eight parameters), or by the six coefficients of their Plücker coordinates, which must satisfy the bilinear Plücker constraint. Another overparameterization is two images of the line (six parameters). The most appropriate representation depends upon the problem considered. For example, the algorithm in [11, Section 15.2] shows that the 'two image lines' representation is well-adapted to the computation of the trifocal tensor, while the sequential algorithm of [20] is based on Plücker coordinates.

Concerning step (ii), many of the previous works assume calibrated cameras, e.g. [14,21,23,27] and use specific Euclidean representations. The linear three view algorithm of [27] and the algorithm of [23] utilize a 'closest point + direction' representation, while [21] 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 nonlinear procedure involving camera and 3D line parameters, attempting to maximize the likelihood of the reconstruction, corresponding to minimizing the reprojection error when the noise on measured features has an

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

identical and independent (i.i.d.) normal distribution. Previously mentioned overparameterizations are not well-adapted to standard nonlinear optimization engines. The 'two point' and the 'two plane' overparameterizations have four degrees of internal gauge freedoms[2] which may induce numerical instabilities. The 'two image lines' parameterization has two degrees of internal gauge freedoms and implies that one may have to choose different images for different lines since all lines may not be visible in all images. Also, one must check that the chosen images are not too close to each other. Finally, direct optimization of Plücker coordinates makes sense only if a constrained optimization technique is used to enforce the bilinear Plücker constraint. An appropriate representation would not 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 nonlinear optimization. We address both of these problems through the following contributions.

In Section 3, we give an overview of various 3D line representations an their characteristics.

In Section 4, we propose triangulation methods based on using Plücker coordinates to represent the lines. A simple and optimal algorithm is obtained based on linearizing the bilinear Plücker constraint within an iteratively reweighted least squares approach.

In Section 5, we propose a nonlinear representation of 3D lines that we call the *orthonormal representation*. This representation allows efficient nonlinear optimization since only the minimum four parameters are computed at each step which allows the use of a standard unconstrained optimization engine. With this representation, there is no internal gauge freedom or consistency constraint, and analytic differentiation of the error function is possible.

Finally, Section 6 validates our algorithms and compares them to existing ones. The next section gives some preliminaries and notations and states the problem.

2. Preliminaries and notation

2.1. Notation

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 $^\top$ and $^{-\top}$. 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. $\mathbf{M}^\top \sim (\bar{\mathbf{M}}^\top | m)$. The $\mathscr{L}_2$-norm of vector v is denoted $\|\mathrm{v}\|$. The identity matrix is denoted I. $SO(2)$ and $SO(3)$ denote the 2D and 3D rotation groups.

The 2D orthogonal (Euclidean) distance between point $\mathbf{q}$ and line $\mathbf{l}$ weighted by q_3 is:

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

[2] For the former one, the position of the points along the line, and the free scale factor of the homogeneous representation of these points.

2.2. Matrix factorization

We make use of the singular value decomposition of matrices, dubbed SVD. The SVD of matrix A is $\mathsf{A}_{m\times n} = \mathsf{U}_{m\times n}\mathbf{\Sigma}_{n\times n}\mathsf{V}^{\top}_{n\times n}$, where U and V are orthonormal, and $\mathbf{\Sigma}$ is diagonal, containing the singular values of A in decreasing order. The QR factorization of matrix A is $\mathsf{A}_{m\times n} = \mathsf{Q}_{m\times m}\,\mathsf{R}_{m\times n}$, with Q orthonormal and R upper triangular. More details on these matrix factorizations can be read in, e.g. [7].

2.3. Maximum likelihood estimation

As noted in [11, Section 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 $d^2_{\perp}(\mathbf{x},\mathbf{l}) + d^2_{\perp}(\mathbf{y},\mathbf{l})$ for two weighted points $\mathbf{x}$, $\mathbf{y}$ on $\mathbf{l}$. We will use this representation for simplicity. If we have 3D lines $\mathscr{S} = \{\mathbf{L}^1,\ldots,\mathbf{L}^m\}$ and cameras $\mathscr{M} = \{\mathsf{P}^1,\ldots,\mathsf{P}^n\}$, the negative log likelihood function $\mathscr{E}(\mathscr{S},\mathscr{M})$ for the reconstruction, corresponding to the reprojection error, can be written in terms of individual reprojection errors $\mathscr{E}(\mathbf{L}^j,\mathscr{M})$ for each line j:

$$\mathscr{E}(\mathscr{S},\mathscr{M}) = \sum_{j=1}^{m} \mathscr{E}(\mathbf{L}^j,\mathscr{M}), \tag{2}$$

$$\mathscr{E}(\mathbf{L}^j,\mathscr{M}) = \sum_{i=1}^{n} (d^2_{\perp}(\mathbf{x}^{ij},\mathbf{l}^{ij}) + d^2_{\perp}(\mathbf{y}^{ij},\mathbf{l}^{ij})). \tag{3}$$

3. Representing 3D lines

We describe several representations for 3D lines in projective space and their characteristics. Some of these representations are 'partial' in the sense that they can only represent a subset of all 3D lines. For example, some work on metric reconstruction, particularly in photogrammetry, assume that the reconstructed lines do not lie at infinity. The goal of this study is to choose a representation for the triangulation and bundle adjustment problems. Concerning the triangulation, the most important criterion is that the reprojected lines is a linear function of the 3D line. Bundle adjustment is a nonlinear procedure allowing more flexibility in the choice of the parameterization. The quality of the parameterization is assessed based on criteria such as the number of internal gauge freedoms or internal constraints. A summary of the reviewed representations is finally provided. The first representation that we describe is the Plücker coordinates. We link all the other representations to Plücker coordinates.

3.1. Complete representations

3.1.1. Plücker coordinates

Given two 3D points $\mathbf{M}^{\top} \sim (\bar{\mathbf{M}}^{\top}|m)$ and $\mathbf{N}^{\top} \sim (\bar{\mathbf{N}}^{\top}|n)$, one can represent the line joining them by a homogeneous 'Plücker' 6-vector $\mathbf{L}^{\top} \sim (\mathbf{a}^{\top}\,|\,\mathbf{b}^{\top})$, see e.g. [11, Section 2.2]:

$$\begin{cases} \mathbf{a} &= \bar{\mathbf{M}} \times \bar{\mathbf{N}}, \\ \mathbf{b} &= m\bar{\mathbf{N}} - n\bar{\mathbf{M}}. \end{cases} \tag{4}$$

Other conventions for Plücker 6-vectors are also possible. Each comes with a bilinear constraint that the 6-vector must satisfy to represent valid line coordinates. For our definition, the constraint is

$$\mathscr{C}(\mathbf{L}) = 0 \quad \text{where} \quad \mathscr{C}(\mathbf{L}) = \mathbf{a}^\top \mathbf{b}. \tag{5}$$

Similarly, one can construct the Plücker coordinates of a line defined as the meet of two planes. The Plücker coordinates of a line defined as the meet of two planes $\mathbf{P}^\top \sim (\bar{\mathbf{P}}^\top | p)$ and $\mathbf{Q}^\top \sim (\bar{\mathbf{Q}}^\top | q)$ are given by:

$$\begin{cases} \mathbf{a} &= p\bar{\mathbf{Q}} - q\bar{\mathbf{P}}, \\ \mathbf{b} &= \bar{\mathbf{P}} \times \bar{\mathbf{Q}}. \end{cases} \tag{6}$$

As an example, triangulation from two views has the following closed-form solution. Let P^1 and P^2 be the two projection matrices and $\mathbf{l}^1$ and $\mathbf{l}^2$ the two imaged lines. The Plücker coordinates of the corresponding 3D line are given as the meet of the two viewing planes $\pi^i \sim \mathsf{P}^{i\top} \mathbf{l}^i$.

Given a standard (3×4) perspective projection matrix $\mathsf{P} \sim (\bar{P}|\mathbf{p})$, a (3×6) matrix projecting Plücker line coordinates [2,5] is given by

$$\tilde{\mathsf{P}} \sim (\det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\top} | [\mathbf{p}]_\times \bar{\mathsf{P}}). \tag{7}$$

It can be easily derived by expanding the expression of the 2D line joining the projections of two points:

$$\begin{aligned} \mathbf{l} &\sim \mathbf{m} \wedge \mathbf{n} \\ &\sim (\mathsf{P}\mathbf{M}) \wedge (\mathsf{P}\mathbf{N}) \\ &\sim (\bar{\mathsf{P}}\bar{\mathbf{M}} + m\mathbf{p}) \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}} + n\mathbf{p}) \\ &\sim (\bar{\mathsf{P}}\bar{\mathbf{M}}) \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}}) + m\mathbf{p} \wedge (\bar{\mathsf{P}}\bar{\mathbf{N}}) - n\mathbf{p} \wedge (\bar{\mathsf{P}}\bar{\mathbf{M}}) \\ &\sim \det(\bar{\mathsf{P}})\bar{\mathsf{P}}^{-\top}(\bar{\mathbf{M}} \wedge \bar{\mathbf{N}}) + [\mathbf{p}] \wedge \bar{\mathsf{P}}(m\bar{\mathbf{N}} - n\bar{\mathbf{M}}) \\ &\sim \tilde{\mathsf{P}}\mathbf{L}. \end{aligned}$$

Seo and Hong [20] use the Plücker coordinates representation for sequential structure-from-motion with a Kalman filtering technique. Pottmann et al. [17] use these coordinates for 3D shape reconstruction and understanding from 3D data.

3.1.2. Pair of points or pair of planes

These are two dual representations, described in details in [11, Section 2.2.2]. In the first case, the line is defined as the join of two points **M** and **N**, and in the second case, it is defined as the intersection of two planes **P** and **Q**. These representations have similar characteristics. They have eight parameters, hence four degrees of gauge freedom, the position of the points along the line (respectively, the position of the planes in the pencil of planes around the line) and the scale factors in the homogeneous coordinates of the points or the planes. For metric reconstruction, if one drops

the lines at infinity, the two point representation has six parameters. There is a direct link with Plücker coordinates using Eqs. (4) and (6). The reprojected line $\mathbf{l}$ is a bilinear function of the entries of the point or the plane coordinates. For example, for the two point representation, $\mathbf{l} \sim (\mathsf{P}\mathbf{M}) \times (\mathsf{P}\mathbf{N})$. Hartley [10] proposes a triangulation algorithm based on these representations. Habib et al. [9] use the two point representation for bundle adjustment. They consider that the line is not at infinity. The ambiguity on the position of the points along the line is fixed by constraining them to reprojected near the end-points observed in one of the images.

3.2. Partial representations

3.2.1. Closest point and direction

A 3D line is represented by its closest point to the origin, with coordinates $\mathbf{Q}^\top \sim (\bar{\mathbf{Q}}^\top 1)$, and its direction, with coordinates $\mathbf{Q}_\infty \sim (\bar{\mathbf{Q}}_\infty^\top 0)$, giving a total of six parameters. This representation does not include lines at infinity and hence cannot be used in projective space. The link with the Plücker line coordinates $\mathbf{L}$ is given by

$$\mathbf{L} \sim \begin{pmatrix} \bar{\mathbf{Q}} \times \bar{\mathbf{Q}}_\infty \\ \bar{\mathbf{Q}}_\infty \end{pmatrix}.$$

Reprojecting the line with the camera matrix $\mathsf{P} \sim (\bar{\mathsf{P}}\mathbf{p})$ is a bilinear function of the line parameters: $\mathbf{l} \sim (\bar{\mathsf{P}}\bar{\mathbf{Q}} + \mathbf{p}) \times (\bar{\mathsf{P}}\bar{\mathbf{Q}}_\infty)$. The line reconstruction algorithms proposed by Weng et al. [27] for three views and by Taylor and Kriegman [23] for multiple views use this representation. In the field of photogrammetry, Tommaselli and Lugnani [24] use this representation for bundle adjustment. Mulawa and Mikhail [16] use the additional constraint $\|\bar{\mathbf{Q}}_\infty\| = 1$.

3.2.2. Two projections

A 3D line can be represented by two projections [10,21]. This is related to the fact that reconstructing a line from two views has in general a unique solution.

Spetsakis and Aloimonos [21] use the intersection of two planes, one parallel to the plane $x = 0$, and the other one parallel to the plane $y = 0$. These two planes are formulated using four parameters a, b, c, and d by $x = az + b$ and $y = cz + d$, respectively. The pencil of points $\mathbf{Q}$ on the 3D line is parameterized by the z coordinate

$$\mathbf{Q} \sim \begin{pmatrix} az + b \\ cz + d \\ z \\ 1 \end{pmatrix}.$$

This representation has obvious singularities: lines which are parallel to the plane $z = 0$ cannot be represented. Indeed, the points lying on such lines have a constant z coordinate, and since the points are parameterized by this coordinate, one always gets the same point if the z coordinate is constant. One can link this representation to

the Plücker coordinates $\mathbf{L}$ of the line by considering any two points lying on the line, e.g. for $z = 0$ and $z = 1$, and Eq. (4), giving

$$\mathbf{L} \sim \begin{pmatrix} d \\ -b \\ bc - da \\ a \\ c \\ 1 \end{pmatrix}.$$

Ayache and Faugeras [4] use this representation for mobile robot navigation. In the field of photogrammetry, Habib [8] extends this representation by using different pairs of planes depending on the 3D line, to avoid the singularities.

Hartley [10] uses two images of the line. This representation has the following singularities: all 3D lines lying in an epipolar plane induced by the two cameras have the same images in both views. The 3D lines that cannot be uniquely represented thus form a Linear Line Complex, see e.g. [22]. Note that these singularities can be encountered in practice. The Plücker coordinates corresponding to this representation can be calculated by intersecting the two viewing planes induced by the two image lines using Eq. (6). Hartley shows that the reprojection of the line in other views in a bilinear function of the parameters.

3.2.3. The Denavit–Hartenberg parameters

The Denavit–Hartenberg representation [3] has become the standard way of representing robots and modeling their motions. The idea is to relate each joint to the next by using the minimal four parameters, namely two distances and two angles. A general 3D Euclidean transformation, between two Euclidean coordinate frames, has six degrees of freedom. For using the Denavit–Hartenberg representation, the x-axis of one coordinate frame has to be aligned with the line orthogonal to the z-axes of both coordinate frames, which cancels out two degrees of freedom, one in rotation, and one in translation. This suggests to represent a 3D line by the z-axis of a coordinate frame, and to parameterize it by the four Denavit–Hartenberg parameters with respect to a reference coordinate frame, e.g. the world coordinate frame. The Plücker coordinates corresponding to these parameters can be obtained by e.g. applying the coordinate transformation given by the four parameters to the z-axis $\mathbf{L}_z^\top \sim (0\ 0\ 0\ 0\ 0\ 1)$ of the reference frame using a 3D line rigid displacement matrix [2]. The projection equation is nonlinear in the Denavit–Hartenberg parameters since it involves products and trigonometric operators.

One problem with this parameterization is that two distances are used as parameters, which prevents from representing the lines at infinity. There is also an indeterminacy in the choice of one of the coordinate frame when the line is parallel to the z-axis of the reference coordinate frame.

Roberts [19] proposes to model 3D lines using two distances and two angles. His representation has drawbacks similar to those described above.

Note that there are other representations for modeling robots. For example, Hayati and Mirmirani [12] introduce an extra rotation parameter to the Denavit–Hartenberg representation to model the error due to near parallel axes. This representation is thus not minimal.

3.3. Summary

Table 1 summarizes the characteristics of the aforementioned representations, and of the orthonormal representation that we propose in Section 5. We observe that the only representation for which the reprojected lines is a linear function of the 3D line parameters is the Plücker coordinates. It is also seen that besides our orthonormal representation, no other complete representation allows a minimal update with four parameters, which is due to gauge freedoms and/or internal consistency constraints. Minimal update is an important criterion for using a representation within bundle adjustment.

4. Triangulation

This section discusses computation of structure given camera motion. We propose direct linear and iterative nonlinear methods to recover Plücker line coordinates. These algorithms are general in the sense that they can be used with calibrated, partially calibrated or uncalibrated cameras.

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 eight-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 (5), similarly to the matrix computed by the eight-point algorithm not being rank deficient [11, Section 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.

Table 1
Summary of different representations for 3D lines with their characteristics

Representation	Complete	Gauge freedoms	Constraints	Reprojection	Minimal update
Closest point and direction	No	1	1	Bilinear	No
Two image lines	No	2	0	Bilinear	No
Denavit–Hartenberg	No	0	0	Nonlinear	Yes
Two points or two planes	Yes	4	0	Bilinear	No
Plücker coordinates	Yes	1	1	Linear	No
Orthonormal representation	Yes	0	0	Nonlinear	Yes

The number of gauge freedoms and internal constraints are strongly linked. The 'reprojection' column is about the equation for reprojecting the 3D line with a perspective camera. The column 'minimal update' indicates if the representation can be updated with four parameters.

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

4.1. Linear algorithm

We describe a linear algorithm, 'LIN.' In the reprojection error (3), each term is based on the square of the 2D point-to-line orthogonal distance $d_\perp$, defined by Eq. (1). The denominator of this distance is the cause of the nonlinearity. Ignoring this denominator leads to an algebraic distance denoted d_a, biased compared to the orthogonal distance. It is linear in the predicted line $\mathbf{l}$ and defined by $d_a^2(\mathbf{q},\mathbf{l}) = d_\perp^2(\mathbf{q},\mathbf{l})w^2 = (\mathbf{q}^\top\mathbf{l})^2$, where the scalar factor w encapsulates the bias as $w^2 = l_1^2 + l_2^2$

$$(w^i)^2 = ((\tilde{\mathsf{P}}^i\mathbf{L})_1)^2 + ((\tilde{\mathsf{P}}^i\mathbf{L})_2)^2. \tag{8}$$

We define the biased linear least squares error function:

$$\mathscr{B}(\mathbf{L},\mathscr{M}) = \sum_{i=1}^{n}\left((\mathbf{x}^{i\top}\tilde{\mathsf{P}}^i\mathbf{L})^2 + (\mathbf{y}^{i\top}\tilde{\mathsf{P}}^i\mathbf{L})^2\right) \tag{9}$$

$$= \|\mathsf{A}_{(2n\times 6)}\mathbf{L}\|^2 \text{ with } \mathsf{A} = \begin{pmatrix} \cdots \\ \mathbf{x}^{i\top}\tilde{\mathsf{P}}^i \\ \mathbf{y}^{i\top}\tilde{\mathsf{P}}^i \\ \cdots \end{pmatrix}. \tag{10}$$

Since $\mathbf{L}$ is an homogeneous vector, we add the constraint $\|\mathbf{L}\|^2 = 1$. The $\mathbf{L}$ that minimizes $\mathscr{B}(\mathbf{L},\mathscr{M})$ is then given by the singular vector of A associated to its smallest singular value, that we compute using SVD. Due to noise, the recovered 6-vector does not in general satisfy the Plücker constraint (5).

4.2. Plücker correction

The Plücker correction procedure is analogous to the standard rank correction of the fundamental matrix based on SVD: the eight-point algorithm linearly computes a full-rank matrix F, whose smallest singular value is nullified to obtained the rank-two matrix $\hat{\mathsf{F}}$, see e.g. [11]. Matrix $\hat{\mathsf{F}}$ is the closest rank-two matrix to F, in the sense of the Frobenius norm. It is used to initialize nonlinear algorithms.

The Plücker correction procedure computes the closest Plücker coordinates to a given 6-vector, where closest is to be understood in the sense of the $\mathscr{L}_2$-norm, equivalent to the matrix Frobenius norm. It is also equivalent to the Euclidean distance between two points in $\mathbb{R}^6$. This correction is necessary to initialize the nonlinear algorithms from the solution provided by linear methods ignoring the Plücker constraint. Pottmann et al. [17] use the Euclidean distance between Plücker coordinate vectors to compare 3D lines. They underline the facts that this distance is practical for minimization purposes and is in accordance with visualization in the region of interest, i.e., near the origin.

More formally, let $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$ be a 6-vector that does not necessarily satisfy the Plücker constraint (5), i.e., $\mathbf{a}^\top\mathbf{b}$ might be nonzero. We seek $\hat{\mathbf{L}}^\top \sim (\mathbf{u}^\top|\mathbf{v}^\top)$, defined

by $\min_{\hat{\mathbf{L}},\mathbf{u}^\top\mathbf{v}=0}\|\hat{\mathbf{L}}-\mathbf{L}\|^2$. This is a linear least squares optimization problem under a nonlinear constraint. Although it has a clear and concise formulation, it is *not* trivial.

Obviously, one can modify one entry of the Plücker coordinates in accordance with the Plücker constraint, e.g. set $a_1 = -(a_2b_2 + a_3b_3)/b_1$. This simple solution has the disadvantage that the entry must be chosen depending on the actual values of the coordinates since the correction rule uses a division. Also, all entries are clearly not treated uniformly.

By comparison, our solution orthogonally projects the 6-vector on the Klein quadric and treats all its entries the same way. Kanatani [13] proposes a general iterative scheme for projecting points on nonlinear manifolds, such as projecting points in $\mathbb{R}^6$ on the Klein quadric. Our algorithm performs this projection in a noniterative manner, which thus guarantees that the optimal projected point on the Klein quadric, i.e., the optimal 3D line, is found. Its derivation is quite tricky but it can be readily implemented with few lines of code from its summary shown in Table 2.

4.2.1. A geometric interpretation

We interpret the 3-vectors **a**, **b**, **u**, and **v** as coordinates of 3D points. These points are not directly linked to the underlying 3D line. This interpretation is just intended to visualize the problem. The Plücker constraint $\mathbf{u}^\top\mathbf{v}$ corresponds to the fact that the lines induced by the origin with **u** and **v** are perpendicular. The correction criterion is the sum of squared Euclidean distances between **a** and **u** and between **b** and **v**. Hence, the problem may be formulated as finding two points **u** and **v**, as close as possible to **a** and **b**, respectively, and such that the lines induced by the origin with **u** and **v** are perpendicular. We begin by rotating the coordinate frame such that **a** and **b** are transferred on the $z = 0$ plane. This is the *reduction of the problem*. We *solve the reduced problem*, by finding two points on the $z = 0$ plane, minimizing the correction criterion and satisfying the Plücker constraint. Finally, we *express the solution* back to the original space.

4.2.2. Reducing the problem

Let us define the (3×2) matrices $\bar{\mathsf{C}} \sim (\mathbf{ab})$ and $\hat{\mathsf{C}} \sim (\mathbf{uv})$. The Plücker constraint is fulfilled if and only if the columns of matrix $\hat{\mathsf{C}}$ are orthogonal. We rewrite the correction criterion as

$$\mathscr{C} = \|\mathbf{L}-\hat{\mathbf{L}}\|^2 = \|\bar{\mathsf{C}}-\hat{\mathsf{C}}\|^2.$$

Table 2
The *Plücker correction* algorithm

- Compute the Singular Value Decomposition $(\mathbf{ab}) = \bar{\mathsf{U}}\bar{\Sigma}\bar{\mathsf{V}}^\top$.
- Let $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^\top$, form matrix $\mathsf{T} = \begin{pmatrix} z_{21} & z_{22} \\ z_{12} & -z_{11} \end{pmatrix}$.
- Compute the singular vector $\hat{\mathbf{v}}$ associated to the smallest singular value of matrix T.
- Define $\bar{\mathsf{V}} = \begin{pmatrix} \hat{v}_1 & -\hat{v}_2 \\ \hat{v}_2 & \hat{v}_1 \end{pmatrix}$ and set $(\mathbf{uv}) \sim \bar{\mathsf{U}}\hat{\mathsf{V}}\mathrm{diag}(\hat{\mathsf{V}}^\top\bar{\Sigma}\bar{\mathsf{V}}^\top)$.

Given a 6-vector $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$, this algorithm computes the closest Plücker coordinates $\hat{\mathbf{L}}^\top \sim (\mathbf{u}^\top|\mathbf{v}^\top)$, i.e., $\mathbf{u}^\top\mathbf{v} = 0$, in the sense of the $\mathscr{L}_2$-norm, i.e., $\|\hat{\mathbf{L}}-\mathbf{L}\|^2$ is minimized.

Using the following SVD $\bar{\mathsf{C}}_{(3\times2)} = \bar{\mathsf{U}}_{(3\times2)}\bar{\Sigma}_{(2\times2)}\bar{\mathsf{V}}^{\top}_{(2\times2)}$

$$\mathcal{O} = \|\bar{\mathsf{U}}\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\mathsf{C}}\|^2 = \|\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}\|^2,$$

since $\bar{\mathsf{U}}$ has orthonormal columns. We define $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^{\top}$ and $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$. Matrix $\bar{\mathsf{V}}$ is orthonormal and $\bar{\Sigma}$ is diagonal, hence the rows of $\bar{\mathsf{Z}}$ are orthogonal (i.e., $\bar{\mathsf{Z}}\bar{\mathsf{Z}}^{\top}$ is diagonal, but not $\bar{\mathsf{Z}}^{\top}\bar{\mathsf{Z}}$). Note that $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ implies $\hat{\mathsf{C}} = \bar{\mathsf{U}}\hat{\mathsf{Z}}$, even if $\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}$ is not the identity.[3] The problem is reduced to finding a column-orthogonal[4] matrix $\hat{\mathsf{Z}}$, as close as possible to the row-orthogonal matrix $\bar{\mathsf{Z}}$.

4.2.3. Solving the reduced problem

We parameterize the column-orthogonal matrix $\hat{\mathsf{Z}}$ as $\hat{\mathsf{Z}} = \hat{\mathsf{V}}\hat{\Sigma}$, where $\hat{\mathsf{V}}$ is orthonormal and $\hat{\Sigma}$ is diagonal. Hence

$$\mathcal{O} = \|\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\mathsf{V}}\hat{\Sigma}\|^2 = \|\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top} - \hat{\Sigma}\|^2.$$

The diagonal matrix $\hat{\Sigma}$ which minimizes this expression is given by the diagonal entries of $\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top}$, and does not depend on the solution for $\hat{\mathsf{V}}$. The orthonormal matrix $\hat{\mathsf{V}} = (\hat{\mathbf{v}}_1\ \hat{\mathbf{v}}_2)$ is given by minimizing the sum of squares of the off-diagonal entries of $\hat{\mathsf{V}}^{\top}\bar{\mathsf{Z}}$, with $\bar{\mathsf{Z}} = \bar{\Sigma}\bar{\mathsf{V}}^{\top} = (\mathbf{z}_1\mathbf{z}_2)$

$$\mathcal{O} = (\hat{\mathbf{v}}_1^{\top}\mathbf{z}_2)^2 + (\hat{\mathbf{v}}_2^{\top}\mathbf{z}_1)^2.$$

Define the 2D rotation matrix with angle $\pi/2$ by $\mathsf{M} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ and parameterize the orthonormal matrix $\hat{\mathsf{V}}$ by a unit vector $\hat{\mathbf{v}}$, as:

$$\begin{cases} \hat{\mathbf{v}}_1 &= \hat{\mathbf{v}}, \\ \hat{\mathbf{v}}_2 &= \mathsf{M}\hat{\mathbf{v}}, \end{cases}$$

The correction criterion can be rewritten as

$$\mathcal{O} = (\hat{\mathbf{v}}^{\top}\mathbf{z}_2)^2 + (\hat{\mathbf{v}}^{\top}\mathsf{M}^{\top}\mathbf{z}_1)^2 = \|\mathsf{T}\hat{\mathbf{v}}\|^2 \text{ with } \mathsf{T} = \begin{pmatrix} \mathbf{z}_2^{\top} \\ \mathbf{z}_1^{\top}\mathsf{M} \end{pmatrix}.$$

The unit vector $\hat{\mathbf{v}}$ minimizing this expression is given by the singular vector associated to the smallest singular value of matrix T.

4.2.4. Expressing the solution

From vector $\hat{\mathbf{v}}$ which solves the reduced problem, we form the orthonormal matrix $\hat{\mathsf{V}} = \begin{pmatrix} \hat{v}_1 & -\hat{v}_2 \\ \hat{v}_2 & \hat{v}_1 \end{pmatrix}$. The diagonal matrix $\hat{\Sigma}$ is given by $\hat{\Sigma} = \mathrm{diag}(\hat{\mathsf{V}}^{\top}\bar{\Sigma}\bar{\mathsf{V}}^{\top})$.

[3] Indeed, denote $\mathbf{u}_i$ the columns of matrix $\bar{\mathsf{U}}$ and form $\mathsf{U} = (\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_1\times\mathbf{u}_2)$. We have $\mathsf{U}^{\top}\bar{\mathsf{U}} = (\mathbf{I}_{(2\times2)}\mathbf{0}_{(2\times1)})^{\top}$. Let us multiply the correction criterion by $\mathsf{U}^{\top}$: $\mathcal{O} = \|(\bar{\mathsf{V}}\bar{\Sigma}\mathbf{0}_{(2\times1)})^{\top} - \mathsf{U}^{\top}\hat{\mathsf{C}}\|^2$. Denote $\mathsf{Y}_{(3\times2)} = \mathsf{U}^{\top}\hat{\mathsf{C}}$. The optimal solution has the form $\mathsf{Y}^{\top} = (\hat{\mathsf{Z}}^{\top}\mathbf{0}_{(2\times1)})$, since, according to the geometric interpretation, the corrected points $\mathbf{u}$ and $\mathbf{v}$ must lie on the plane defined by points $\mathbf{a}$, $\mathbf{b}$ and the origin, the plane $z = 0$. Therefore, we obtain $\hat{\mathsf{C}} = \mathsf{U}\mathsf{Y} = \bar{\mathsf{U}}\bar{\mathsf{Y}}$.

[4] The fact that matrix $\hat{\mathsf{Z}} = \bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ is column-orthogonal is induced from the Plücker constraint. Indeed, this constraint implies that $\hat{\mathsf{C}}$ is column-orthogonal, hence $\hat{\mathsf{C}}^{\top}\hat{\mathsf{C}}$ is diagonal. Matrix $\mathsf{U}^{\top}\hat{\mathsf{C}}$, where $SO(3) \ni \mathsf{U} = (\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_1\times\mathbf{u}_2) = (\bar{\mathsf{U}}\bar{\mathbf{u}})$, is also column-orthogonal. Observe that $\hat{\mathsf{C}}^{\top}\mathsf{U}\mathsf{U}^{\top}\hat{\mathsf{C}} = \hat{\mathsf{C}}^{\top}\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}} + \hat{\mathsf{C}}^{\top}\bar{\mathbf{u}}\bar{\mathbf{u}}^{\top}\hat{\mathsf{C}} = \hat{\mathsf{C}}^{\top}\bar{\mathsf{U}}\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ since $\bar{\mathbf{u}}^{\top}\hat{\mathsf{C}} = \mathbf{0}^{\top}$. Hence, matrix $\bar{\mathsf{U}}^{\top}\hat{\mathsf{C}}$ is column-orthogonal.

4.3. Quasi-linear algorithms

We describe algorithms 'QLIN1' and 'QLIN2,' that consider the reprojection error (3). They are based on an iterative bias-correction, through reweighting of the biased error function (9). Such algorithms are coined quasi-linear.

We showed previously that the orthogonal and the algebraic distances are related by a scalar factor, given by Eq. (8), depending on the 3D line. The reprojection error and the biased error functions are therefore related by a set of such factors, one for each image of the 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 three or four iterations.

Experimental results show that this naive approach performs very badly, see Section 6. 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 (5), as follows. The algorithm is summarized in Table 3. Rewrite the constraint as $\mathscr{C}(\mathbf{L}) = \mathbf{L}^\top \mathsf{G} \mathbf{L}$ where $\mathsf{G}_{(6\times 6)} = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}$. 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 $\mathbf{L}_{k+1}$:

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

We follow the constrained linear least squares optimization method summarized in [11, Section A3.4.3] to enforce this linearized constraint, as well as $\|\mathbf{L}_{k+1}\| = 1$. The idea is to find an orthonormal basis of all possible vectors satisfying the constraint and to solve for a 5-vector $\boldsymbol{\gamma}$ expressed in this basis. Such an orthonormal basis is provided by computing the nullspace of $\mathbf{L}_k^\top \mathsf{G}$ using SVD. Let $\bar{\mathsf{V}}$ be a (6×5) orthonormal matrix whose columns span the basis (i.e., $\mathbf{L}_k^\top \mathsf{G} \bar{\mathsf{V}} = \mathbf{0}$), we define $\mathbf{L}_{k+1} = \bar{\mathsf{V}} \boldsymbol{\gamma}$, hence $\mathscr{C}_k(\mathbf{L}_{k+1}) = \mathbf{L}_k^\top \mathsf{G} \bar{\mathsf{V}} \boldsymbol{\gamma} = 0$ and $\|\mathbf{L}_{k+1}\| = \|\boldsymbol{\gamma}\|$. We solve for $\boldsymbol{\gamma}$ by substituting in

Table 3
The quasi-linear algorithm 'QLIN2' for optimal triangulation

1. *Plücker correction procedure described in Section 4.2. Set* $k = 0$
2. *Constraint linearization:* Compute the singular value decomposition $\mathbf{L}_k^\top \mathsf{G} \sim \mathbf{u}^\top \mathrm{diag}(1,0,0,0,0,0)\left(\mathbf{v}_{(6\times 1)} | \bar{\mathsf{V}}_{(6\times 5)}\right)^\top$
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 Eq. (8)
5. *Iteration:* Iterate steps 2, 3, and 4 until convergence

Eq. (10) ($\|\mathsf{A}\mathbf{L}_{k+1}\|^2 = \|\mathsf{A}\bar{\mathsf{V}}\boldsymbol{\gamma}\|^2$). The singular vector associated to the smallest singular value of matrix $\mathsf{A}\bar{\mathsf{V}}$ provides the solution vector with unit $\mathscr{L}_2$-norm such that $\mathscr{B}(\mathbf{L}_{k+1}, \mathscr{M})$ is minimized.

5. Bundle adjustment

Bundle adjustment is the nonlinear minimization of the reprojection error (2), over camera and 3D line parameters. We focus on the parameterization of 3D lines. Parameterizing the camera motion has been addressed in e.g. [1,11, Section A4.6].

5.1. Problem statement

As said in Section 1, there are various possibilities to overparameterize the four-dimensional set of 3D lines. In the context of nonlinear optimization, choosing an overparameterized representation may induce the following problems. First, the computational cost of each iteration is increased by superfluous parameters. Second, artificial freedoms in the parameter set (internal gauge freedoms) are induced and may give rise to numerical instabilities. Third, some internal consistency constraints, such as the Plücker constraint, may have to be enforced.

These reasons motivate the need for a representation of 3D lines allowing nonlinear optimization with the minimum four parameters. In that case, there is no free scale induced by homogeneity or internal consistency constraints, and an unconstrained optimization engine can be used.

5.2. The orthonormal representation

The orthonormal representation has been introduced in [1] for the nonlinear optimization of the fundamental matrix with the minimum seven 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.

Commonly used nonlinear optimization engine, e.g., Newton type such as Levenberg–Marquardt, often require the Jacobian matrix of the error function with respect to the update parameters. In the orthonormal representation framework, we split it as the product of the Jacobian matrix of the error function considered with respect to the 'standard' entity representation, e.g., the fundamental matrix or Plücker coordinates, and the *orthonormal Jacobian matrix*, i.e., for the 'standard' representation with respect to the update parameters.

5.2.1. Example: representing $\mathbb{P}^1$

We derive the orthonormal representation of the one-dimensional projective space $\mathbb{P}^1$. This is used in Section 5.3 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 one degree of freedom. We represent it by an $SO(2)$ matrix W defined by

$$\mathsf{W} = \frac{1}{\|\boldsymbol{\sigma}\|} \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}. \tag{11}$$

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

$$\left.\frac{\partial \boldsymbol{\sigma}}{\partial \theta}\right|_{\theta_0} = \left.\frac{\partial \mathbf{w}_1}{\partial \theta}\right|_{\theta_0} = \begin{pmatrix} -\sigma_2 \\ \sigma_1 \end{pmatrix} = \mathbf{w}_2, \tag{12}$$

where $\mathbf{w}_i$ is the ith column of W.

5.2.2. Updating $SO(3)$

A matrix $\mathsf{U} \in SO(3)$ can be locally updated using three parameters by any well-behaved (locally nonsingular) representation, such as three Euler angles $\boldsymbol{\theta}^\top = (\theta_1 \mid \theta_2 \mid \theta_3)$ as

$$\mathsf{U} \leftarrow \mathsf{U}\mathsf{R}(\boldsymbol{\theta}) \text{ with } \mathsf{R}(\boldsymbol{\theta}) = \mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2)\mathsf{R}_z(\theta_3), \tag{13}$$

where $\mathsf{R}_x(\theta_1)$, $\mathsf{R}_y(\theta_2)$, and $\mathsf{R}_z(\theta_3)$ are $SO(3)$ matrices representing 3D rotations around the x-, y- and z-axes with angle θ_1, θ_2, and θ_3, respectively. The Jacobian matrix is derived as follows. As in the $SO(2)$ case, the update is with respect to a base rotation. The orthonormal Jacobian matrix is therefore evaluated at $\boldsymbol{\theta}_0 = \mathbf{0}_{(3\times 1)}$

$$\left.\frac{\partial \mathsf{U}}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_0} = \left(\left.\frac{\partial \mathsf{U}}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} \left|\left.\frac{\partial \mathsf{U}}{\partial \theta_2}\right|_{\boldsymbol{\theta}_0} \right|\left.\frac{\partial \mathsf{U}}{\partial \theta_3}\right|_{\boldsymbol{\theta}_0} \right).$$

After minor algebraic manipulations, we obtain

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} = \left.\frac{\partial (\mathsf{U}\mathsf{R}_x(\theta_1)\mathsf{R}_y(\theta_2)\mathsf{R}_z(\theta_3))}{\partial \theta_1}\right|_{\boldsymbol{\theta}_0} = (\mathbf{0}_3 | \mathbf{u}_3 | - \mathbf{u}_2), \tag{14}$$

where $\mathbf{u}_i$ is the ith column of U. Similarly:

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_2}\right|_{\boldsymbol{\theta}_0} = (-\mathbf{u}_3 | \mathbf{0}_3 | \mathbf{u}_1) \tag{15}$$

$$\left.\frac{\partial \mathsf{U}}{\partial \theta_3}\right|_{\boldsymbol{\theta}_0} = (\mathbf{u}_2 | - \mathbf{u}_1 | \mathbf{0}_3). \tag{16}$$

These expressions are vectorized to form the orthonormal Jacobian matrix.

5.3. 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*

$$(\mathsf{U},\mathsf{W}) \in SO(3) \times SO(2),$$

where $SO(2)$ and $SO(3)$ are the Lie groups of respectively (2×2) and (3×3) rotation matrices. (U,W) 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.

Note that this result is consistent with the fact that a 3D line has four degrees of freedom, since an element of $SO(2)$ has one degree of freedom and an element of $SO(3)$ has three degrees of freedom.

Using this representation of 3D lines, we show that there exists a locally nonsingular minimal parameterization. Therefore, 3D lines can be locally updated with the minimum four parameters. The update scheme is inspired from those given above for 2D and 3D rotation matrices, and can be plugged into most of the existing nonlinear optimization algorithms. These results are summarized in Table 4.

5.3.1. Relating Plücker coordinates and the orthonormal representation

The orthonormal representation of a 3D line can be computed from its Plücker coordinates $\mathbf{L}^\top \sim (\mathbf{a}^\top \mid \mathbf{b}^\top)$, as follows. Let $\bar{\mathsf{C}}_{(3\times 2)} \sim (\mathbf{a}|\mathbf{b})$ be factored as

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

In practice, we use QR decomposition, $\bar{\mathsf{C}}_{(3\times 2)} = \mathsf{U}_{(3\times 3)}\Sigma_{(3\times 2)}$. The special form of matrix Σ, i.e., the zero at the (1,2) entry is due to the Plücker constraint. While $\mathsf{U} \in SO(3)$, the two nonzero entries of Σ defined up to scale can be represented by an $SO(2)$ matrix W, as shown in Section 5.2.

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

Table 4
Elements for 3D line optimization using the minimal four parameters through the orthonormal representation

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

- Compute the orthonormal representation $(\mathsf{U},\mathsf{W}) \in SO(3)\times SO(2)$ of $\mathbf{L}_0$ by QR decomposition

$$(\mathbf{a}_0|\mathbf{b}_0) = \mathsf{U}\begin{pmatrix} \sigma_1 & \\ & \sigma_2 \end{pmatrix} \text{ and set } \mathsf{W} = \begin{pmatrix} \sigma_1 & -\sigma_2 \\ \sigma_2 & \sigma_1 \end{pmatrix}$$

- The four optimization parameters are $\mathbf{p}^\top = (\boldsymbol{\theta}^\top \mid \theta)$ 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 $\mathbf{L}^\top \sim (w_{11}\mathbf{u}_1^\top|w_{21}\mathbf{u}_2^\top)$ and $\partial\mathbf{L}/\partial\mathbf{p}$ is given by Eq. (18)
- Compute $\mathbf{p}$ by minimizing some criterion
- Update U and W: $\mathsf{U} \leftarrow \mathsf{U}\,\mathsf{R}(\boldsymbol{\theta})$ and $\mathsf{W} \leftarrow \mathsf{W}\,\mathsf{R}(\theta)$

$$\mathbf{L}^\top \sim (w_{11}\mathbf{u}_1^\top | w_{21}\mathbf{u}_2^\top), \tag{17}$$

where $\mathbf{u}_i$ is the ith column of U.

5.3.2. A 4-parameter update

Consider $(\mathsf{U},\mathsf{W}) \in SO(3) \times SO(2)$, the orthonormal representation of a 3D line. Since $\mathsf{U} \in SO(3)$, as reviewed in Section 5.2, it can not be minimally parameterized but can be locally updated using Eq. (13), 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 $\mathbf{p}^\top \sim (\boldsymbol{\theta}^\top \mid \theta)$.

We denote J the (6×4) Jacobian matrix of the Plücker coordinates, with respect to the orthonormal representation. Matrix J must be evaluated at $\mathbf{p}_0 = \mathbf{0}_{(4\times1)}$:

$$\mathsf{J}|_{\mathbf{p}_0} = \left(\frac{\partial \mathbf{L}}{\partial \theta_1}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta_2}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta_3}\bigg|_{\mathbf{p}_0} | \frac{\partial \mathbf{L}}{\partial \theta}\bigg|_{\mathbf{p}_0} \right).$$

By using the orthonormal representation to Plücker coordinates Eq. (17) and the Jacobian matrices for $SO(2)$ and $SO(3)$, as defined by Eqs. (12), (14)–(16), we obtain, after minor algebraic manipulations:

$$\mathsf{J}_{(6\times4)} = \begin{pmatrix} \mathbf{0}_{(3\times1)} & -\sigma_1\mathbf{u}_3 & \sigma_1\mathbf{u}_2 & -\sigma_2\mathbf{u}_1 \\ \sigma_2\mathbf{u}_3 & \mathbf{0}_{(3\times1)} & -\sigma_2\mathbf{u}_1 & \sigma_1\mathbf{u}_2 \end{pmatrix}. \tag{18}$$

5.3.3. Geometric interpretation

Each of the four above-defined update parameters $\mathbf{p}$ has a geometric interpretation. Matrix W encapsulates the ratio $\|\mathbf{a}\|/\|\mathbf{b}\|$, hence the distance d from the origin $\mathbf{O}$ to $\mathbf{L}$. Thus, 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 two update parameters θ_2 and θ, while to leave the distance of the line to the origin invariant, one may use the three update parameters $\boldsymbol{\theta}$. This allows to solve constrained line estimation cases, as summarized in the table below, indicating which update parameters to optimize in which case

Scenario	θ_1	θ_2	θ_3	θ
Fixed direction		×		×
Fixed normal to the plane formed with the origin	×			×
Fixed distance to the origin	×	×	×	

6. Experimental results

6.1. Simulated data

Our simulated experimental setup consists of a set of cameras looking inwards at 3D lines randomly chosen in a sphere with a 1 m radius. Cameras are spread widely around the sphere, at a distance of roughly 10 m away from the centre of 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 vary the parameters of this setup to assess and compare the quality of the different estimators on various scene configurations.

We compare the four 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 [11, Section 15.4.1], denoted by 'MLE_HARTLEY.' This method consists in nonlinearly computing the trifocal tensor as well as reconstructed lines by minimizing the reprojection error (2) 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, dubbed NLIN: the two methods gave undistinguishable results in all our experiments. Note that most existing methods, e.g. [14,21,23,27] can be applied only when camera calibration is available.

We measure the quality of an estimate using the *estimation error*, as described in [11, Section 4], which also provides the theoretical lower bound. The estimation error is equivalent to the value of the negative log likelihood (2) (i.e., the reprojection error).

The results are shown on graphs on Figs. 1 and 2. We observe that the different methods are always in the same order. Three distinct behaviours can be seen. Methods LIN and QLIN1 give similar results since they are subject to the same bias induced by ignoring the Plücker constraint until the final correction. Methods QLIN2 and NLIN are undistinguishable. They give better results than the biased methods. Finally, methods MLE and MLE_HARTLEY are hardly ever distinguishable. Their results are the best since they adjust the camera positions along with the 3D line parameters.

In more details, we vary the added noise level from 0 to 2 pixels, while considering 20 lines and three views on Fig. 1A. One observes that, beyond one 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 one pixel noise and three views on Fig. 1B. 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.

Fig. 2A shows the results when the number of images is varied from 3 to 12. The algorithms that do not optimize the cameras, namely LIN, QLIN1, QLIN2, and NLIN,

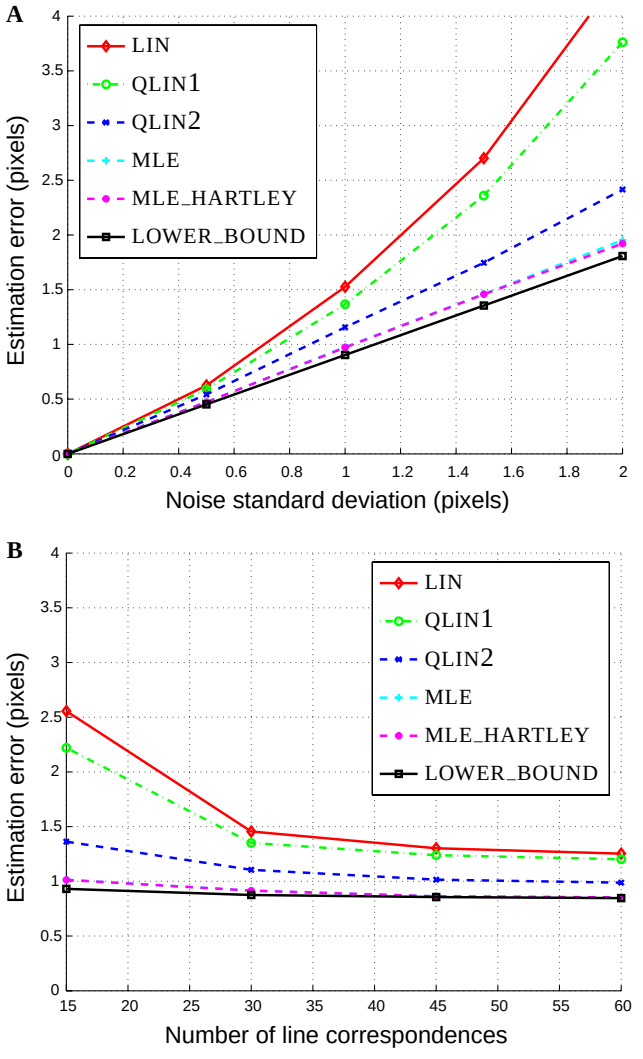


Fig. 1. Estimation error for different methods when varying the variance of added noise on image endpoints (A) and the number of lines considered (B).

have an error which increases with the number of images, whereas the bundle adjustment algorithms, namely MLE and MLE_HARTLEY, have an error which decreases. This is due to the fact that when the number of images increases, the initial camera estimation degrades, which is characteristic of the camera initialization algorithm.

When the distance between the lines and the cameras increases, Fig. 2B shows that the error decreases for all methods. This is explained by the fact that the cloud of 3D lines gets smaller and smaller in the images, which decrease the estimation error, but does not mean that the estimate is better.

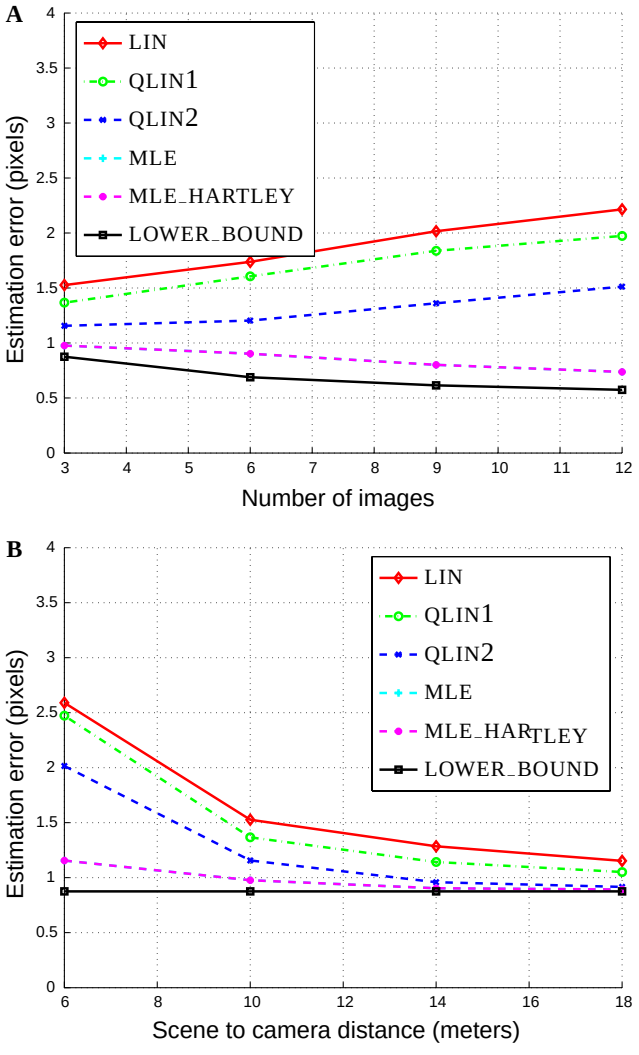


Fig. 2. Estimation error for different methods when varying the number of images (A) and the scene to camera distance (B).

We observed that the quasi-linear methods always converge within five iterations.

6.2. Real data

We tested our algorithms on several image sequences. For two of them, we show results. We compared methods LIN, QLIN1, QLIN2, and MLE, since MLE_HARTLEY is for three views only.

We observed that QLIN1 generally needs more iterations to converge than QLIN2. This is due to the Plücker correction step that significantly modifies the estimate

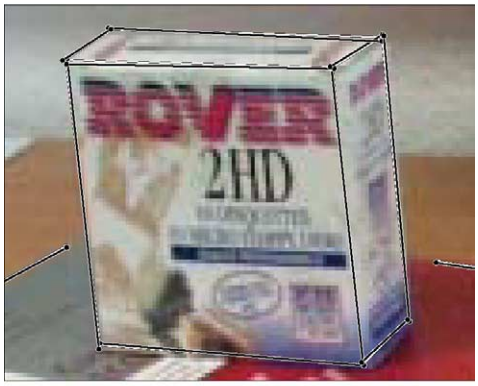
in QLIN1, while in QLIN2, since the constraint is linearized and enforced in the estimation, the correction applied to the estimate is less important.

6.2.1. The 'books' sequence

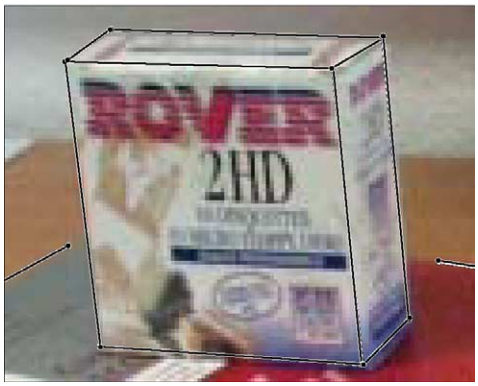
Fig. 3 shows images from this 5-frame sequence. We provided 45 line correspondences by hand. Note that some of them are visible in two views only. We used these line correspondences to compute the trifocal tensor corresponding to each subsequence formed by triplets of consecutive images, using the linear method described in e.g. [11, Section 15.2]. We used 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 had a suboptimal guess of metric structure and motion. We further refined it using our triangulation algorithms, to reconstruct each line by taking into account all of its images. The corresponding estimation errors are, respectively

Fig. 3. Sample images out of the 5-frame 'books' sequence overlaid with manually provided lines. Note that the optical distortion is not corrected.

LIN & QLIN1



QLIN2



MLE

Fig. 4. Zoom on some original (white) and reprojected lines (black) for the 'books' sequence for different methods.

for LIN, QLIN1, and QLIN2, 2.30, 2.27, and 1.43 pixels. Note the significant improvement of QLIN2 compared to the biased methods LIN and QLIN1. Methods QLIN1 and QLIN2, respectively, took four and three iterations to converge.

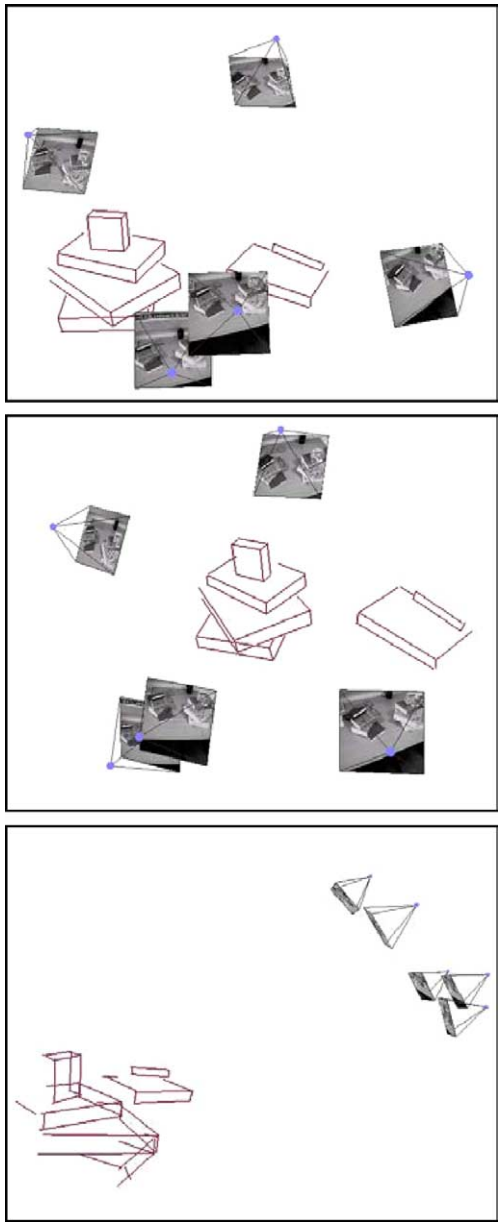


Fig. 5. Snapshots of the cameras and lines reconstructed by method MLE for the 'books' sequence. The images shown in Fig. 3 correspond to the top- and bottom-most cameras.

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.

For each estimation, we reconstructed the end-points corresponding to the first view (shown on the left of Fig. 3). The maximum likelihood end-points are given by orthogonally projecting their images onto the image of the corresponding line.

These results are visible on Fig. 4. Note the significant improvement of method MLE over methods LIN, QLIN1 and QLIN2. The lines predicted by MLE and the original lines are undistinguishable. Fig. 5 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.

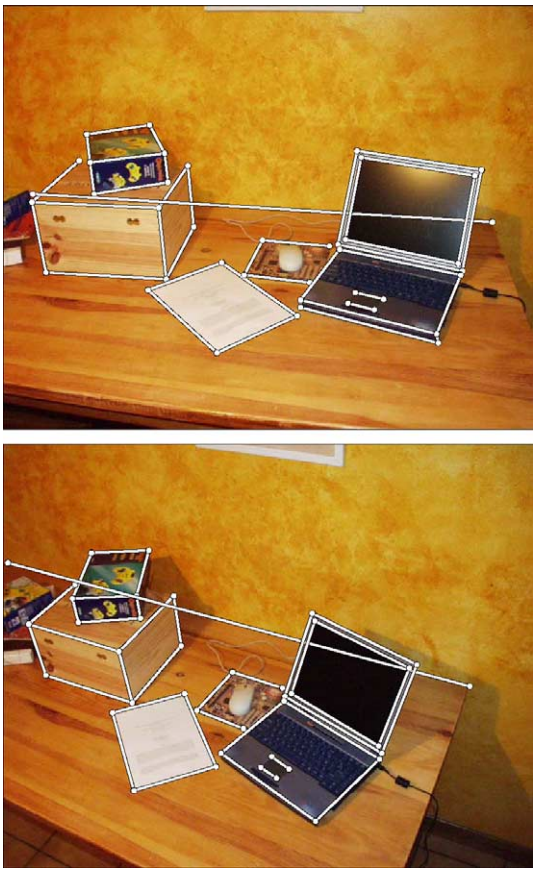


Fig. 6. Sample images out of the 8-frame 'laptop' sequence overlaid with manually provided lines. Note that the optical distortion is not corrected.

6.2.2. The 'laptop' sequence

Fig. 6 shows sample images for the 8-frame 'laptop' sequence, overlaid with the 40 manually entered line correspondences. We performed 3D reconstruction by apply-

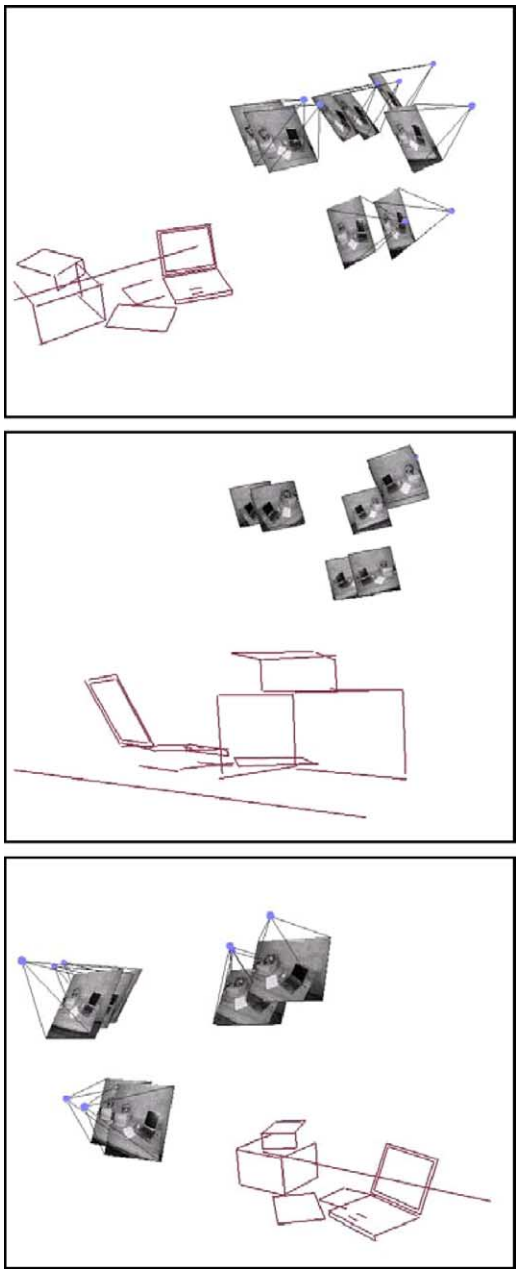


Fig. 7. Snapshots of the cameras and lines reconstructed by method MLE for the 'laptop' sequence.

ing the same algorithms as for the 'books' sequence. We obtained the following estimation errors for the triangulation algorithms, namely LIN: 1.34 pixels, QLIN1: 1.29 pixels, and QLIN2: 1.04 pixels. Methods QLIN1 and QLIN2 took respectively seven and five iterations to converge. For the bundle adjustment algorithms, we obtained an estimation error of 0.82 pixels. Fig. 7 shows snapshots of the reconstructed 3D models.

These results show that accurate reconstructed models can be obtained on real images taken by amateur digital cameras. They also show the importance of running a final bundle adjustment after initial triangulation.

7. Conclusion

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

First, we proposed an optimal triangulation algorithm. Given camera motion, the Plücker coordinates of the 3D lines are estimated by minimizing the reprojection error. The algorithm relies on an iteratively reweighted least squares scheme. We linearized the bilinear Plücker constraint to incorporate it up to first order in the estimation process. A Plücker correction procedure is proposed to find the nearest Plücker coordinates to a given 6-vector.

Second, we proposed the orthonormal representation of 3D lines, which allows nonlinear optimization with the minimal four parameters within an unconstrained optimization engine, contrarily to previously proposed overparameterizations. This representation is well-conditioned and allows analytic differentiation.

Experimental results on simulated and real data show that the standard linear method and its naive bias-corrected extension perform very badly in many cases and should only be used to initialize a nonlinear estimator. Our bias-corrected algorithm including the Plücker constraint performs as well as direct Levenberg–Marquardt-based triangulation. It is therefore a good solution to initialize subsequent bundle adjustment. 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 [11, Section 15.4.1], while being usable with any number of views.

References

[1] A. Bartoli, On the nonlinear optimization of projective motion using minimal parameters, in: Proc. 7th Eur. Conf. on Computer Vision, vol. 2, Copenhagen, Denmark, 2002, pp. 340–354.
[2] A. Bartoli, P. Sturm, The 3D line motion matrix and alignment of line reconstructions, Internat. J. Comput. Vision 57 (3) (2004).
[3] J. Denavit, R.S. Hartenberg, A kinematic notation for lower pair mechanisms based on matrices, ASME J. Appl. Mech. 22 (1955) 215–221.
[4] N. Ayache et Faugeras, Maintaining representations of the environment of a mobile robot, IEEE Trans. Robotics Autom. 5 (6) (1989) 804–819.

[5] O. Faugeras, B. Mourrain, On the geometry and algebra of the point and line correspondences between *n* images, in: Proc. 5th Internat. Conf. on Computer Vision, Cambridge, Massachusetts, USA, 1995, pp. 951–956.
[6] A.W. Fitzgibbon, A. Zisserman, Automatic camera recovery for closed or open image sequences, in: Eur. Conf. on Computer Vision, 1998, pp. 311–326.
[7] G.H. Golub, C.F. van Loan, Matrix Computation, The Johns Hopkins University Press, Baltimore, 1989.
[8] A. Habib, Motion parameter estimation by tracking stationary three-dimensional straight lines in image sequences, Internat. Arch. Photogramm. Remote Sensing 53 (1998).
[9] A. Habib, M. Morgan, Y.-R. Lee, Bundle adjustement with self-calibration using straight lines, Photogramm. Rec. (2002).
[10] R.I. Hartley, Lines and points in three views and the trifocal tensor, Internat. J. Comput. Vision 22 (2) (1997) 125–140.
[11] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, 2000.
[12] S.A. Hayati, M. Mirmirani, Improving the absolute positioning accuracy of robot manipulators, J. Robotic Syst. 2 (4) (1985) 397–441.
[13] Kanatani K, Statistical Optimisation for Geometric Computation: Theory and Practice, Elsevier Science, Amsterdam, 1996.
[14] Y. Liu, T.S. Huang, A linear algorithm for motion estimation using straight line correspondences, Comput. Vision Graph. Image Process. 44 (1) (1988) 35–57.
[15] D. Martinec, T. Pajdla. Line reconstruction from many perspective images by factorization, in: Proc. Conf. on Computer Vision and Pattern Recognition, vol. I, Madison, Wisconsin, USA, IEEE Computer Society Press, 2003, pp. 497–502.
[16] D.C. Mulawa, E.M. Mikhail, Photogrammetric treatment of linear features, Internat. Arch. Photogramm. Remote Sensing 27 (1988) 383–393.
[17] H. Pottmann, M. Hofer, B. Odehnal, J. Wallner. Line geometry for 3D shape understanding and reconstruction, in: Proc. Eur. Conf. on Computer Vision, 2004.
[18] L. Quan, T. Kanade, Affine structure from line correspondences with uncalibrated affine cameras, IEEE Trans. Pattern Anal. Mach. Intell. 19 (8) (1997) 834–845.
[19] K. Roberts, A new representation for a line, in: Proc. Conf. on Computer Vision and Pattern Recognition, San Diego, California, USA, 1988, pp. 635–640.
[20] Y. Seo, K.S. Hong, Sequential reconstruction of lines in projective space, in: Proc. 13th Internat. Conf. on Pattern Recognition, Vienna, Austria, 1996, pp. 503–507.
[21] M. Spetsakis, J. Aloimonos, Structure from motion using line correspondences, Internat. J. Comput. Vision 4 (1990) 171–183.
[22] G.P. Stein, A. Shashua, On degeneracy of linear reconstruction from three views: linear line complex and applications, IEEE Trans. Pattern Anal. Mach. Intell. 21 (3) (1999) 244–251.
[23] C.J. Taylor, D.J. Kriegman, Structure and motion from line segments in multiple images, IEEE Trans. Pattern Anal. Mach. Intell. 17 (11) (1995) 1021–1032.
[24] A. Tommaselli, J. Lugnani, An alternative mathematical model to collinearity equations using straight features, Internat. Arch. Photogramm. Remote Sensing 27 (1998) 765–774.
[25] B. Triggs, Factorization methods for projective structure and motion, in: Proc. Conf. on Computer Vision and Pattern Recognition, San Francisco, California, USA, 1996, pp. 845–851.
[26] T. Viéville, Q.T. Luong, O.D. Faugeras, Motion of points and lines in the uncalibrated case, Internat. J. Comput. Vision 17 (1) (1995).
[27] J. Weng, T.S. Huang, N. Ahuja, Motion and structure from line correspondences: closed-form solution, uniqueness, and optimization, IEEE Trans. Pattern Anal. Mach. Intell. 14 (3) (1992) 318–336.