# Egomotion using Assorted Features

Vivek Pradeep
University of Southern California
Los Angeles, California, USA
vivekpra@usc.edu

Jongwoo Lim
Honda Research Institute
Mountain View, California, USA
jlim@honda-ri.com

## Abstract

*We describe a novel and robust minimal solver for performing online visual odometry with a stereo rig. The proposed method can compute the underlying camera motion given any arbitrary, mixed combination of point and line correspondences across two stereo views. This facilitates a hybrid visual odometry pipeline that is enhanced by well-localized and reliably-tracked line features while retaining the well-known advantages of point features. Utilizing trifocal tensor geometry and quaternion representation of rotation matrices, we develop a polynomial system from which camera motion parameters can be robustly extracted in the presence of noise. We show how the more popular approach of using direct linear/subspace techniques fail in this regard and demonstrate improved performance using our formulation with extensive experiments and comparisons against the 3-point and line-sfm algorithms.*

## 1. Introduction

Real-time estimation of camera motion using only sparse sets of features from visual input is an active research topic in the computer vision and robotics communities. The number of features observed, noise-level (in feature localization as well as tracking) and their distribution, all have a major impact on the final motion estimate. Due to their abundance in natural scenes, salient corners in image data have been primarily used as interest points in most visual odometry systems. The development of novel techniques for extracting and matching these features (such as SIFT [18]) and breakthrough minimal solvers for point correspondences have led to robust and efficient visual odometry systems [21, 23]. In practical settings, however, it has been emipirically observed [5] that leveraging image lines (which might be obtained, say, from edge detection) instead of points can lead to improved performance in detection/matching (due to multipixel support), occlusion handling and dealing with

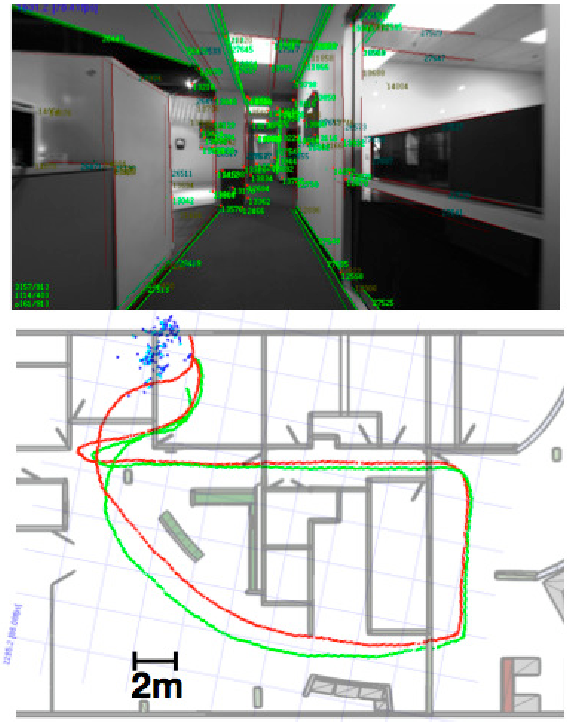


Figure 1. (Top) Lines and points tracked in our system. (Below) Estimated visual odometry superimposed on the floor-plan of an indoor office environment (Red: 3-point algorithm, green: proposed algorithm using assorted combinations of point and line features). The 3-point algorithm estimates a path that goes 'through the wall' towards the end and incorrectly estimates the camera position beyond the bounds of the room. For details, please refer to the rest of the paper and accompanying video.

T-junctions. Furthermore, the abundance of edge features in man-made environments (cityscapes and indoor structures) can be exploited to reduce tracking failures significantly, thereby minimizing situations where odometry systems can get 'lost' and also help to reconstruct high-level scene information. On the other hand, it is also well-known that the constraints imposed by line correspondences on camera pose are much weaker than those provided by points [10] and there is considerable ambiguity when dealing with curved surfaces like cylindrical columns. In [24], an interesting exposition of the complementary noise statistics and failure modes of line-based and point-based feature trackers is provided and a robust tracker is built by fusing both systems.

Given these conditions, it might be desirable to have a visual odometry algorithm that can incorporate any combination of point and line features as available in the image data and yield a camera motion estimate using the combination set that generates the most accurate solution (see figure 1). For a real-time and robust implementation, it is preferable to have a unified framework that, independent of feature type, computes the 6dof motion from minimal sets over the available data. This paper presents a novel minimal solver that accomplishes this and so is ideal for use in such a hypothesize-and-test setting [8].

Since this work is motivated by an application for autonomous navigation, the visual input and underlying geometry of our algorithm stem from a calibrated stereo pair. This simplifies the task of motion recovery and facilitates scale observability. Given any set of matches containing at least *three points* or *two lines* or *two points and one line* across two stereo views, our algorithm can compute the underlying camera motion using the same solver. This approach is much more elegant than simply integrating the state-of-art line-based and point-based systems and enables the evaluation of associated costs in a unified RANSAC setting. Using a pair of calibrated trifocal tensors, we form a low-degree polynomial system of equations that enforces the orthonormality constraint by representing rotations by unit quaternions. This is a different algebraic approach from point-based minimal solvers of [20], where the orthonormality constraints are explictly enforced. The quaternion representation has a significant impact on the noise performace of our algorithm and is an interesting result when taking into account the well-documented problem of recovering consistent camera motion from noisy trifocal tensors. The key contributions of this paper are:

- A novel quaternion-based geometrical formulation and polynomial solver for any combination of point or line feature correspondences over two stereo views for robustly computing camera motion.
- Extensive experiments using synthetic and real data demonstrating the usefulness of using both point and line features in visual odometry as opposed to a single feature type.

## 2. Related Work

The three-point method [9, 21] is currently the most popular algorithm for performing visual odometry (from feature points) with stereo cameras. Since the polynomial constraint for deriving the motion parameters is set up using the triangle law of cosines, this approach works only for a configuration of three points in general position and is therefore used in a RANSAC [8] framework for establishing support. Other methods, which address monocular schemes too, solve polynomial equation systems that are established from geometrical constraints robustified by enforcing algebraic conditions (like rotation matrix orthonormality). Several flavors of such algorithms [3, 20, 29] exist and these vary in mathematical structure, assumptions on available camera calibration and even in the minimal number of correspondences required for a feasible solution. Research in this area has also resulted in the development of numerically efficient and stable methods for solving the corresponding polynomial systems and popular techniques include Groebner basis [27], polynomial eigen-value problem (PEP) [15] and the hidden-variable [16]. While all this points to significant research effort in the direction of developing minimal solver based systems for point feature based odometry, not much has been done in developing robust, real-time formulations for lines and even less has been done for hybrid systems.

Traditionally, line features have been employed in structure from motion algorithms using the multifocal tensor framework [2, 17]. The trifocal tensor is a $3\times3\times3$ cube operator that expresses the (projective) geometric constraints between three views independent of scene structure. However, it can be computed given at least 13 line or 7 point correspondences. In general, the trifocal tensor has 27 parameters (26 upto a scale), but only 18 degrees of freedom (upto projective ambiguity). The remaining 9 constraints must be enforced to obtain a consistent solution. [28] introduces a cubic polynomial system for extracting the tensor from 6 points only and a method to uncover the underlying camera matrices is presented in [10]. The latter also introduces, for the first time, a closed-form linear solution from a combination of point or line features. For a calibrated setting, these matrices can be decomposed to obtain the camera orientation and perform visual odometry. The four-view extension of this concept, called the quadrifocal tensor, was investigated in [26]. From a purely geometrical standpoint, the work described in [6] is the most similar to our approach (albeit it does not address the problem of a minimal solver for mixed combinations of features), as it exploits the known orientation between the stereo pair in the quadrifocal tensor to enforce constraints between image intensities of adjacent stereo pairs. It is evident from all these works, however, that enforcing non-linear dependencies within the tensor indices requires substantial book-keeping [11, 13] and is ultimately too cumbersome for estimating a 6-dof motion. Another approach, which is algebraically similar in methodology to our work and that of [20] for point-feature based odometry is described in [4], which constructs a small low-degree polynomial system and explicitly enforces orthonormality constraints.

A unified representation for points, lines and planes by repesenting linear features as Gaussian density functions was presented in [25]. However, as mentioned by the authors themselves, this representation falls short of being directly applicable for a motion-and-structure recovery framework due to unresolved issues in defining join operations and performing inverse projections. Some attempt has been made towards integrating point and line features for the perspective pose estimation problem [1, 7]. A factorization based multi-frame SfM algorithm also utilizing point and line features is presented in [22]. The proposed method, however, is restricted to cases where translation is very small and involves iterative recomputations of rotation and translation parameters. To the best of our knowledge, therefore, no unified, closed-form formulation for dealing with point and line features exists and for multifocal approaches, there is room for improvement in terms of noise performance.

## 3. Using Assorted Features

**Notation.** Unless otherwise stated, a 3D point in space is represented by a homogeneous 4-vector $\mathbf{X} = [\mathrm{X}_1\ \mathrm{X}_2\ \mathrm{X}_3\ \mathrm{X}_4]^\top \in \mathbb{R}^4$ and its projection on the image plane of camera $i$ by a homogeneous 3-vector $\mathbf{x^i} = \left[x_1^i\ x_2^i\ x_3^i\right]^\top \in \mathbb{R}^3$. Similarly, any line and its projection on the image plane are denoted by the parameters $\mathbf{L} = [\mathrm{L}_1\ \mathrm{L}_2\ \mathrm{L}_3\ \mathrm{L}_4]^\top$ and $\mathbf{l^i} = \left[l_1^i\ l_2^i\ l_3^i\right]^\top$ respectively. A projective camera is given by a $3\times4$ matrix $\mathbf{K}\,[\mathbf{R} \quad \mathbf{t}]$, with $\mathbf{K}$ being the $3\times3$ internal calibration matrix and the $3\times3$ rotation $\mathbf{R}$ and $3\times1$ translation vector $\mathbf{t}$ representing the exterior camera orientation. In the remainder of this paper, without loss of generality, we assume normalized camera coordinates and therefore, $\mathbf{K}$ is set to be the identity matrix $\mathbf{I}$ . The trifocal tensor $\mathcal{T} = \{\mathbf{T}_\mathrm{i}\}_{\mathrm{i}=1,2,3}$, with each $3\times3$ submatrix denoted by $\mathbf{T}_\mathrm{i}$ is the geometric object of interest in this work. The $[...]_\times$ notation denotes the skew-symmetric matrix for forming a vector cross-product and the colon operator ':' is used to reference rows or columns of matrices (for instance, $\mathbf{A}(:, i)$ is the $i^{th}$ column, $\mathbf{A}(:, [i : j, k])$ references columns $i$ to $j$ and column $k$) .

### 3.1. Review of the trifocal relations

Consider a canonical three camera configuration given by $\mathbf{P_1} = [\mathbf{I} \quad \mathbf{0}]$ , $\mathbf{P_2} = [\mathbf{A} \quad \mathbf{a}]$ and $\mathbf{P_3} = [\mathbf{B} \quad \mathbf{b}]$. As described in [12], the trifocal tensor for these three views, $\mathcal{T} = \{\mathbf{T}_\mathrm{i}\}$ is given by

$$\mathbf{T_i} = \mathbf{A^i}\mathbf{b}^\top - \mathbf{a}\mathbf{B^i}^\top \tag{1}$$

$\mathbf{A^i} = \mathbf{A}(:, i), \mathbf{B^i} = \mathbf{B}(:, i)$. Let us assume that we have determined a line correspondence $\mathbf{l^1} \Leftrightarrow \mathbf{l^2} \Leftrightarrow \mathbf{l^3}$ and a point correspondence $\mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^3}$ across $\mathbf{P_1}, \mathbf{P_2}, \mathbf{P_3}$ respectively. The relationships between these features and the tensor can be expressed by the following equations:

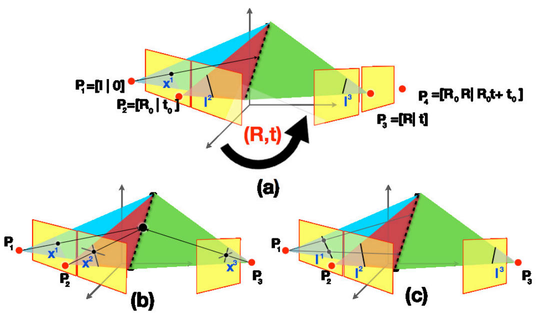


Figure 2. (a) Stereo geometry for two views and the point-line-line configuration (b) point-point-point and (c) line-line-line configuration in terms of point-line-line configuration

$$l_i^1 = \mathbf{l^2}^\top \mathbf{T_i} \mathbf{l^3} \tag{2}$$

$$\left[\mathbf{x^2}\right]_\times \left(\sum_i x_i^1 \mathbf{T_i}\right) \left[\mathbf{x^3}\right]_\times = \mathbf{0}_{3\times3}, \tag{3}$$

In the general case, a single point triplet generates four and a line-line-line correspondence provides two linearly independent constaints over the tensor parameters. Geometrically, both these configurations are best understood in terms of a hypothetical point-line-line correspondence $\mathbf{x^1} \Leftrightarrow \mathbf{l^2} \Leftrightarrow \mathbf{l^3}$ (see figure 2(a)). The constraint expressed by this arrangement is simply the incidence relationship between $\mathbf{x^1}$ and the line transferred from $\mathbf{l^3}$ via the homography induced by $\mathbf{l^2}$. For a line triplet therefore, each of any two points (figure 2(c)) on the line in the first image establishes a point-line-line configuration giving two equations, while for a case of three matching points figure2(b)), we have four possible permutations obtained by choosing any two linearly independent lines spanned by the points in the $\mathbf{P_2}$ and $\mathbf{P_3}$.

### 3.2. Calibrated tensors and linear solution

We represent a binocular stereo rig (see figure 2(a)) in its canonical form, with the left and right camera matrices $\mathbf{P_1} = [\mathbf{I} \quad \mathbf{0}]$ and $\mathbf{P_2} = [\mathbf{R_0} \quad \mathbf{t_0}]$ respectively. Here, $(\mathbf{R_0}$,$\mathbf{t_0})$ encodes the rigid geometry of the rig, and is fixed and known a-priori. After undergoing arbitrary rotation $\mathbf{R}$ and translation $\mathbf{t}$ , the corresponding cameras in the same coordinate system can be written as:

$$\begin{aligned} \mathbf{P}_3 &= [\mathbf{R} \quad \mathbf{t}] & (4) \\ \mathbf{P}_4 &= [\mathbf{R_0}\mathbf{R} \quad \mathbf{R_0}\mathbf{t} + \mathbf{t}_0] & (5) \end{aligned}$$

These forms can be recursively applied to every pair of stereo views before and after motion, and the goal of our visual odometry algorithm is to estimate $(\mathbf{R}$,$\mathbf{t})$. These representations can be simplified further by taking into consideration that for a rectified stereo pair (without loss of generality, as any stereo pair can be rectified), $\mathbf{R_0} = \mathbf{I}$

and $\mathbf{t_0} = \begin{bmatrix} t_x & 0 & 0 \end{bmatrix}^\top$, where $t_x$ is given by the baseline. The geometric framework of our algorithm is composed of two trifocal tensors; $\mathcal{T}^{\mathrm{L}} = \{\mathbf{T}^{\mathbf{L}}_{\mathbf{i}}\}$ arising out of image correspondences between cameras $\mathbf{P_1}$,$\mathbf{P_2}$,$\mathbf{P_3}$ and $\mathcal{T}^{\mathrm{R}} = \{\mathbf{T}^{\mathbf{R}}_{\mathbf{i}}\}$ from correspondences between $\mathbf{P_1}$,$\mathbf{P_2}$,$\mathbf{P_4}$. These tensors, using equations [1-5], are given as

$$\mathbf{T}^{\mathbf{L}}_{\mathbf{i}} = \mathbf{R}^{\mathbf{i}}_{\mathbf{0}}\mathbf{t}^\top - \mathbf{t_0}\mathbf{R}^{\mathbf{i}\top} \quad (6)$$
$$\mathbf{T}^{\mathbf{R}}_{\mathbf{i}} = \mathbf{R}^{\mathbf{i}}_{\mathbf{0}}\left(\mathbf{R_0}\mathbf{t} + \mathbf{t_0}\right)^\top - \mathbf{t_0}\left(\mathbf{R_0}\mathbf{R}\right)^{i\top} \quad (7)$$

Since the stereo configuration is fixed and known, it is only required to estimate the twelve parameters of the underlying motion to fix $\mathcal{T}^{\mathbf{L}}$ and $\mathcal{T}^{\mathbf{R}}$. From correspondence sets of the form $\{\mathbf{l^1} \Leftrightarrow \mathbf{l^2} \Leftrightarrow \mathbf{l^3}; \mathbf{l^1} \Leftrightarrow \mathbf{l^2} \Leftrightarrow \mathbf{l^4}\}$ or $\{\mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^3}; \mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^4}\}$ in equations (2) and (3), one can write a concatenated linear system in terms of the twelve unknowns. Furthermore, an image line can be parameterized by taking any two arbitary points lying on it. We can thus form

$$\mathbf{A}\mathbf{y} = \mathbf{0} \quad (8)$$

$$\mathbf{y} = [r_{11}\ r_{21}\ r_{31}\ r_{12}\ r_{22}\ r_{32}\ r_{13}\ r_{23}\ r_{33}\ t_1\ t_2\ t_3\ 1]^\top \quad (9)$$

where $r_{\mathrm{ij}}$ is the $(i,j)^{th}$ element of the rotation matrix $\mathbf{R}$ and translation $\mathbf{t} = \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^\top$. Note that this linear system will be inhomogeneous due to the form of equation (7). We now geometrically derive the minimum number of feature correspondences (over the four views) required to solve equation (8). Let us consider $\mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^3}$ first. With $\mathbf{P_1}$ and $\mathbf{P_2}$ fixed, a corresponding 3D point $\mathbf{X}$ is defined from the first two views. Thus, $\mathbf{x^3} = \mathbf{P_3}\mathbf{X}$ provides only two linearly independent equations for the unknown $(\mathbf{R}, \mathbf{t})$. Theoretically, therefore, $\{\mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^3}; \mathbf{x^1} \Leftrightarrow \mathbf{x^2} \Leftrightarrow \mathbf{x^4}\}$ must generate 4 linearly independent equations. However, $\mathbf{P_3}$ and $\mathbf{P_4}$ form a stereo pair and the following holds for any rectified stereo pair $i$ and $j$:

$$x_3^i x_2^j - x_2^i x_3^j = 0 \quad (10)$$

Thus, on concatenating point correspondence constraints from $\mathcal{T}^{\mathbf{L}}$ and $\mathcal{T}^{\mathbf{R}}$, only 3 linearly independent equations are obtained. Arguing similarly but noting that equation (10) is not invoked for general points on matching lines, it can be shown that a line quartet provides 4 linearly independent equations. These dependencies can also be seen by performing row operations on $\mathbf{A}$ matrix. Hence, given $n$ point- and $m$ line-correpondence sets, matrix $\mathbf{A}$ has $3n+4m$ independent rows. A linear solution, therefore, can be obtained for {4 points} or {3 lines} or {3 points+1 line} (overconstrained) or {2 points+2 lines} (overconstrained). In the presence of noise, it is recommended to use more than the minimum number of equations per correspondence. However, with noisy features, this approach is also not recommended because $\mathbf{R}$ will not be obtained as an orthonormal rotation matrix. One could refine the solution further by minimizing the Frobenius norm with respect to an orthonormal matrix, but better techniques are presented in sections 3.3 and 3.4.

### 3.3. Subspace solution

This approach is similar to the works described in [20, 15, 16]. Using nonlinear techniques, it is possible solve for $\mathbf{y}$ in (8) from a combination of only {3 points} or {2 points + 1 line} or {2 lines + 1 point}, each of which provides 9, 10 or 11 linearly independent equations. A {2 line} solution can also be obtained (8 equations), but is unstable in the presence of noise. We therefore ignore this condition and solve nonlinearly given at least 3 correspondences (the {3 lines} case has a linear solution). With this formulation, the least number of constraints we might expect is 9 equations from {3 points} (for 12 unknowns), and so, the solution can be given by a (12-9 =) 3 dimensional subspace. However, as the system of equations is non-homogeneous, we write,

$$\mathbf{y} = \mathbf{y_p} + \alpha\mathbf{y_1} + \beta\mathbf{y_2} + \gamma\mathbf{y_3} \quad (11)$$

$\mathbf{y_p}$ is the so-called 'particular solution' of a non-homogeneous linear system that can be computed from the psuedo-inverse of $\mathbf{A}$ and $\mathbf{y_1}$, $\mathbf{y_2}$ and $\mathbf{y_3}$ are the eigenvectors corresponding to the smallest eigenvalues from a singular value decomposition on $\mathbf{A}$. To solve for $\alpha$, $\beta$ and $\gamma$ and simultaneously ensure orthonormality for the rotation part the six polynomial constraints expressing unit norm for each row of $\mathbf{R}$ and orthogonality between any two rows can be applied. The resulting polynomial system of equations may then be solved by a variety of techniques such as Groebner basis [14] or polynomial eigenvalue problem (PEP) [16].
While better than the linear approach, the subspace of $\mathbf{A}$ is still very unstable in the presence of noise. This instability can be traced back to the underlying noisy tensor constraints that do not faithfully encode a camera representing rigid motion. This problem of robustly extracting camera matrices from the trifocal tensor has been reported elsewhere in [10] and a work-around is described that requires estimating epipoles and applying additional constraints on the camera matrices. We describe a method in section 3.4, our main contribution, that yields a much more robust solution without requiring any additional information or imposing further constraints. In section 4, we provide further experimential evidence that justifies why the next method might be preferred over the subspace solution.

### 3.4. Quaternion-based direct solution

A rotation matrix $\mathbf{R}$ can be parameterized in terms of the unit quaternion $\mathbf{q}$= $a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$:

$$\mathbf{R} = \begin{bmatrix} a^2+b^2-c^2-d^2 & 2(bc-ad) & 2(ac+bd) \\ 2(ad+bc) & a^2-b^2+c^2-d^2 & 2(cd-ab) \\ 2(bd-ac) & 2(ab+cd) & a^2-b^2-c^2+d^2 \end{bmatrix}$$

Rewriting equation (8) using this parameterization, we get the following system of equations:

$$\mathbf{A}'\mathbf{y}' = \mathbf{0} \tag{12}$$

$$\mathbf{y}' = \left[\begin{array}{c} a^2\ b^2\ c^2\ d^2\ ab\ ac\ ad\ bc\ bd\ cd\ t_1\ t_2\ t_3\ 1 \end{array}\right]^\top$$

We will first solve for the quaternion parameters. To this end, using the equations in (12), $(t_1, t_2, t_3)$ are expressed in terms of the other parameters by performing the following steps:

$$Solve\ \mathbf{A}'(:,[11:13])\mathbf{C_t} = \mathbf{A}'(:,[1:10,14]) \tag{13}$$

$$\mathbf{A_t} = \mathbf{A}'(:,[1:10,14]) - \mathbf{A}'(:,[11:13])\mathbf{C_t} \tag{14}$$

$$\mathbf{A_t}\mathbf{y_t} = \mathbf{0} \tag{15}$$

$$\mathbf{y_t} = \left[\begin{array}{c} a^2\ b^2\ c^2\ d^2\ ab\ ac\ ad\ bc\ bd\ cd\ 1 \end{array}\right]^\top$$

The translation part is given by $\left[\begin{array}{ccc} t_1 & t_2 & t_3 \end{array}\right]^\top = -\mathbf{C_t}\mathbf{y_t}$ and we now have to only solve the polynomial system (15). At this point, we also introduce the quaternion constraint $a^2 + b^2 + c^2 + d^2 = 1$ to ensure that in the presence of noise, a consistent rotation is obtained. To solve for $\mathbf{y_t}$, we adopt the method of elimination by writing the monomials $\left(b^2, c^2, d^2, bc, bd, cd\right)$ in terms of $\left(a^2, ab, ac, ad, 1\right)$ :

$$\begin{bmatrix} b^2 \\ c^2 \\ d^2 \\ bc \\ bd \\ cd \end{bmatrix} = \mathbf{C_q} \begin{bmatrix} a^2 \\ ab \\ ac \\ ad \\ 1 \end{bmatrix} \tag{16}$$

where,

$$\mathbf{A_t}(:,[2:4,8:10])\mathbf{C_q} = \mathbf{A_t}(:,[1,5:7,11]) \tag{17}$$

There must be sufficient correspondences so that equation (17) is not underconstrained. $\mathbf{A_t}(:,[2:4,8:10])$ submatrix must have at least rank 6, implying atleast 6 linearly independent rows. Since one independent constraint is already provided by quaternion unit norm condition, 5 or more independent rows must come from the point or line correspondences in $\mathbf{A}'_\mathbf{t}$. However, we note that the manipulation in (14) introduces dependencies in $\mathbf{A_t}$ and therefore, minimal configuration sets are {3 points} or {2 lines} or {2 points + 1 line}. The {2 lines} configuration tends to be less stable in the presence of noise, and therefore, we employ {3 lines} and include {2 lines + 1 point} sets. From a systems point of view, this also lets use a uniform set size of 3 features for any combination in a RANSAC setting. The minimal 3 feature criterion ensures that it will be always possible to solve equations (13) and (17). In the presence of noise, it is recommended to use more than just the linearly independent rows provided by each correspondence. Now, the equations in (16) can be rewritten as

$$\begin{bmatrix} b^2 \\ c^2 \\ d^2 \\ bc \\ bd \\ cd \end{bmatrix} = \begin{bmatrix} [1] & [1] & [1] & [2] \\ [1] & [1] & [1] & [2] \\ [1] & [1] & [1] & [2] \\ [1] & [1] & [1] & [2] \\ [1] & [1] & [1] & [2] \\ [1] & [1] & [1] & [2] \end{bmatrix} \begin{bmatrix} b \\ c \\ d \\ 1 \end{bmatrix} \tag{18}$$

$[i]$ represents an $i^{th}$ degree polynomial in $a$. We note that not all terms in the LHS of equation (18) are independent and they should infact, satisfy the following conditions:

$$\begin{array}{ll} (bc)^2 = (b^2)(c^2) & (bc)(bd) = (b^2)(cd) \\ (bd)^2 = (b^2)(d^2) & (bc)(cd) = (c^2)(bd) \\ (cd)^2 = (c^2)(d^2) & (bd)(cd) = (d^2)(bc) \end{array}$$

Applying these to the RHS in (18), we obtain the following final system of constraints:

$$\begin{bmatrix} [3] & [3] & [3] & [4] \\ [3] & [3] & [3] & [4] \\ [3] & [3] & [3] & [4] \\ [3] & [3] & [3] & [4] \\ [3] & [3] & [3] & [4] \\ [3] & [3] & [3] & [4] \end{bmatrix} \begin{bmatrix} b \\ c \\ d \\ 1 \end{bmatrix} = \mathbf{0} \tag{19}$$

Since by hypothesis, there exists a solution to $\left[\begin{array}{cccc} b & c & d & 1 \end{array}\right]^\top$, any 4×4 submatrix in (19) must have a determinant equal to 0. This gives a 13-degree polynomial in $a$.

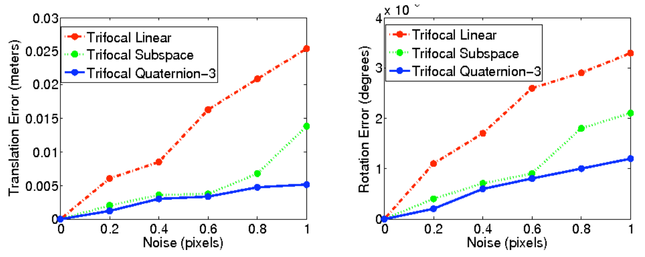
$$(k_1a^{10} + k_2a^8 + k_3a^6 + k_4a^4 + k_5a^2 + k_6)a^3 = 0 \tag{20}$$

Let $\alpha = a^2$ :

$$k_1\alpha^5 + k_2\alpha^4 + k_3\alpha^3 + k_4\alpha^2 + k_5\alpha + k_6 = 0 \tag{21}$$

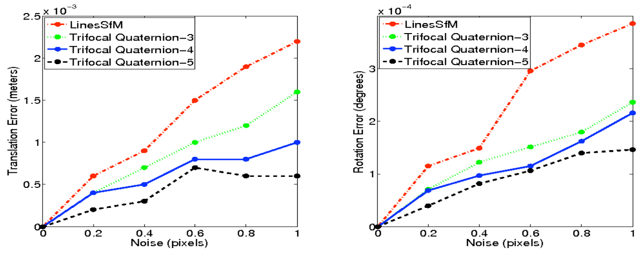
This 5-degree polynomial can be easily solved and the real-values $a$'s retained. Thus, the rotation matrix $\mathbf{R}$ can be composed after $(b, c, d)$ are obtained from the null-vector of the 4×4 submatrix and translation $\left[\begin{array}{ccc} t_1 & t_2 & t_3 \end{array}\right]^\top = -\mathbf{C_t}\mathbf{y_t}$.

The multiple solutions are not a problem as the motion parameters with the largest support will be picked in a RANSAC setting. Compared to the standard 3-point algorithm, one can employ more than just the minimal number of feature correspondences to obtain better motion hypotheses in the presence of noise (using more features, as we will see, outperforms the 3-point algorithm). This can potentially lead to more number of trials due to an increased probability of adding an outlier feature in the minimal set. However, in practice, we found that larger feature sets generate better hypotheses and this usually offsets any negative influences of the chance outliers (see Section 4). Furthermore, once the best solution and corresponding inliers have been found, they can all be plugged into equation (12) to recompute a more consistent solution.



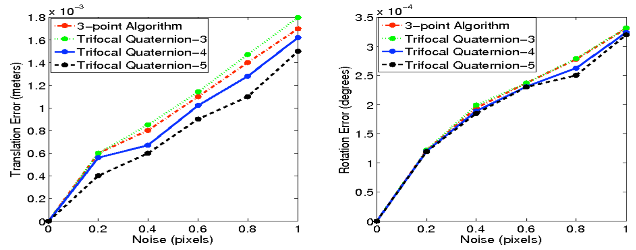
(a) Translation Error (b) Rotation Error

Figure 3. Rotation and translation errors comparing performance of linear (red) subspace (green) and quaternion (blue) based formulations of the trifocal minimal solver using arbitrary feature combinations.



(a) Translation Error (b) Rotation Error

Figure 5. Rotation and translation errors comparing performance of the lines Sfm algorithm (red) against trifocal-quaternion solver using 3 (green), 4 (blue) and 5 (back) lines.



(a) Translation Error (b) Rotation Error

Figure 4. Rotation and translation errors comparing performance of the 3-point algorithm (red) against trifocal-quaternion solver using 3 (green), 4 (blue) and 5 (black) points.

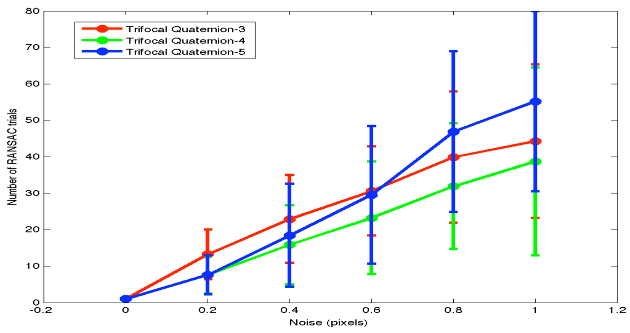


Figure 6. Number of RANSAC trials before an acceptable solution was found for trifocal quaternion-3 (red), -4(green) and -5(blue).

## 4. Experiments

In this section, we report experimental results for synthetic and real data. Synthetic data experiments are performed to compare the results obtained using the different versions of our trifocal formulations - trifocal linear (Section 3.2), trifocal subspace (Section 3.3) and trifocal quaternion - $n$ (Section 3.4). The $n$ suffix indicates the number of feature correspondences used in the solver (with $n = 3$ being the minimal case). We also perform comparisons against the popularly used three point algorithm [9] and the line-only based motion estimation setup described in [4], which we refer to in the experiments as 'LinesSfM'. We believe that a mechanism to incorporate more than just the minimal set in a closed-form for generating a single hypothesis is one of the advantages of our method, when compared to these algorithms. This is demonstrated in the various experiments below by setting $n = 4$ or 5. Since the development of a mixed-feature solver was motivated by practical difficulties in sparsely textured or badly illuminated indoor environments, we also performed some real experiments that exhibit such pathological behavior to evaluate our algorithms against state of the art.

### 4.1. Synthetic Data

We quantify the performance of the different algorithms discussed in the paper across various noise levels. Camera geometry is created by fixing the baseline to 7.5 cm (similar to one of our actual stereo cameras) and generating a random stereo pair. The second stereo pair is displaced with a random motion. 3D scene geometry and image correspondences are also created by randomly generating 3D points and lines in space, and projecting them onto the four cameras. Zero mean, Gaussian noise with varying standard deviations is added to the coordinates of image lines and points in each view. Furthermore, similar to [4, 20], we use only the lower quartile of error distributions for all algorithms as the targeted use is in a RANSAC framework where finding a fraction of good hypotheses is more important than consistency. All the experiments are based on 1000 trials.

Figure 3 shows the rotation and translation errors for the various flavors of our trifocal formulation, given different minimal sets. Random permutations of points and lines were generated for each trial. As alluded to earlier, the linear approach has the worst performance, while the quaternion-based polynomial system works the best. We next compare performance using only points against the standard three-point algorithm (figure 4) and using only lines (figure 5) against LinesSfm. Since the latter algorithm works best only for small motions, we generated random motion such that it does not break this method. For the points-only case, the quaternion-3 method shows

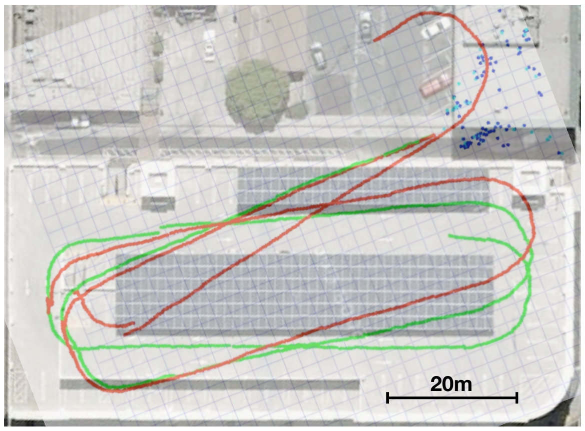


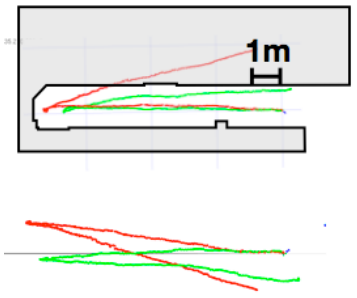
Figure 7. Sequence for two loops around a parking structure (red: 3 point algorithm, green: trifocal quaternion-5)

slightly inferior error behavior compared to the three point algorithm, but the quaternion-4 and quaternion-5 methods outperform. Using only lines, all versions of the trifocal method do significantly better. Since we are proposing a scheme that can incorporate more than the minimal set, an evaluation of its impact on the number of RANSAC trials is presented in figure 6. For this test, a random set of 30 points and 15 lines was generated and RANSAC trials by selecting random combinations of features were carried out until a threshold reprojection error (in all four images) was reached. ANOVA tests between the three groups at each noise level found no significant difference, but these results might differ for real data. Note that, for the sake of fairness, we only use the particular class of features the benchmark methods have been designed to work in figures 4 and 5, though we argue that incorporating mixed feature combinations adds an additional dimension to the odometry pipeline with potential for greater accuracy. It makes more sense to demonstrate this behavior with real data, as line and point features have different error statistics with significant impact on performance, and are difficult to replicate in simulated experiments. We do so in the next section.

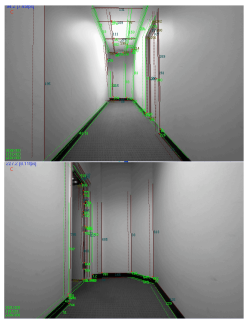
### 4.2. Real Data[1]

We have implemented a visual odometry system that uses the standard 3-point algorithm and the proposed trifocal-quaternion algorithm. It takes a stereo video stream as input, performs stereo rectification, point and lines feature detection and tracking, and compues the motion using the tracked features. Here, we briefly describe some of the implementation details. For point features, we follow the steps outlined in for KLT feature tracking [19]. For line features, short edgels are formed from the edge maps after non-maximal suppression, and then the edgels that are close to one another and lie on a same line are linked together to make a longer line. Line tracking is done by tracking individual edges in a line, then taking votes for the candidate lines in the previous frame. For feature detection and tracking, NVidia's CUDA framework is used for faster processing speed. The current system runs at ~9 fps on a laptop (Intel Core2Duo 2.5 GHz and NVidia GeForce 8600M GT) with both line and point features, and ~20 fps with only point features on the same machine.

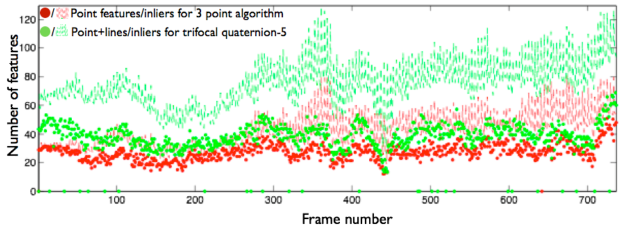
[1]Videos are available at http://www.vivekpradeep.alturl.com and tracking data can be downloaded from http://vision.ucsd.edu/~jwlim/



(a) Estimated camera paths (red : 3-point algorithm, green: trifocal quaternion-5). (Top) Top-down view and (Bottom) sideways view.



(b) Low textured corridor images with very few point features.



(c) Plot of number of features per frame and RANSAC inliers

Figure 8. Performance in low-textured, corridor sequence

We compare performance in two very challenging environments. Given our simulation data, we contrast trifocal quaternion-5 against the 3-point algorithm. In figure 7, the sequence is collected around a parking structure. This is a large, open space and several frames have either very little texture or point features are clustered on one side. The 3-point algorithm performs noticably worse, exhibiting a severe break (see near top of figure) and veering off from the loop. About 2036 frames were captured for this sequence. In figure 8(a), we show the result in a corridor sequence, where lack of sufficient texture in some frames (figure 8(b)), leads to very few points being tracked. The actual motion simply consists of a walk along with corridor, entrance into an elevator and a return. As is obvious from the estimated trajectory, the use of points and lines in the trifocal quaternion algorithm leads to much better performance. The 3-point algorithm drifts significantly into the wall. The bottom plot in 8(a) shows a sideways view of the results. Since the stereo camera was fixed and transported on a platform, the motion was more or less confined along a plane parallel to the ground. Ideally, the estimated camera path should lie on the thin horizontal line from this perspective. The 3-point algorithm begins to drift from this very quickly, while the effect is minimized for the trifocal quaternion-5 algorithm. In figure 8(c), a plot of the number of features available to both algorithms and RANSAC inliers found per frame is shown. This clearly shows the advantage of exploiting both feature types in such situations. Please refer to the accompanying video for further details.

## 5. Conclusion and Future Work

We have presented a robust algorithm for estimating camera motion for a stereo pair using combinations of point and line features. As shown by our experiments, using such assorted feature sets leads to better motion estimates than state of the art visual odometry algorithms in real world scenarios. This helps in handling low-textured regions, where the conventional method of point-based odometry will fail. The quaternion based solution yields significantly better motion estimates than the conventional linear/subspace methods in the presence of noise and provides a robust mechanism for extracting camera matrices from noisy trifocal tensors. Furthermore, our formulation allows one to use more than the minimal set in a RANSAC setting, leading to significantly better results with no severe penalties on real-time performance. To show the quality and stability of our proposed algorithm, all the presented results have been generated without any bundle adjustment or inter-frame non-linear refinement. Visual odometry, however, can be substantially improved by applying such techniques and more robust feature tracking like SIFT. We plan to carry out further validation and implement aforementioned methods in an extended version of this paper.

## References

[1] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *IEEE PAMI*, 25(5):578–589, 2003.

[2] A. Bartoli and P. Sturm. Multiple-view structure and motion from line correspondences. *ICCV*, 2003.

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the p4p problem for camera with unknown focal length. *CVPR*, pages 1–8, 2008.

[4] M. Chandraker, J. Lim, and D. J. Kreigman. Moving in stereo: Efficient structure and motion using lines. *ICCV*, 2009.

[5] S. Christy and R. Horaud. Iterative pose computation from line correspondences. *CVIU*, 73(1):137–144, 1999.

[6] A. Comport, E. Malis, and P. Rives. Accurate quadrifocal tracking for robust 3d visual odometry. *ICRA*, pages 40–45, 2007.

[7] F. Dornaika and C. Garcia. Pose estimation using point and line correspondences. *Real-Time Imaging*, 5(3):215–230, 1999.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *IJCV*, 22(2):125–140, 1997.

[9] R. Haralick, C. Lee, K. Ottenberg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. *CVPR*, 1991.

[10] R. Hartley. Lines and points in three views and the trifocal tensor. *IJCV*, 22(2):125–140, March 1997.

[11] R. Hartley. Computation of the trifocal tensor. *ECCV*, pages 20–35, 1998.

[12] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[13] A. Heyden. Geometry and algebra of multiple projective transformations. *PhD thesis, Lund University*, 1995.

[14] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. *ECCV*, pages 302–315, 2008.

[15] Z. Kukelova, M. Bujnak, and T. Pajdla. Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems. *BMVC*, 2008.

[16] H. Li and R. Hartley. Five-point motion estimation made easy. *ICPR*, 1, 2006.

[17] Y. Liu and T. Huang. A linear algorithm for motion estimation using straight line correspondences. *ICPR*, pages 213–219, 1988.

[18] D. Lowe. Object recognition from local scale-invariant features. *ICCV*, pages 1150–1157, 1999.

[19] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *IJCAI*, pages 1151–1156, 1981.

[20] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE PAMI*, 26(6):756–770, 2004.

[21] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. *CVPR*, 1:652–659, 2004.

[22] J. Oliensis and M. Werman. Structure from motion using points, lines, and intensities. *CVPR*, 2, 2000.

[23] M. Pollefeys, D. Nister, and et al. Detailed real-time urban 3d reconstruction from video. *IJCV*, 2007.

[24] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. *ICCV*, 2:1508–1515, October 2005.

[25] S. Seitz and P. Anandan. Implicit representation and scene reconstruction from probability density functions. *CVPR*, 2, 1999.

[26] A. Shashua and L. Wolf. On the structure and properties of the quadrifocal tensor. *Lecture notes in computer science*, pages 710–724, 2000.

[27] H. Stewénius, C. Engels, and D. Nister. Recent developments on direct relative orientation. *J. Photogrammetry and Remote Sensing*, 60:284–294, 2006.

[28] P. H. S. Torr and A. Zisserman. Robust parameterization and computation of the trifocal tensor. *Image and Vision Computing*, 15:591–605, 1997.

[29] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. *ICCV*, pages 278–284, 1999.