Generalised Linear Pose Estimation

Andreas Ess, Alexander Neubeck, Luc Van Gool
Computer Vision Lab
ETH Zurich
8092 Zurich, Switzerland
{aess, aneubeck, vangool}@vision.ee.ethz.ch
http://www.vision.ee.ethz.ch

Abstract

This paper investigates several aspects of 3D-2D camera pose estimation, aimed at robot navigation in poorly-textured scenes. The major contribution is a fast, linear algorithm for the general case with six or more points. We show how to specialise this to work with only four or five points, which is of utmost importance in a test and hypothesis framework. Our formulation allows for an easy inclusion of lines, as well as the handling of other camera geometries, such as stereo rigs. We also treat the special case of planar motion, a valid restriction for most indoor environments. We conclude the paper with extensive simulated tests and a real test case, which substantiate the algorithm's usability for our application domain.

1 Introduction

Given known points in the world and their projected images, pose estimation refers to recovering the pose of the camera from such data. In photogrammetry, this problem has been studied as early as 1841 by Grunert [8]. For a recent survey, consult e.g. [9].

We are interested in pose estimation for the purpose of navigating a robot with a stereo rig through a poorly-textured room. Besides the usual demands on speed and accuracy, a pose estimation algorithm should use as few feature points as possible during robust estimation in such a setting. To ease the problem of lacking feature points, it is advantageous to allow other features besides points, such as lines, in the estimation. Moreover, the algorithm should be extendable to a stereo rig. Lastly, as the robot movements might be restricted to planar ones, it should be possible to include this constraint in the formulation. All these demands are fulfilled by our algorithm: four points suffice for the estimation in the general case, we show how lines can be used in conjunction with points, we investigate the special case of planar motion, and demonstrate the application of the algorithm to a stereo rig in this planar case. While the presented results are for a robotic scenario, the algorithm is also directly applicable to e.g. augmented reality.

Before presenting the actual algorithm in Section 2, the next subsection lists some existing solutions to the general problem of linear pose estimation. Section 3 investigates the specialisation of the algorithm for the planar case. Then, Section 4 gives results from both simulated and real-world scenarios, before the paper is concluded in Section 5.

BMVC 2007 doi:10.5244/C.21.22

1.1 Related Work

A linearisation of the pose estimation problem is not only tempting for reasons of efficiency. An accurate linear solution also provides a valuable starting point for an iterative algorithm, resulting in its faster convergence. First, we discuss three prior contributions to linearising the n-point pose estimation problem.

The first linear approach for determining the camera pose was proposed by Fiore [6]. Based on a projective invariant, he constructs a linear equation system to solve for the depths of the image points. Using the recovered depths, 3D-3D pose estimation (also called absolute orientation, [10]) is carried out to obtain the actual pose. In the general case, the method needs at least six point correspondences to work.

Based on a general procedure for linearising quadratic systems, Ansar and Daniilidis [1] suggest methods for pose estimation from points and lines. For points, their approach is based on depth recovery by considering ratios between the points' depths. For lines, their respective angles and the invariance of inner and outer products are used to directly estimate the pose. In both cases, two linear systems are constructed to guarantee a unique solution. While mathematically interesting, this approach mainly suffers from high-dimensional equation systems, which considerably hamper the performance of the algorithm, both in terms of speed and numerical accuracy.

Quan and Lan [15] also base their algorithm for depth recovery on ratios between different points' depths. They apply their method to each point in turn and solve a set of fourth degree polynomial constraints. Their algorithm works for $n \geq 4$ points.

All the above methods need an additional step of absolute orientation. The algorithm proposed in this paper recovers the pose directly. It works for both points ($n \geq 4$) and lines ($n \geq 5$), and a mixed solution is possible. As shown in the results section, is performance is comparable with or better than other algorithms.

As a note on minimal solutions, Nister [12] presents a generalised solution for the 3-point case (an additional point is needed for resolving a 4-fold ambiguity). Besides minimality, an advantage of the method is its applicability to stereo rigs or other generalised camera systems by allowing non-concurrent rays. The algorithm presented here offers the same generality, but also allows more points or the usage of lines.

Phong *et al.* [14] also present an algorithm for calculating the pose from points and lines, however, their solution is iterative. Liu *et al.* [11] study a two-step process for pose estimation from lines that first iteratively solves for rotation and then linearly for translation. Point tuplets can be included as lines. They also present a linear solution for the first step, which however is not minimal in the number of needed correspondences, as opposed to the one presented here. Also, no special cases (planar motion, multiple cameras) are studied.

2 Pose from Points

Given a set of world points $\mathbf{p}_i$ and a corresponding set of *normalised*[1] image points $\mathbf{q}_i$, the pose estimation problem consists of finding a rotation matrix $\mathtt{R}$ and a translation vector $\mathbf{t}$ that minimise the geometric reprojection error:

$$\sum_i \|\mathtt{R}\mathbf{p}_i + \mathbf{t} - \lambda_i \mathbf{q}_i\| / \lambda_i \rightarrow min. \tag{1}$$

[1] For actual points $\tilde{\mathbf{q}}$ in the image, $\mathbf{q} = \mathtt{K}^{-1}\tilde{\mathbf{q}}$, for lines $\tilde{\mathbf{n}}$, $\mathbf{n} = \mathtt{K}^{\top}\tilde{\mathbf{n}}$, with $\mathtt{K}$ the internal calibration matrix.

In Eq. (1), λ_i models the depth of a point from the given view. Since this is a non-linear problem in λ_i, iterative algorithms are needed for the solution. However, instead of considering the geometric reprojection error, one could minimise the distance between the two back-projected rays, also known as object space collinearity:

$$\sum_i \|\mathtt{R}\mathbf{p}_i + \mathbf{t} - \lambda_i \mathbf{q_i}\| \rightarrow min. \tag{2}$$

While this formulation does not take the depth-dependent uncertainty of the 3D location into account, it allows for a linear, and hence fast and simple, solution to the problem. By setting the derivatives of λ_i to zero, the λ_i can be eliminated. Let $\mathtt{Q}_i = \mathtt{I} - \frac{\mathbf{q_i}\mathbf{q_i}^\top}{\|\mathbf{q_i}\|}$ denote the projection operator into $\mathbf{q_i}$. After some rearranging, Eq. (2) can be reformulated as

$$\sum_i \|\mathtt{Q}_i(\mathtt{R}\mathbf{p}_i + \mathbf{t})\| \rightarrow min. \tag{3}$$

Assuming an affine transformation instead of R, Eq. (3) becomes a linear system $\mathtt{M}\mathbf{x} = \mathbf{0}$ with twelve unknowns whose least squares solution is wanted. This can be solved in a straightforward manner. However, as the values of R are between -1 and 1, while the ones for $\mathbf{t}$ can take any value, $\mathbf{t}$'s variables can dominate the solution vector, resulting in unsatisfactory precision for R. It is not possible to impose the hard-constraints of orthonormality on R in a linear fashion due to their quadratic nature. Still, the less restrictive constraint $\|\mathtt{R}\|_F = \sqrt{3}$, where $\|\cdot\|_F$ is the Frobenius norm, can be enforced, as shown in the following. The above problem is partitioned as follows:

$$(\mathtt{A}\ \mathtt{B}) \begin{pmatrix} \mathbf{r} \\ \mathbf{t} \end{pmatrix} = \mathbf{0}, \tag{4}$$

where $\mathbf{r}$ corresponds to the nine variables of the rotation, and A and B to their respective columns in M. To find a least squares solution, we square Eq. (4):

$$\begin{pmatrix} \mathbf{r}^\top\ \mathbf{t}^\top \end{pmatrix} (\mathtt{A}\ \mathtt{B})^\top (\mathtt{A}\ \mathtt{B}) \begin{pmatrix} \mathbf{r} \\ \mathbf{t} \end{pmatrix} \rightarrow min. \tag{5}$$

Deriving of the term for $\mathbf{t}$ and setting to zero yields

$$\mathbf{t} = -(\mathtt{B}^\top\mathtt{B})^+\mathtt{B}^\top\mathtt{A}\mathbf{r} = -\mathtt{B}^+\mathtt{A}\mathbf{r}, \tag{6}$$

where $\mathtt{B}^+$ denotes the pseudo-inverse of the matrix B. Eq. (4) can now be rewritten as

$$\mathtt{A}\mathbf{r} - \mathtt{B}\mathtt{B}^+\mathtt{A}\mathbf{r} = 0. \tag{7}$$

Considering the economy-size SVD[2] of $\mathtt{B} = \mathtt{U}\mathtt{S}\mathtt{V}^\top$, this simplifies to

$$(\mathtt{A} - \mathtt{U}\mathtt{U}^\top\mathtt{A})\mathbf{r} = 0. \tag{8}$$

Eq. (8) is a least squares problem in the rotation $\mathbf{r}$, whose solution can be found as the nullvector of SVD. By definition of the SVD, the resulting vector $\mathbf{r}$ will have length 1. Hence, a multiplication by $\sqrt{3}$ gives the desired Frobenius norm for the rotation. $\mathbf{t}$ is then obtained by backsubstitution into Eq. (6).

As we are basically only solving for an affine transformation, it is advisable to orthogonalise the obtained solution for R *before* solving for $\mathbf{t}$. Compared to QR decomposition and quaternions, SVD yielded the best results for this (QR and quaternions were both about 10% worse in average w.r.t. accuracy). The SVD might return an orthogonal R with $\det(\mathtt{R}) = -1$, which can be fixed with a sign reversal of both R and $\mathbf{t}$.

[2]The economy-size SVD of an $m \times n$ matrix yields an $m \times n$ matrix U, containing only the columns corresponding to non-zero singular values. $\mathtt{U}\mathtt{U}^\top$ consequently does *not* correspond to the identity. S and V are $n \times n$.

2.1 4 or 5 Points

As each point correspondence imposes only two constraints, a minimum of six point correspondences is needed to solve the pose estimation problem uniquely. For less points, the dimension N of the nullspace of Eq. (8) will not be 1 (disregarding degenerate cases, $N = 12 - 2n$). Let's denote the nullspace of Eq. (8) by

$$\mathbf{X} = \{\mathbf{r}_1, \ldots, \mathbf{r}_N\} \tag{9}$$

The correct solution $\mathbf{r}$ is a linear combination of the vectors $\mathbf{r}_i$. That is, we need to find scalars α_i, such that

$$\mathbf{r} = \sum_i \alpha_i \mathbf{r}_i. \tag{10}$$

To obtain this, a quadratic system based on the constraints $\mathtt{R}^\top\mathtt{R} = \mathtt{R}\mathtt{R}^\top = \mathtt{I}$ can be constructed. These quadratic constraints on $\mathtt{R}$ are formulated in the quadratic terms of its elements, $r^{(i)}r^{(j)}$, where $r^{(i)}$ is given by Eq. (10). Now, we linearise the resulting quadratic terms in $\alpha_i\alpha_j$ by substituting them with new variables α_{ij}. Due to the linearised nature of the formulation, there are eleven independent constraints on the r_{ij}, and hence on the α_{ij}. Thereby, we obtain a linear equation system in the $\frac{N(N+1)}{2}$ scalars α_{ij}. For five points, this corresponds to three unknowns, for four points, to ten unknowns. The α_i can be recovered up to a global sign ambiguity from the α_{ii}. As above, this ambiguity is resolved by enforcing $\det(\mathtt{R}) = 1$.

2.2 Lines

The chosen formulation also allows for the direct inclusion of lines (or other features, for that matter) into the pose estimation. In contrast to previous work [1], both lines and points can be used at the same time to constrain the pose. A line's direction imposes the following constraint on $\mathtt{R}$:

$$\mathbf{n}^\top\mathtt{R}\mathbf{d} = 0, \tag{11}$$

with $\mathbf{d}$ being the unit vector of the 3D line's direction and $\mathbf{n} = (a,b,c)^\top$ the normalised line in 2D. $\mathbf{n}$ is set to unit length as well, and equals the normal vector of the plane obtained from the 2D line's back-projection. In this formulation—which proved to be the most stable one in our experiments—each line imposes only one constraint on $\mathtt{R}$. Still, using the steps from subsection 2.1, five instead of nine lines suffice to estimate the pose.

2.3 Multiple Cameras

When the joint pose estimation from multiple, externally calibrated cameras is desired, the right hand side of Eq. (4) is not $\mathbf{0}$ anymore. The solution to the resulting constrained problem $\mathtt{A}\mathbf{x} = \mathbf{b}, \|\mathbf{x}\| = 1$ then can be found either using differential-geometry methods [4], or by introducing a homogeneous variable, as done for planar motion in Section 3. As latter disables the direct constraint $\|R\|_F = 3$, former is to be preferred.

2.4 Depth Dependency

Clearly, using object-space collinearity instead of the more correct image-space collinearity seems like a disadvantage. Please note however that the depth λ_i is mainly a weighting

factor in Eq. (1), for which in many cases an estimate is available. In Section 4, we also examine the algorithm's performance subject to different depth ranges.

3 Planar Motion

The special case of robot navigation in indoor environments typically restricts the movements to planar ones. This greatly reduces the number of unknowns and can thereby increase stability and performance. There has been quite some research activity in this field, mostly in the uncalibrated case, cf. e.g. [2, 5]. Stewenius *et al.* [16] treat various problems for multi-camera setups subject to planar motion. Their resection algorithm is similar to the one presented in the following, was however very susceptible to noise in our tests.

In the planar case, the motion variables reduce to

$$\mathtt{R} = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}, \quad \mathbf{t} = \begin{pmatrix} t_x \\ 0 \\ t_z \end{pmatrix} \tag{12}$$

Inserting into Eq. (3), this yields equations of the form

$$(Q_{11}p_x + Q_{31}p_z)c + (-Q_{31}p_x + Q_{11}p_z)s + (1-Q_{11})t_x + (-Q_{13})t_z = -Q_{21} \tag{13}$$

$$(Q_{13}p_x + Q_{33}p_z)c + (-Q_{33}p_x + Q_{13}p_z)s + (Q_{31})t_x + (1-Q_{33})t_z = -Q_{23}, \tag{14}$$

where Q_{ij} corresponds to the entries of the projection matrix $\mathtt{Q}$. Besides the necessary variables for $c = \cos\theta$ and $s = \sin\theta$, we introduce an additional homogeneous variable ρ for Q_{21} and Q_{23} to arrive at a formulation needed for Eq. (4). After having solved for $\mathbf{r} = (c', s', \rho)^\top$ in Eq. (8), we remove the homogeneous quantity by dividing through ρ. A valid rotation is then inferred from c and s.

In the reduced formulation, three points suffice to determine the pose using Eq. (8) and Eq. (6) without any further processing. Vertical lines can be treated as points when considering their intersection with a horizontal plane. Also, due to the introduction of the homogeneous variable, camera rigs can be handled in a straight-forward fashion.

4 Results

In the following, we will give various simulation results of our algorithm ("GLPE") compared to the approaches suggested by [1] ("Ansar") and [6] ("Fiore"), as well as the solution obtained using Levenberg-Marquardt ("LM", initialised using GLPE). Due to its iterative nature, the latter is expected to always come closest to the actual minimum; we include it mainly as a reference. In all cases, the rotational error is measured as the absolute error in the unit quaternion, $|q - q_0|$. The relative translational error is measured as $\frac{2\|t-t_0\|_2}{\|t\|_2+\|t_0\|_2}$. q_0 and t_0 refer to the ground truth for rotation and translation, respectively.

4.1 Simulation

All the results in this section were obtained by averaging over 5,000 runs for the respective scenario. Different configurations for the point clouds (different sizes and shapes, such

as pyramids and cubes) were tested, but the basic results with respect to accuracy and the relative ordering among the algorithms remained the same. Thus, we only report the results that were obtained by randomly sampling points in a $10,000 \times 10,000 \times 10,000$ cube, centered around the z-axis, and placed in front of a camera in the origin. The point cloud's barycentre is then assumed as new origin, around which the entire system is rotated randomly. A focal length of $1,500$ was chosen, with principal point $\mathbf{0}$, and aspect ratio 1. Gaussian noise is added to the projected image points.

Fig. 1 shows the results obtained by keeping the number of points constant at six and increasing the noise level. For the rotational error, our algorithm clearly outperforms the other linear approaches. The error in translation is comparable between the three algorithms. As an interesting sidenote, Fiore performs very comparably to Ansar, as opposed to the findings by [1].

Next, we compare the performance with respect to the number of points used, Fig. 2. We perform slightly worse than Ansar for the 4-point case, as only eleven constraints are imposed on the ten unknowns, cf. Section 2.1. The 5-point case exhibits a drastic gain in accuracy and outperforms Ansar's algorithm. There is another considerable improvement for more than six points, as this presents the overdetermined case. Note that Fiore's approach could not be used with less than six points. In general, it performs rather similar to our algorithm. However, besides offering various useful extensions (lines, planar motion), we found our algorithm to be more reliable in a real scenario, as shown later.

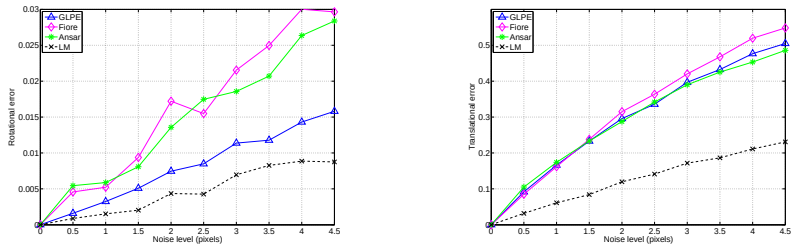

Figure 1: Rotational and translational error with increasing Gaussian noise (6 points).

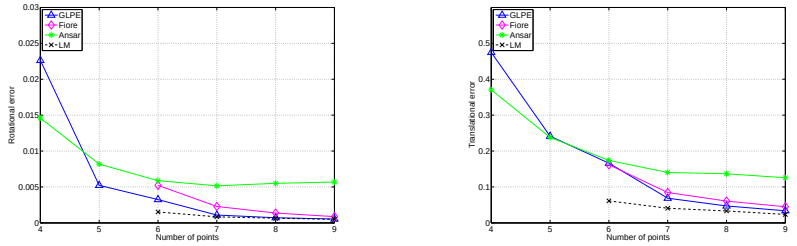

Figure 2: Performance with increasing number of points (noise $\sigma = 1.5$ pixels).

For lines, we compare our algorithm to the line algorithm proposed in [1], Fig. 3. While our formulation needs at least five lines (as opposed to four), it is more accurate in all tests. Also, keep in mind that our formulation would allow for a mixed solution, which is not possible using their approach.

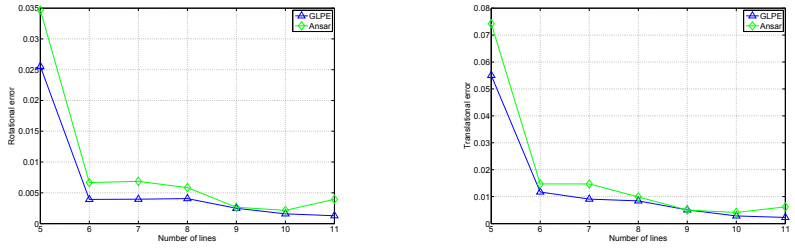

Figure 3: Performance with increasing number of lines (noise $\sigma = 1.5$ pixels).

4.2 Depth Dependency

Lastly, we investigate the performance of the various algorithms by increasing the depth range where points can come from, Figure 4. Again, we use 6 points and a noise standard deviation of 1.5 pixels. As can be seen, the rotational error is largely unaffected by increasing depth range, however, translational error grows. As mentioned above, in most cases, a depth estimate is known for the points (especially when processing video sequences), which can be used to weight the influence of each point.

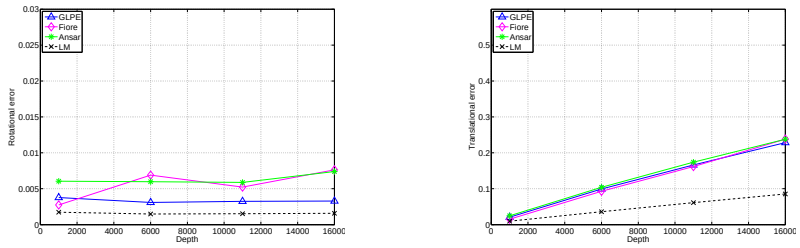

Figure 4: Performance with increasing depth range (6 points, noise $\sigma = 1.5$).

4.2.1 Planar Motion

For the planar case, we compare our algorithm with the one of Stewenius *et al.* [16]. Obviously, it is not quite fair to compare any of the above algorithms with the special case of planar movement. Nonetheless, for illustrative purposes, we add Fiore's algorithm to the comparison. Fig. 5 and Fig. 6 again give results for varying noise level and number of points, respectively. As to be expected, our algorithm outperforms Fiore for the same number of points, especially when noise levels are increasing. Notice however that the planar version already outperforms Fiore given three points less, which is basically the number of missing degrees of freedom. The algorithm of Stewenius proved to be very susceptible to noise in our experiments and was hence left out in Fig. 6.

4.3 Timings

We time straightforward C-implementations of the aforementioned algorithms, using VXL for the numerical routines (everything was compiled *without* optimisations). Table 4.3 gives an overview of the runtimes in ms on a 2.4 GHz AMD Opteron for various numbers

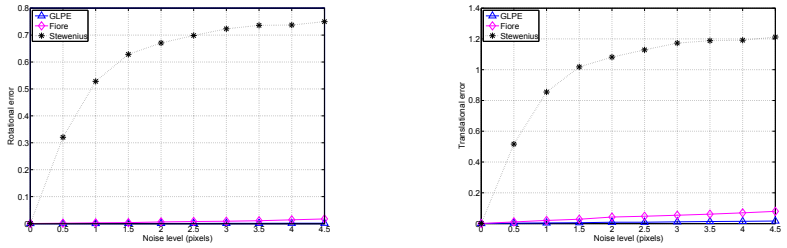

Figure 5: Performance with increasing Gaussian noise in the planar case (6 points).

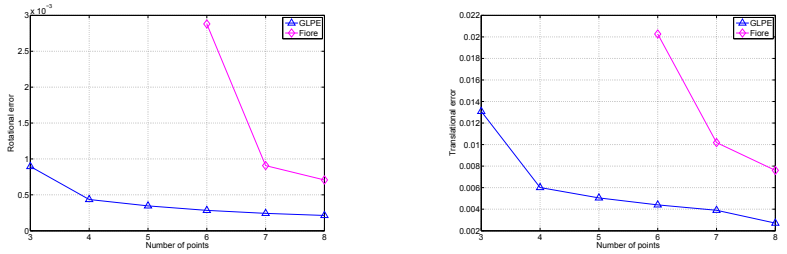

Figure 6: Performance with increasing number of points in the planar case.

of points, averaged over 10,000 runs. As expected, our algorithm has very low runtimes due to its small equation system. In the 4- and 5-point cases, some additional overhead is needed for finding the correct linear combination in the nullspace (see section 2.1), hence the slightly higher runtime. The size of this second equation system decreases enough in the 5-point case to yield attractive timing figures. Fiore provides an overall fast performance, but is slightly slower than our algorithm. With an increasing number of points, the margin between Fiore and our algorithm increases (note however that Fiore proposes an optimisation that is not implemented here). Using many points at low cost in time is attractive for providing a better initialisation using all inliers, and hence quicker convergence, for a nonlinear algorithm after running RANSAC [7]. Ansar's algorithm becomes very slow for an increasing number of points, as the size of the equation system increases quadratically as opposed to linearly as in our and Fiore's formulation.

The planar version takes 0.060 ms for the 4-point and 0.086 ms for the 8-point case.

Algorithm	4 points	5 points	6 points	8 points	50 points	100 points
GLPE	0.346	0.205	0.194	0.227	0.865	1.657
Ansar	0.727	2.016	5.159	28.031	-	-
Fiore	-	-	0.201	0.311	9.362	38.590

Table 1: Runtimes in ms for different algorithms on a 2.4 GHz AMD Opteron.

4.4 Real Scenario

To verify the results obtained in the above simulations, we apply the presented method to a small video sequence taken from a stereo rig, which undergoes a planar motion. Two

moving persons are walking in front of the robot, introducing an additional challenge. In all images, SURF features [3] are detected. Similar to [13], an initial triangulation from the stereo rig is used for the world reference points. The pose is estimated using either Fiore's, Ansar's, or our approach with 6 points, or our planar stereo 4-point algorithm. At every iteration, all inliers are used to iteratively refine the pose. Every fifth frame, the world points are discarded and new ones are triangulated from the current stereo frame. A few sample frames, as well as the obtained tracks are shown in Fig. 7. Even without using any bundle adjustment, our algorithms yield better results than using either Fiore's or Ansar's method. Already the 5-point version gives an acceptable result. The planar version, with sampling from both cameras of the stereo rig, exposes its advantages in this constrained scenario: while the other methods show quite some variance in y-direction (up to 20cm, not visible in the figures), its track sticks nicely to the ground. For comparison purposes, we also include the track generated using the 3-point algorithm and bundle adjustment.

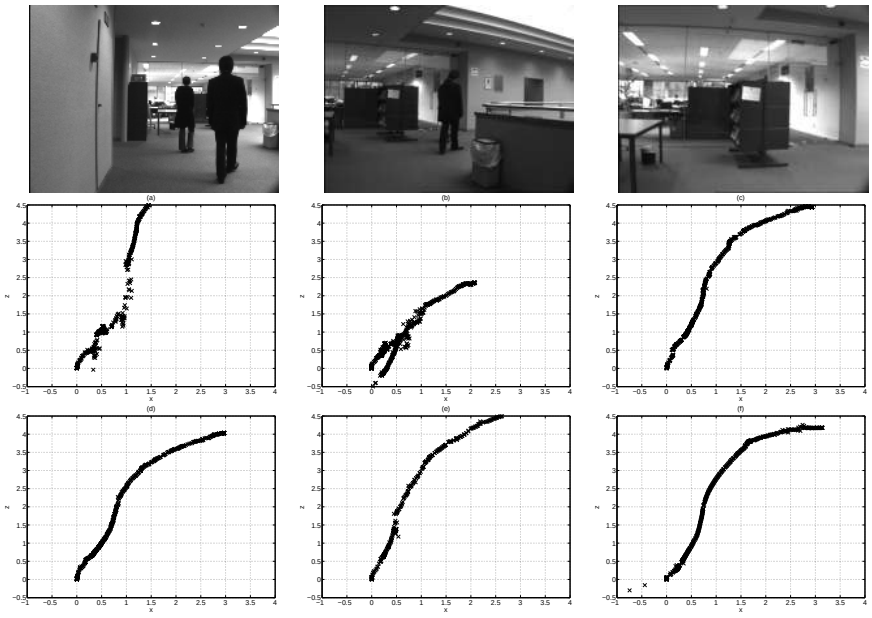

Figure 7: Real scenario. Top: frames with two moving objects. Middle: tracks found by 6-point versions of Fiore (a), Ansar (b) and GLPE (c). Bottom: 5-point version of GLPE (d), stereo planar GLPE 4-point (e), and 3-point relative pose with bundle adjustment (f).

As mentioned in the introduction, the algorithm can be applied to other scenarios such as augmented reality. In general, a better performance can be expected in cases where e.g. a 3D CAD model is already given, since this omits additional errors from 3D reconstruction as in the example shown here.

5 Conclusions

We introduced a novel algorithm for linear pose estimation from $n \geq 4$ points. Aimed at robot navigation in poorly-textured indoor environments, the algorithm's main advantage is its adaptability to different scenarios such as planar movement, stereo rigs or the inclusion of lines and other features besides points. Extensive simulation results show its general performance to be comparable or slightly better than other linear algorithms, at increased speed. In a real scenario, the algorithm outperforms other linear approaches, even without making use of advantages such as the support of stereo rigs or the ability to constrain the motion.

Acknowledgements

The authors acknowledge the support of Toyota Motor Europe and Toyota Motor Corporation. They would like to thank Kostas Daniilidis for providing the Matlab source code for their point algorithm.

References

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

[2] M. Armstrong, A. Zisserman, and R. I. Hartley. Self-calibration from image triplets. In *ECCV (1)*, pages 3–16, 1996.

[3] H. Bay, T. Tuytelaars, and L. van Gool. Surf: Speeded up robust features. In *ECCV*, 2006.

[4] L. Elden. Solving quadratically constrained least squares problems using a differential-geometric approach. *BIT*, 42:323–335, 2002.

[5] O.D. Faugeras, L. Quan, and P.F. Sturm. Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera. *PAMI*, 22(10):1179–1185, 2000.

[6] P.D. Fiore. Efficient linear solution of exterior orientation. *PAMI*, 23(2):140–148, 2001.

[7] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6), 1981.

[8] J.A. Grunert. Das pothenotische problem in erweiterter gestalt nebst über seine anwendungen in der geodäsie. *Grunerts Archiv für Mathematik und Physik, Band 1*, pages 238–248, 1841.

[9] R. M. Haralick, C. Lee, K. Ottenberg, and M. Noelle. Review and analysis of solutions of the three point perspective pose estimation problem. *IJCV*, 13(3):331–356, 1994.

[10] B. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of Optical Society of America A*, Vol 4., No. 4:629–642, 1987.

[11] Y. Liu, T. S. Huang, and O. D. Faugeras. Determination of camera location from 2-d to 3-d line and point correspondences. *PAMI*, 12(1):28–37, 1990.

[12] D. Nistér. A minimal solution to the generalised 3-point pose problem. In *CVPR (1)*, 2004.

[13] D. Nistér, O. Naroditsky, and J. R. Bergen. Visual odometry. In *CVPR (1)*, 2004.

[14] T.Q. Phong, R. Horaud, A. Yassine, and P.D. Tao. Object pose from 2-d to 3-d point and line correspondences. *IJCV*, 15:225–243, 1995.

[15] L. Quan and Z.-D. Lan. Linear n-point camera pose determination. *PAMI*, 21(8):774–780, 1999.

[16] H. Stewénius, M. Oskarsson, and K. Åström. Reconstruction from planar motion image sequences with applications for autonomous vehicles. In *SCIA*, pages 609–618, 2005.