Int J Comput Vis (2009) 81: 155–166
DOI 10.1007/s11263-008-0152-6

EPnP: An Accurate $O(n)$ Solution to the PnP Problem

Vincent Lepetit · Francesc Moreno-Noguer · Pascal Fua

Received: 15 April 2008 / Accepted: 25 June 2008 / Published online: 19 July 2008

Abstract We propose a non-iterative solution to the PnP problem—the estimation of the pose of a calibrated camera from n 3D-to-2D point correspondences—whose computational complexity grows linearly with n. This is in contrast to state-of-the-art methods that are $O(n^5)$ or even $O(n^8)$, without being more accurate. Our method is applicable for all $n \geq 4$ and handles properly both planar and non-planar configurations. Our central idea is to express the n 3D points as a weighted sum of four virtual control points. The problem then reduces to estimating the coordinates of these control points in the camera referential, which can be done in $O(n)$ time by expressing these coordinates as weighted sum of the eigenvectors of a 12×12 matrix and solving a small *constant* number of quadratic equations to pick the right weights. Furthermore, if maximal precision is required, the output of the closed-form solution can be used to initialize a Gauss-Newton scheme, which improves accuracy with negligible amount of additional time. The advantages of our method are demonstrated by thorough testing on both synthetic and real-data.[1]

Keywords Pose estimation · Perspective-n-Point · Absolute orientation

[1]The Matlab and C++ implementations of the algorithm presented in this paper are available online at http://cvlab.epfl.ch/software/EPnP/.

V. Lepetit · F. Moreno-Noguer (✉) · P. Fua
Computer Vision Laboratory, École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland
e-mail: fmorenoguer@gmail.com

1 Introduction

The aim of the Perspective-n-Point problem—PnP in short—is to determine the position and orientation of a camera given its intrinsic parameters and a set of n correspondences between 3D points and their 2D projections. It has many applications in Computer Vision, Robotics, Augmented Reality and has received much attention in both the Photogrammetry (McGlove et al. 2004) and Computer Vision (Hartley and Zisserman 2000) communities. In particular, applications such as feature point-based camera tracking (Skrypnyk and Lowe 2004; Lepetit and Fua 2006) require dealing with hundreds of noisy feature points in real-time, which requires computationally efficient methods.

In this paper, we introduce a non-iterative solution with better accuracy and much lower computational complexity than non-iterative state-of-the-art methods, and much faster than iterative ones with little loss of accuracy. Our approach is $O(n)$ for $n \geq 4$ whereas all other methods we know of are either specialized for small fixed values of n, very sensitive to noise, or much slower. The specialized methods include those designed to solve the P3P problem (Gao et al. 2003; Quan and Lan 1999). Among those that handle arbitrary values of n (Fischler and Bolles 1981; Dhome et al. 1989; Horaud et al. 1989; Haralick et al. 1991; Quan and Lan 1999; Triggs 1999; Fiore 2001; Ansar and Daniilidis 2003; Gao et al. 2003), the lowest-complexity one (Fiore 2001) is $O(n^2)$ but has been shown to be unstable for noisy 2D locations (Ansar and Daniilidis 2003). This is currently addressed by algorithms that are $O(n^5)$ (Quan and Lan 1999) or even $O(n^8)$ (Ansar and Daniilidis 2003) for better accuracy whereas our $O(n)$ approach achieves even better accuracy and reduced sensitivity to noise, as depicted by Fig. 1 in the $n = 6$ case and demonstrated for larger values of n in the result section.

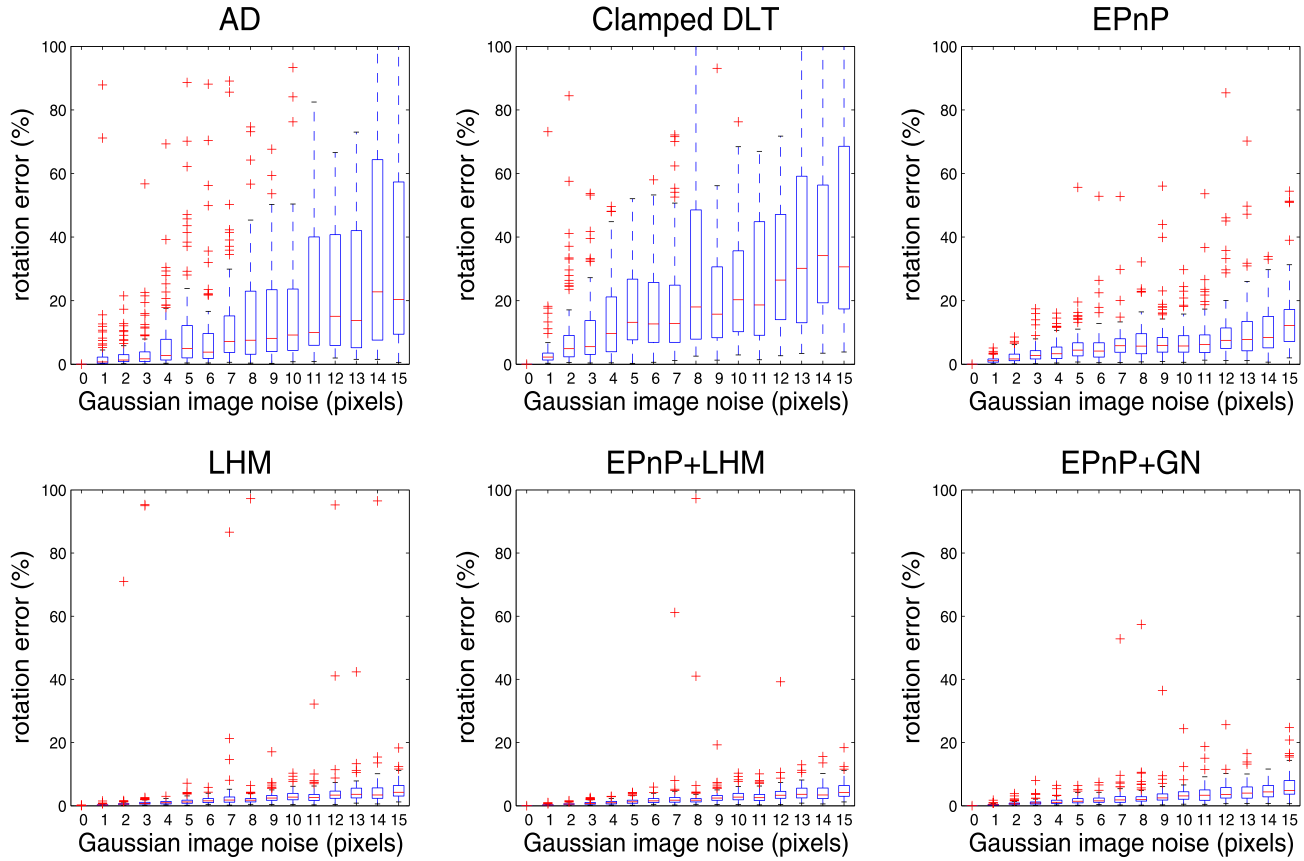

Fig. 1 Comparing the accuracy of our method against state-of-the-art ones. We use the boxplot representation: The boxes denote the first and third quartiles of the errors, the lines extending from each end of the box depict the statistical extent of the data, and the crosses indicate observations that fall out of it. *Top row*: Accuracy of non-iterative methods as a function of noise when using $n = 6$ 3D-to-2D correspondences: AD is the method of Ansar and Daniilidis (2003); Clamped DLT is the DLT algorithm after clamping the internal parameters with their known values; and EPnP is our method. *Bottom row*: Accuracy of iterative methods using $n = 6$: LHM is Lu's et al. method (Lu et al. 2000) initialized with a weak perspective assumption; EPnP + LHM is Lu's et al. algorithm initialized with the output of our algorithm; EPnP + GN, our method followed by a Gauss-Newton optimization

A natural alternative to non-iterative approaches are iterative ones (Lowe 1991; DeMenthon and Davis 1995; Horaud et al. 1997; Kumar and Hanson 1994; Lu et al. 2000) that rely on minimizing an appropriate criterion. They can deal with arbitrary numbers of correspondences and achieve excellent precision when they converge properly. In particular, Lu et al. (2000) introduced a very accurate algorithm, which is fast in comparison with other iterative ones but slow compared to non-iterative methods. As shown in Figs. 1 and 2, our method achieves an accuracy that is almost as good, and is much faster and without requiring an initial estimate. This is significant because iterative methods are prone to failure if poorly initialized. For instance, Lu's et al. approach relies on an initial estimation of the camera pose based on a weak-perspective assumption, which can lead to instabilities when the assumption is not satisfied. This happens when the points of the object are projected onto a small region on the side of the image and our solution performs more robustly under these circumstances.

Furthermore, if maximal precision is required our output can be used to initialize Lu's et al., yielding both higher stability and faster convergence. Similarly, we can run a Gauss-Newton scheme that improves our closed-form solution to the point where it is as accurate as the one produced by Lu's et al. method when it is initialized by our method. Remarkably, this can be done with only very little extra computation, which means that even with this extra step, our method remains much faster. In fact, the optimization is performed in constant time, and hence, the overall solution still remains $O(n)$.

Our central idea is to write the coordinates of the n 3D points as a weighted sum of four virtual control points. This reduces the problem to estimating the coordinates of the control points in the camera referential, which can be done in $O(n)$ time by expressing these coordinates as weighted sum of the eigenvectors of a 12×12 matrix and solving a small *constant* number of quadratic equations to pick the right weights. Our approach also extends to planar config-

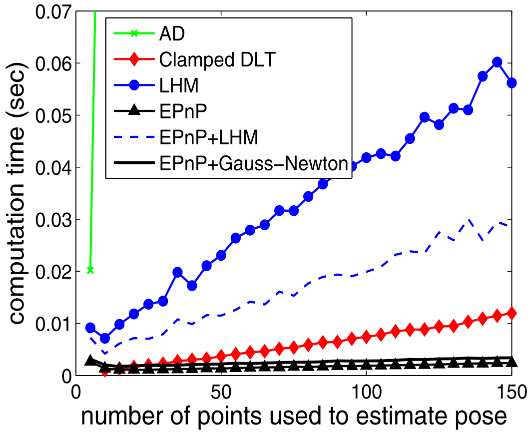

Fig. 2 Comparing computation times of our method against the state-of-the-art ones introduced in Fig. 1. The computation times of a MATLAB implementation on a standard PC, are plotted as a function of the number of correspondences. Our method is both more accurate—see Fig. 1—and faster than the other non-iterative ones, especially for large amounts of noise, and is almost as accurate as the iterative LHM. Furthermore, if maximal precision is required, the output of our algorithm can be used to initialize a Gauss-Newton optimization procedure which requires a negligible amount of additional time

urations, which cause problems for some methods as discussed in Oberkampf et al. (1996), Schweighofer and Pinz (2006), by using three control points instead of four.

In the remainder of the paper, we first discuss related work focusing on accuracy and computational complexity. We then introduce our new formulation and derive our system of linear and quadratic equations. Finally, we compare our method against the state-of-the-art ones using synthetic data and demonstrate it using real data. This paper is an expanded version of that in Moreno-Noguer et al. (2007), where a final Gauss-Newton optimization is added to the original algorithm. In Sect. 4 we show that optimizing over a reduced number of parameters, the accuracy of the closed-solution proposed in Moreno-Noguer et al. (2007) is considerably improved with almost no additional computational cost.

2 Related Work

There is an immense body of literature on pose estimation from point correspondences and, here, we focus on non-iterative approaches since our method falls in this category. In addition, we will also introduce the Lu et al. (2000) iterative method, which yields very good results and against which we compare our own approach.

Most of the non-iterative approaches, if not all of them, proceed by first estimating the points 3D positions in the camera coordinate system by solving for the points depths. It is then easy to retrieve the camera position and orientation as the Euclidean motion that aligns these positions on the given coordinates in the world coordinate system (Horn et al. 1988; Arun et al. 1987; Umeyama 1991).

The P3P case has been extensively studied in the literature, and many closed form solutions have been proposed such as Dhome et al. (1989), Fischler and Bolles (1981), Gao et al. (2003), Haralick et al. (1991), Quan and Lan (1999). It typically involves solving for the roots of an eight-degree polynomial with only even terms, yielding up to four solutions in general, so that a fourth point is needed for disambiguation. Fisher and Bolles (1981) reduced the P4P problem to the P3P one by taking subsets of three points and checking consistency. Similarly, Horaud et al. (1989) reduced the P4P to a 3-line problem. For the 4 and 5 points problem, Triggs (1999) derived a system of quadratic polynomials, which solves using multiresultant theory. However, as pointed out in Ansar and Daniilidis (2003), this does not perform well for larger number of points.

Even if four correspondences are sufficient in general to estimate the pose, it is nonetheless desirable to consider larger point sets to introduce redundancy and reduce the sensitivity to noise. To do so, Quan and Lan (1999) consider triplets of points and for each one derive four-degree polynomials in the unknown point depths. The coefficients of these polynomials are then arranged in a $\frac{(n-1)(n-2)}{2} \times 5$ matrix and singular value decomposition (SVD) is used to estimate the unknown depths. This method is repeated for all of the n points and therefore involves $O(n^5)$ operations.[2] It should be noted that, even if it is not done in Quan and Lan (1999), this complexity could be reduced to $O(n^3)$ by applying the same trick as we do when performing the SVD, but even then, it would remain slower than our method. Ansar and Daniilidis (2003) derive a set of quadratic equations arranged in a $\frac{n(n-1)}{2} \times (\frac{n(n+1)}{2} + 1)$ linear system, which, as formulated in the paper, requires $O(n^8)$ operations to be solved. They show their approach performs better than Quan and Lan (1999).

The complexity of the previous two approaches stems from the fact that quadratic terms are introduced from the inter-point distances constraints. The linearization of these equations produces additional parameters, which increase the complexity of the system. Fiore's method (Fiore 2001) avoids the need for these constraints: He initially forms a set of linear equations from which the world to camera rotation and translation parameters are eliminated, allowing the direct recovery of the point depths without considering the inter-point distances. This procedure allows the estimation of the camera pose in $O(n^2)$ operations, which makes real-time performance possible for large n. Unfortunately, ignoring nonlinear constraints produces poor results in the presence of noise (Ansar and Daniilidis 2003).

[2]Following Golub and Van Loan (1996), we consider that the SVD for a $m \times n$ matrix can be computed by a $O(4m^2n + 8mn^2 + 9n^3)$ algorithm.

By contrast, our method is able to consider nonlinear constraints but requires $O(n)$ operations only. Furthermore, in our synthetic experiments, it yields results that are more accurate than those of Ansar and Daniilidis (2003).

It is also worth mentioning that for large values of n one could use the Direct Linear Transformation (DLT) algorithm (Abdel-Aziz and Karara 1971; Hartley and Zisserman 2000). However, it ignores the intrinsic camera parameters we assume to be known, and therefore generally leads to less stable pose estimate. A way to exploit our knowledge of the intrinsic parameters is to clamp the retrieved values to the known ones, but the accuracy still remains low.

Finally, among iterative methods, Lu's et al. (2000) is one of the fastest and most accurate. It minimizes an error expressed in 3D space, unlike many earlier methods that attempt to minimize reprojection residuals. The main difficulty is to impose the orthonormality of the rotation matrix. It is done by optimizing alternatively on the translation vector and the rotation matrix. In practice, the algorithm tends to converge fast but can get stuck in an inappropriate local minimum if incorrectly initialized. Our experiments show our closed-form solution is slightly less accurate than Lu's et al. when it find the correct minimum, but also that it is faster and more stable. Accuracies become similar when after the closed-form solution we apply a Gauss-Newton optimization, with almost negligible computational cost.

3 Our Approach to the P*n*P Problem

Let us assume we are given a set of n *reference points* whose 3D coordinates are known in the world coordinate system and whose 2D image projections are also known. As most of the proposed solutions to the P*n*P Problem, we aim at retrieving their coordinates in the camera coordinate system. It is then easy and standard to retrieve the orientation and translation as the Euclidean motion that aligns both sets of coordinates (Horn et al. 1988; Arun et al. 1987; Umeyama 1991).

Most existing approaches attempt to solve for the depths of the reference points in the camera coordinate system. By contrast, we express their coordinates as a weighted sum of virtual *control points*. We need 4 non-coplanar such control points for general configurations, and only 3 for planar configurations. Given this formulation, the coordinates of the control points in the camera coordinate system become the unknown of our problem. For large n's, this is a much smaller number of unknowns that the n depth values that traditional approaches have to deal with and is key to our efficient implementation.

The solution of our problem can be expressed as a vector that lies in the kernel of a matrix of size $2n \times 12$ or $2n \times 9$. We denote this matrix as $\mathbf{M}$ and can be easily computed from the 3D world coordinates of the reference points and their 2D image projections. More precisely, it is a weighted sum of the null eigenvectors of $\mathbf{M}$. Given that the correct linear combination is the one that yields 3D camera coordinates for the control points that preserve their distances, we can find the appropriate weights by solving small systems of quadratic equations, which can be done at a negligible computational cost. In fact, for n sufficiently large—about 15 in our implementation—the most expensive part of this whole computation is that of the matrix $\mathbf{M}^\top\mathbf{M}$, which grows linearly with n.

In the remainder of this section, we first discuss our parameterization in terms of control points in the generic nonplanar case. We then derive the matrix $\mathbf{M}$ in whose kernel the solution must lie and introduce the quadratic constraints required to find the proper combination of eigenvectors. Finally, we show that this approach also applies to the planar case.

3.1 Parameterization in the General Case

Let the reference points, that is, the n points whose 3D coordinates are known in the world coordinate system, be

$$\mathbf{p}_i, \quad i = 1, \ldots, n.$$

Similarly, let the 4 control points we use to express their world coordinates be

$$\mathbf{c}_j, \quad j = 1, \ldots, 4.$$

When necessary, we will specify that the point coordinates are expressed in the world coordinate system by using the ^{w} superscript, and in the camera coordinate system by using the ^{c} superscript. We express each reference point as a weighted sum of the control points

$$\mathbf{p}_i^w = \sum_{j=1}^{4} \alpha_{ij}\mathbf{c}_j^w, \quad \text{with} \sum_{j=1}^{4} \alpha_{ij} = 1, \tag{1}$$

where the α_{ij} are homogeneous barycentric coordinates. They are uniquely defined and can easily be estimated. The same relation holds in the camera coordinate system and we can also write

$$\mathbf{p}_i^c = \sum_{j=1}^{4} \alpha_{ij}\mathbf{c}_j^c. \tag{2}$$

In theory the control points can be chosen arbitrarily. However, in practice, we have found that the stability of our method is increased by taking the centroid of the reference points as one, and to select the rest in such a way that they form a basis aligned with the principal directions

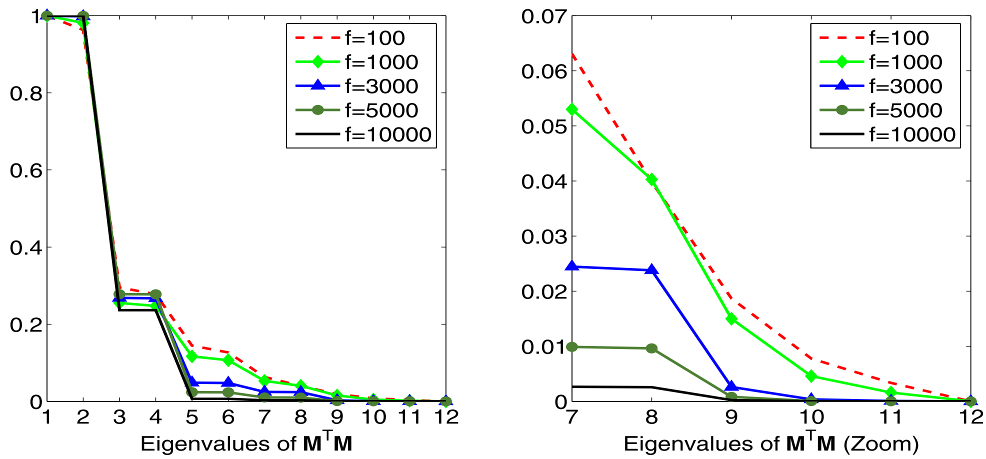

Fig. 3 *Left*: Singular values of $\mathbf{M}^{\top}\mathbf{M}$ for different focal lengths. Each curve averages 100 synthetic trials. *Right*: Zooming in on the smallest eigenvalues. For small focal lengths, the camera is perspective and only one eigenvalue is zero, which reflects the scale ambiguity. As the focal length increases and the camera becomes orthographic, all four smallest eigenvalues approach zero

of the data. This makes sense because it amounts to conditioning the linear system of equations that are introduced below by normalizing the point coordinates in a way that is very similar to the one recommended for the classic DLT algorithm (Hartley and Zisserman 2000).

3.2 The Solution as Weighted Sum of Eigenvectors

We now derive the matrix $\mathbf{M}$ in whose kernel the solution must lie given that the 2D projections of the reference points are known. Let $\mathbf{A}$ be the camera internal calibration matrix and $\{\mathbf{u}_i\}_{i=1,\dots,n}$ the 2D projections of the $\{\mathbf{p}_i\}_{i=1,\dots,n}$ reference points. We have

$$\forall i, \quad w_i \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} = \mathbf{A}\mathbf{p}_i^c = \mathbf{A}\sum_{j=1}^{4}\alpha_{ij}\mathbf{c}_j^c, \tag{3}$$

where the w_i are scalar projective parameters. We now expand this expression by considering the specific 3D coordinates $[x_j^c, y_j^c, z_j^c]^{\top}$ of each $\mathbf{c}_j^c$ control point, the 2D coordinates $[u_i, v_i]^{\top}$ of the $\mathbf{u}_i$ projections, and the f_u, f_v focal length coefficients and the (u_c, v_c) principal point that appear in the $\mathbf{A}$ matrix. Equation 3 then becomes

$$\forall i, \quad w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4}\alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix}. \tag{4}$$

The unknown parameters of this linear system are the 12 control point coordinates $\{(x_j^c, y_j^c, z_j^c)\}_{j=1,\dots,4}$ and the n projective parameters $\{w_i\}_{i=1,\dots,n}$. The last row of (4) implies that $w_i = \sum_{j=1}^{4}\alpha_{ij}z_j^c$. Substituting this expression in the first two rows yields two linear equations for each reference point:

$$\sum_{j=1}^{4}\alpha_{ij}f_u x_j^c + \alpha_{ij}(u_c - u_i)z_j^c = 0, \tag{5}$$

$$\sum_{j=1}^{4}\alpha_{ij}f_v y_j^c + \alpha_{ij}(v_c - v_i)z_j^c = 0. \tag{6}$$

Note that the w_i projective parameter does not appear anymore in those equations. Hence, by concatenating them for all n reference points, we generate a linear system of the form

$$\mathbf{M}\mathbf{x} = \mathbf{0}, \tag{7}$$

where $\mathbf{x} = [\mathbf{c}_1^{c\top}, \mathbf{c}_2^{c\top}, \mathbf{c}_3^{c\top}, \mathbf{c}_4^{c\top}]^{\top}$ is a 12-vector made of the unknowns, and $\mathbf{M}$ is a $2n \times 12$ matrix, generated by arranging the coefficients of (5) and (6) for each reference point. Unlike in the case of DLT, we do not have to normalize the 2D projections since (5) and (6) do not involve the image referential system.

The solution therefore belongs to the null space, or kernel, of $\mathbf{M}$, and can be expressed as

$$\mathbf{x} = \sum_{i=1}^{N}\beta_i\mathbf{v}_i \tag{8}$$

where the set $\mathbf{v}_i$ are the columns of the right-singular vectors of $\mathbf{M}$ corresponding to the N null singular values of $\mathbf{M}$. They can be found efficiently as the null eigenvectors of matrix $\mathbf{M}^{\top}\mathbf{M}$, which is of small constant (12×12) size. Computing the product $\mathbf{M}^{\top}\mathbf{M}$ has $O(n)$ complexity, and is the most time consuming step in our method when n is sufficiently large, about 15 in our implementation.

3.3 Choosing the Right Linear Combination

Given that the solution can be expressed as a linear combination of the null eigenvectors of $\mathbf{M}^{\top}\mathbf{M}$, finding it amounts to computing the appropriate values for the $\{\beta_i\}_{i=1,\dots,N}$ coefficients of (8). Note that this approach applies even when the system of (7) is under-constrained, for example because

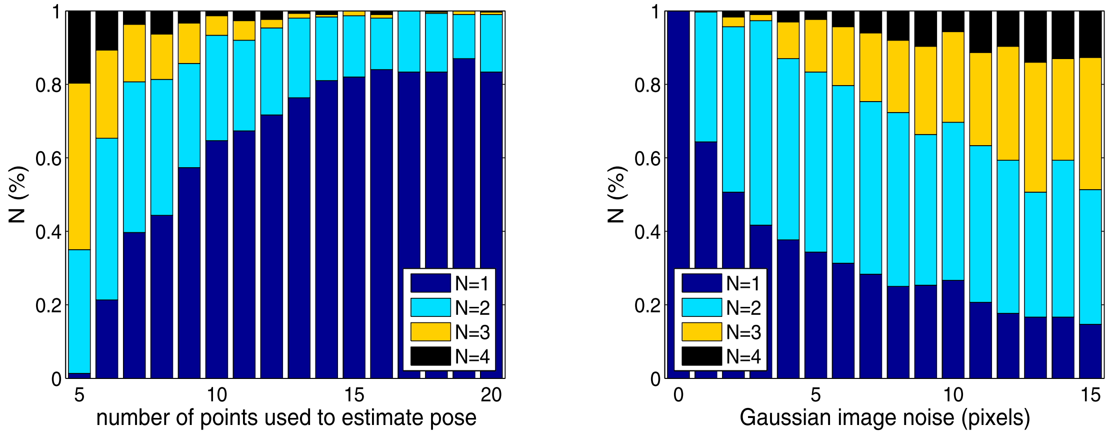

Fig. 4 Effective number N of null singular values in $\mathbf{M}^\top\mathbf{M}$. Each vertical bar represents the distributions of N for a total of 300 experiments. On the *left*, we plot the results for a fixed image noise of $\sigma = 10$ pixels and an increasing number of reference points, and the results on the *right* correspond to a fixed $n = 6$ number of reference points and increasing level of noise in the 2D projections

the number of input correspondences is (4) or (5), yielding only (8) or (10) equations, which is less than the number of unknowns.

In theory, given perfect data from at least six reference points imaged by a perspective camera, the dimension N of the null-space of $\mathbf{M}^\top\mathbf{M}$ should be exactly one because of the scale ambiguity. If the camera becomes orthographic instead of perspective, the dimension of the null space increases to four because changing the depths of the four control points would have no influence on where the reference points project. Figure 3 illustrates this behavior: For small focal lengths, $\mathbf{M}^\top\mathbf{M}$ has only one zero eigenvalue. However, as the focal length increases and the camera becomes closer to being orthographic, all four smallest eigenvalues approach zero. Furthermore, if the correspondences are noisy, the smallest of them will be tiny but not strictly equal to zero.

Therefore, we consider that the effective dimension N of the null space of $\mathbf{M}^\top\mathbf{M}$ can vary from 1 to 4 depending on the configuration of the reference points, the focal length of the camera, and the amount of noise, as shown in Fig. 4. In this section, we show that the fact that the distances between control points must be preserved can be expressed in terms of a small number of quadratic equations, which can be efficiently solved to compute $\{\beta_i\}_{i=1,\dots,N}$ for $N = 1, 2, 3$ and 4.

In practice, instead of trying to pick a value of N among the set $\{1, 2, 3, 4\}$, which would be error-prone if several eigenvalues had similar magnitudes, we compute solutions for all four values of N and keep the one that yields the smallest reprojection error

$$\mathrm{res} = \sum_i \mathrm{dist}^2\left(\mathbf{A}[\mathbf{R}|\mathbf{t}]\begin{bmatrix}\mathbf{p}_i^w \\ 1\end{bmatrix}, \mathbf{u}_i\right), \tag{9}$$

where $\mathrm{dist}(\widetilde{\mathbf{m}}, \mathbf{n})$ is the 2D distance between point $\mathbf{m}$ expressed in homogeneous coordinates, and point $\mathbf{n}$. This improves robustness without any noticeable computational penalty because the most expensive operation is the computation of $\mathbf{M}^\top\mathbf{M}$, which is done only once, and not the solving of a few quadratic equations. The distribution of values of N estimated in this way is depicted by Fig. 4.

We now turn to the description of the quadratic constraints we introduce for $N = 1, 2, 3$ and 4.

Case $N = 1$: We simply have $\mathbf{x} = \beta\mathbf{v}$. We solve for β by writing that the distances between control points as retrieved in the camera coordinate system should be equal to the ones computed in the world coordinate system when using the given 3D coordinates.

Let $\mathbf{v}^{[i]}$ be the sub-vector of $\mathbf{v}$ that corresponds to the coordinates of the control point $\mathbf{c}_i^c$. For example, $\mathbf{v}^{[1]}$ will represent the vectors made of the three first elements of $\mathbf{v}$. Maintaining the distance between pairs of control points $(\mathbf{c}_i, \mathbf{c}_j)$ implies that

$$\|\beta\mathbf{v}^{[i]} - \beta\mathbf{v}^{[j]}\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2. \tag{10}$$

Since the $\|\mathbf{c}_i^w - \mathbf{c}_j^w\|$ distances are known, we compute β in closed-form as

$$\beta = \frac{\sum_{\{i,j\}\in[1;4]} \|\mathbf{v}^{[i]} - \mathbf{v}^{[j]}\| \cdot \|\mathbf{c}_i^w - \mathbf{c}_j^w\|}{\sum_{\{i,j\}\in[1;4]} \|\mathbf{v}^{[i]} - \mathbf{v}^{[j]}\|^2}. \tag{11}$$

Case $N = 2$: We now have $\mathbf{x} = \beta_1\mathbf{v}_1 + \beta_2\mathbf{v}_2$, and our distance constraints become

$$\|(\beta_1\mathbf{v}_1^{[i]} + \beta_2\mathbf{v}_2^{[i]}) - (\beta_1\mathbf{v}_1^{[j]} + \beta_2\mathbf{v}_2^{[j]})\|^2 = \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2. \tag{12}$$

β_1 and β_2 only appear in the quadratic terms and we solve for them using a technique called "linearization" in cryptography, which was employed by Ansar and Daniilidis (2003) to estimate the point depths. It involves solving a linear system in $[\beta_{11}, \beta_{12}, \beta_{22}]^\top$ where $\beta_{11} = \beta_1^2, \beta_{12} = \beta_1\beta_2$, $\beta_{22} = \beta_2^2$. Since we have four control points, this produces a linear system of six equations in the β_{ab} that we write as:[3]

$$\mathbf{L}\boldsymbol{\beta} = \boldsymbol{\rho}, \tag{13}$$

where $\mathbf{L}$ is a 6×3 matrix formed with the elements of $\mathbf{v}_1$ and $\mathbf{v}_2$, $\boldsymbol{\rho}$ is a 6-vector with the squared distances $\|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$, and $\boldsymbol{\beta} = [\beta_{11}, \beta_{12}, \beta_{22}]^\top$ is the vector of unknowns. We solve this system using the pseudoinverse of $\mathbf{L}$ and choose the signs for the β_a so that all the $\mathbf{p}_i^c$ have positive z coordinates.

This yields β_1 and β_2 values that can be further refined by using the formula of (11) to estimate a common scale β so that $\mathbf{c}_i^c = \beta(\beta_1\mathbf{v}_1^{[i]} + \beta_2\mathbf{v}_2^{[i]})$.

Case $N = 3$: As in the $N = 2$ case, we use the six distance constraints of (12). This yields again a linear system $\mathbf{L}\boldsymbol{\beta} = \boldsymbol{\rho}$, although with larger dimensionality. Now $\mathbf{L}$ is a square 6×6 matrix formed with the elements of $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$, and $\boldsymbol{\beta}$ becomes the 6D vector $[\beta_{11}, \beta_{12}, \beta_{13}, \beta_{22}, \beta_{23}, \beta_{33}]^\top$. We follow the same procedure as before, except that we now use the inverse of $\mathbf{L}$ instead of its pseudo-inverse.

Case $N = 4$: We now have four β_a unknowns and, in theory, the six distance constraints we have been using so far should still suffice. Unfortunately, the linearization procedure treats all 10 products $\beta_{ab} = \beta_a\beta_b$ as unknowns and there are not enough constraints anymore. We solve this problem using a *relinearization* technique (Kipnis and Shamir 1999) whose principle is the same as the one we use to determine the control points coordinates.

The solution for the β_{ab} is in the null space of a first homogeneous linear system made from the original constraints. The correct coefficients are found by introducing new quadratic equations and solving them again by linearization, hence the name "relinearization". These new quadratic equations are derived from the fact that we have, by commutativity of the multiplication

$$\beta_{ab}\beta_{cd} = \beta_a\beta_b\beta_c\beta_d = \beta_{a'b'}\beta_{c'd'}, \tag{14}$$

where $\{a', b', c', d'\}$ represents any permutation of the integers $\{a, b, c, d\}$.

[3] We use the indices a and b for the β's in order to differentiate from the indices i and j used for the 3D points.

3.4 The Planar Case

In the planar case, that is, when the moment matrix of the reference points has one very small eigenvalue, we need only three control points. The dimensionality of $\mathbf{M}$ is then reduced to $2n \times 9$ with 9D eigenvectors $\mathbf{v}_i$, but the above equations remain mostly valid. The main difference is that the number of quadratic constraints drops from 6 to 3. As a consequence, we need use of the relinearization technique introduced in the $N = 4$ case of the previous section for $N \geq 3$.

4 Efficient Gauss-Newton Optimization

We will show in the following section that our closed-form solutions are more accurate than those produced by other state-of-the-art non-iterative methods. Our algorithm also runs much faster than the best iterative one we know of Lu et al. (2000) but can be slightly less accurate, especially when the iterative algorithm is provided with a good initialization. In this section, we introduce a refinement procedure designed to increase the accuracy of our solution at very little extra computational cost. As can be seen in Figs. 1 and 2, computing the solution in closed form and then refining it as we suggest here yields the same accuracy as our reference method (Lu et al. 2000), but still much faster.

We refine the four values $\boldsymbol{\beta} = [\beta_1, \beta_2, \beta_3, \beta_4]^\top$ of the coefficients in (8) by choosing the values that minimize the change in distance between control points. Specifically, we use Gauss-Newton algorithm to minimize

$$\mathrm{Error}(\boldsymbol{\beta}) = \sum_{(i,j)\ \text{s.t.}\ i<j} \left(\|\mathbf{c}_i^c - \mathbf{c}_j^c\|^2 - \|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2 \right), \tag{15}$$

with respect $\boldsymbol{\beta}$. The distances $\|\mathbf{c}_i^w - \mathbf{c}_j^w\|^2$ in the world coordinate system are known and the control point coordinates in camera reference are expressed as a function of the $\boldsymbol{\beta}$ coefficients as

$$\mathbf{c}_i^c = \sum_{j=1}^{4} \beta_j \mathbf{v}_j^{[i]}. \tag{16}$$

Since the optimization is performed only over the four β_i coefficients, its computational complexity is independent of the number of input 3D-to-2D correspondences. This yields fast and constant time convergence since, in practice, less than 10 iterations are required. As a result, the computational burden associated to this refinement procedure is almost negligible as can be observed in Fig. 2. In fact, the time required for the optimization may be considered as constant, and hence, the overall complexity of the closed-form solution and Gauss-Newton remains linear with the number of input 3D-to-2D correspondences.

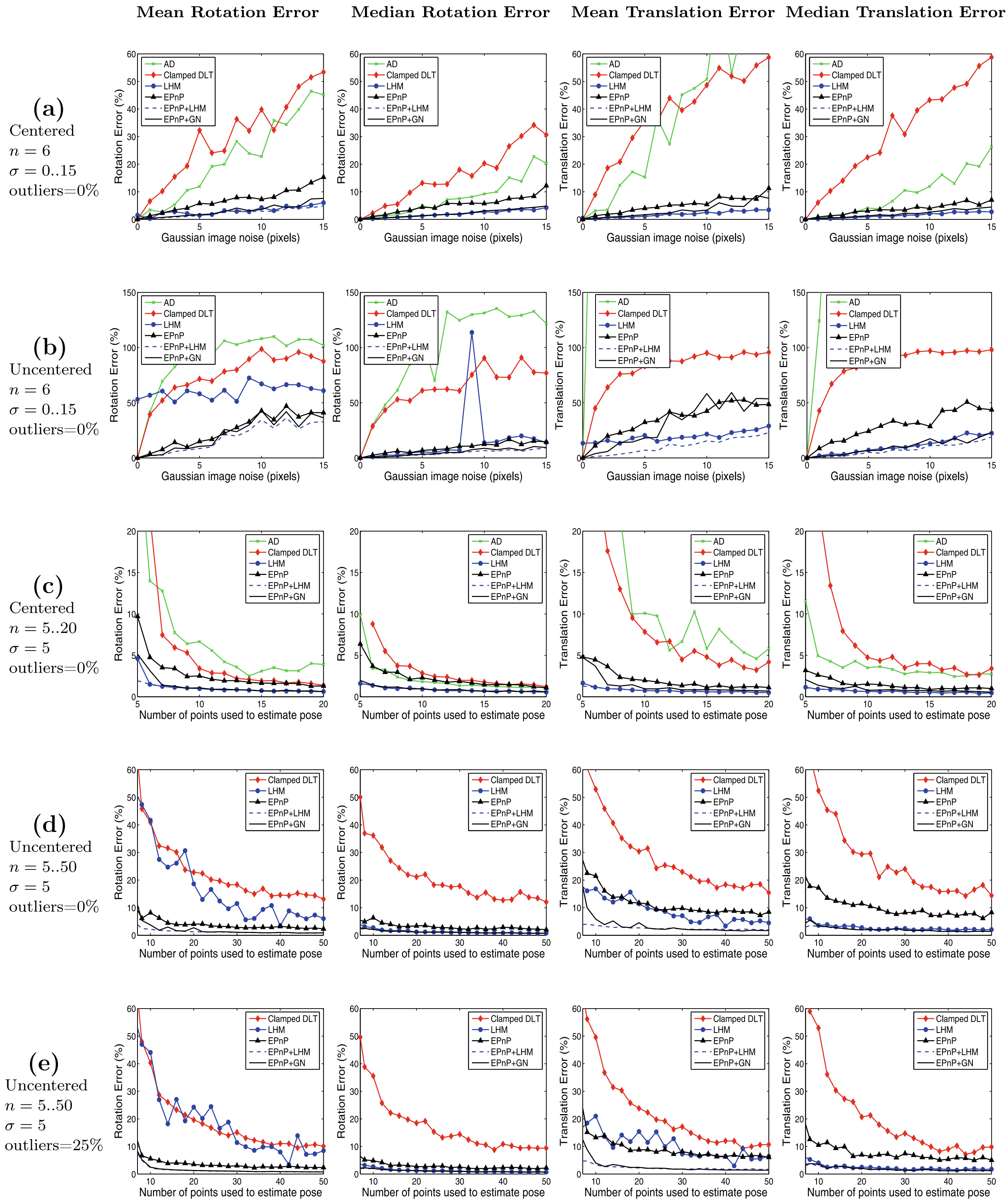

Fig. 5 (Color online) Non Planar case. Mean and median rotation and translation errors for different experiments

5 Results

We compare the accuracy and speed of our approach against that of state-of-the-art ones, both on simulated and real image data.

5.1 Synthetic Experiments

We produced synthetic 3D-to-2D correspondences in a 640 × 480 image acquired using a virtual calibrated camera with an effective focal length of $f_u = f_v = 800$ and a

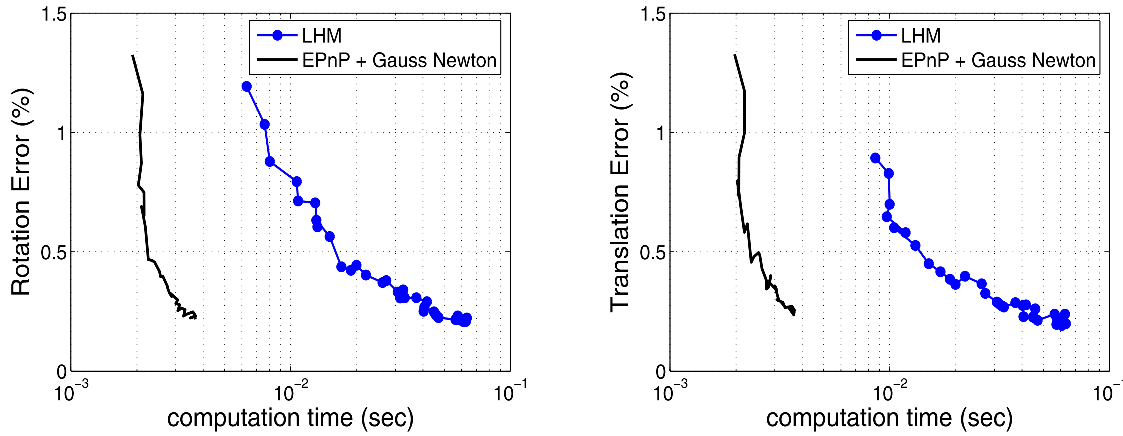

Fig. 6 Median error as a function of required amount of computation, itself a function of the number of points being used, for our approach and the LHM iterative method (Lu et al. 2000)

principal point at $(u_c, v_c) = (320, 240)$. We generated different sets for the input data. For the *centered* data, the 3D reference points were uniformly distributed into the x,y,z interval $[-2, 2] \times [-2, 2] \times [4, 8]$. For the *uncentered* data, the ranges were modified to $[1, 2] \times [1, 2] \times [4, 8]$. We also added Gaussian *noise* to the corresponding 2D point coordinates, and considered a percentage of *outliers*, for which the 2D coordinate was randomly selected within the whole image.

Given the true camera rotation $\mathbf{R}_{true}$ and translation $\mathbf{t}_{true}$, we computed the relative error of the estimated rotation $\mathbf{R}$ by $E_{rot}(\%) = \|\mathbf{q}_{true} - \mathbf{q}\|/\|\mathbf{q}\|$, where $\mathbf{q}$ and $\mathbf{q}_{true}$ are the normalized quaternions corresponding to the rotation matrices. Similarly, the relative error of the estimated translation $\mathbf{t}$ is determined by $E_{trans}(\%) = \|\mathbf{t}_{true} - \mathbf{t}\|/\|\mathbf{t}\|$.

All the plots discussed in this section were created by running 300 independent MATLAB simulations. To estimate running times, we ran the code 100 time for each example and recorded the average run time.

5.1.1 The Non-Planar Case

For the non-planar case, we compared the accuracy and running times of our algorithm, which we denote as EPnP, and EPnP + GN when it was followed by the optimization procedure described above, to: *AD*, the non-iterative method of Ansar and Daniilidis (2003); *Clamped DLT*, the DLT algorithm after clamping the internal parameters with their known values; *LHM*, the Lu's et al. (2000) iterative method initialized with a weak perspective assumption; EPnP + LHM, Lu's et al. algorithm initialized with the output of our algorithm.

On Fig. 1, we plot the rotational errors produced by the three non-iterative algorithms, and the three iterative ones as a function of noise when using $n = 6$ points. We use the boxplot representation,[4] where each column depicts the distribution of the errors for the 300 different simulations. A concise way to summarize the boxplot results is to plot both the mean and median results for all simulations: The difference between the mean and the median mainly comes from the high errors represented as red crosses in the boxplots. The greater it is, the less stable the method. This is shown in Fig. 5a, where in addition to the rotation error we also plot the translation error. The closed form solution we propose is consistently more accurate and stable than the other non-iterative ones, especially for large amounts of noise. It is only slightly less accurate than the LHM iterative algorithm. When the Gauss-Newton optimization is applied the accuracy of our method becomes then similar to that of LHM and, as shown in Fig. 5b, it even performs better when instead of using well spread data as in the previous case, we simulate data that covers only a small fraction of the image.

In Fig. 5c, we plot the errors as a function of the number of reference points, when the noise is fixed to $\sigma = 5$. Again, EPnP performs better than the other non-iterative techniques and very nearly as well as LHM. It even represents a more stable solution when dealing with the uncentered data of Fig. 5d and data which includes outliers, as in Fig. 5e. Note that in all the cases where LHM does not converge perfectly, the combination EPnP + LHM provides accurate results, which are similar to the EPnP + GN solution we propose. In the last two graphs, we did not compare the performance of AD, because this algorithm does not normalize the 2D coordinates, and hence, cannot deal well with uncentered data.

[4]The boxplot representation consists of a box denoting the first $Q1$ and third $Q3$ quartiles, a horizontal line indicating the median, and a dashed vertical line representing the data extent taken to be $Q3 + 1.5(Q3 - Q1)$. The red crosses denote points lying outside of this range.

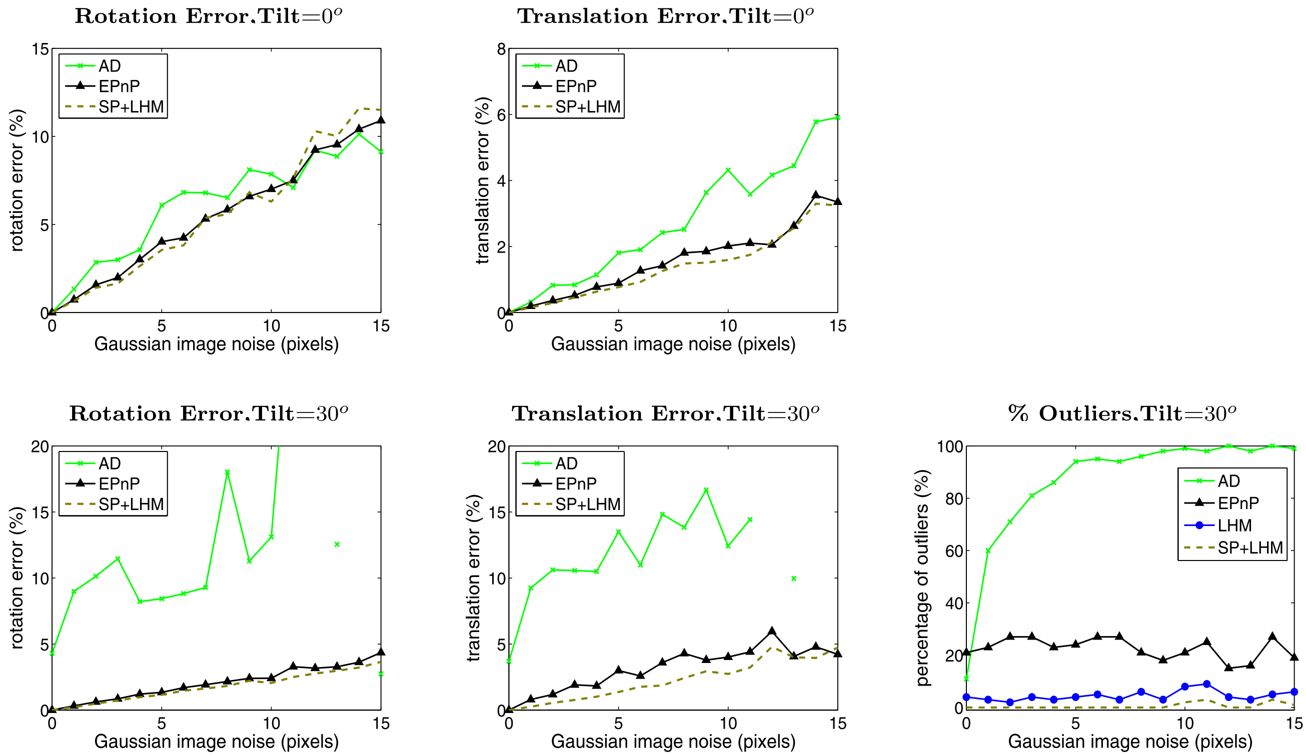

Fig. 7 Planar case. Errors as a function of the image noise when using $n = 6$ reference points. For Tilt $= 0°$, there are no pose ambiguities and the results represent the mean over all the experiments. For Tilt $= 30°$, the errors are only averaged among the solutions not affected by the pose ambiguity. The *right-most figure* represents the number of solutions considered as outliers, which are defined as those for which the average error in the estimated 3D position of the reference points is larger than a threshold

As shown in Fig. 2, the computational cost of our method grows linearly with the number of correspondences and remains much lower than all the others. It even compares favorably to clamped DLT, which is known to be fast. As shown in Fig. 6, EPnP + GN requires about a twentieth of the time required by LHM to achieve similar accuracy levels. Although the difference becomes evident for a large number of reference points, it is significant even for small numbers. For instance, for $n = 6$ points, our algorithm is about 10 times faster than LHM, and about 200 times faster than AD.

5.1.2 The Planar Case

Schweighofer and Pinz (2006) prove that when the reference points lie on a plane, camera pose suffers from an ambiguity that results in significant instability. They propose a new algorithm for resolving it and refine the solution using Lu's et al. (2000) method. Hereafter, we will refer to this combined algorithm as SP + LHM, which we will compare against EPnP, AD, and LHM. We omit Clamped DLT because it is not applicable in the planar case. We omit as well the EPnP + GN, because for the planar case the closed-form solution for the non-ambiguous cases was already very accurate, and the Gauss-Newton optimization could not help to resolve the ambiguity in the rest of cases.

Figure 7 depicts the errors as a function of the image noise, when $n = 10$ and for reference points lying on a plane with tilt of either 0 or 30 degrees. To obtain a fair comparison we present the results as was done in Schweighofer and Pinz (2006) and the errors are only averaged among those solutions not affected by the pose ambiguity. We also report the percentage of solutions which are considered as outliers. When the points are lying on a frontoparallel plane, there is no pose ambiguity and all the methods have a similar accuracy, with no outliers for all the methods, as shown in the first row of Fig. 7. The pose ambiguity problem appears only for inclined planes, as shown by the bottom-row graphs of Fig. 7. Note that for AD, the number of outliers is really large, and even the errors for the inliers are considerable. EPnP and LHM produce a much reduced number of outliers and similar results accuracy for the inliers. As before, the SP + LHM method computes the correct pose for almost all the cases. Note that we did not plot the errors for LHM, because when considering only the correct poses, it is the same as SP + LHM.

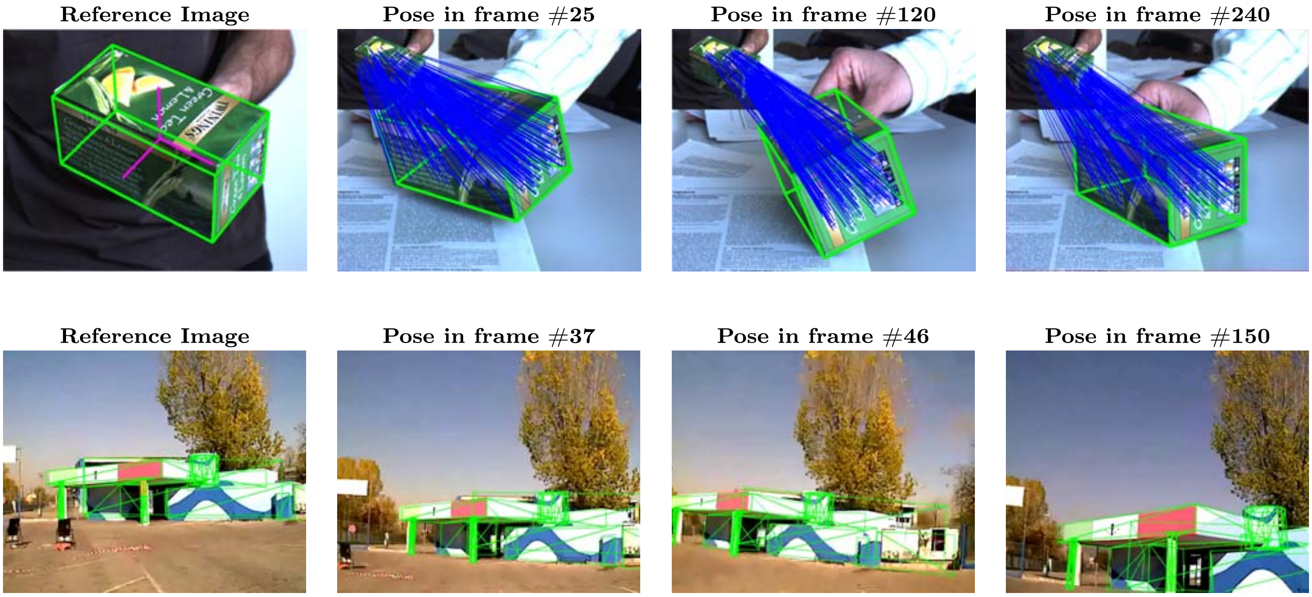

Fig. 8 Real images. *Top*. *Left*: Calibrated reference image. *Right*: Reprojection of the model of the box on three video frames. The camera pose has been computed using the set of correspondences depicted by the thin blue lines. *Bottom*. *Left*: Calibrated reference image. *Right*: Reprojection of the building model on three video frames. The registration remains accurate even when the target object is partially occluded

As in the non-planar case, the EPnP solution proposed here is much faster than the others. For example for $n = 10$ and a tilt of 30°, our solution is about 200 times faster than AD, 30 times faster than LHM, even though the MATLAB code for the latter is not optimized.

5.2 Real Images

We tested our algorithm on noisy correspondences, that may include erroneous ones, obtained on real images with our implementation of the keypoint recognition method of (Lepetit and Fua 2006). Some frames of two video sequences are shown in Fig. 8. For each case, we trained the method on a calibrated reference image of the object to be detected, for which the 3D model was known. These reference images are depicted in Fig. 8-left. At run time, the method generates about 200 correspondences per image. To filter out the erroneous ones, we use RANSAC on small subsets made of 7 correspondences from which we estimate the pose using our PnP method. This is effective because, even though our algorithm is designed to work with a large number of correspondences, it is also faster than other algorithms for small numbers of points, as discussed above. Furthermore, once the set of inliers has been selected, we use all of them to refine the camera pose. This gives a new set of inliers and the estimation is iterated until no additional inliers are found. Figure 8-right shows different frames of the sequences, where the 3D model has been reprojected using the retrieved pose.

6 Conclusion

We have proposed an $O(n)$ non-iterative solution to the PnP problem that is faster and more accurate than the best current techniques. It is only slightly less accurate than one the most recent iterative ones (Lu et al. 2000) but much faster and more stable. Furthermore, when the output of our algorithm is used to initialize a Gauss-Newton optimization, the precision is highly improved with a negligible amount of additional time.

Our central idea—expressing the 3D points as a weighted sum of four virtual control points and solving in terms of their coordinates—is very generic. We demonstrated it in the context of the PnP problem but it is potentially applicable to problems ranging from the estimation of the Essential matrix from a large number of points for Structure-from-Motion applications (Stewènius et al. 2006) to shape recovery of deformable surfaces. The latter is particularly promising because there have been many approaches to parameterizing such surfaces using control points (Sederberg and Parry 1986; Chang and Rockwood 1994), which would fit perfectly into our framework and allow us to recover not only pose but also shape. This is what we will focus on in future research.

Acknowledgements The authors would like to thank Dr. Adnan Ansar for providing the code of his PnP algorithm. This work was supported in part by the Swiss National Science Foundation and by funds of the European Commission under the IST-project 034307 DYVINE (Dynamic Visual Networks).

References

Abdel-Aziz, Y. I., & Karara, H. M. (1971). Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In *Proc. ASP/UI symp. close-range photogrammetry* (pp. 1–18).

Ansar, A., & Daniilidis, K. (2003). Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(5), 578–589.

Arun, K. S., Huang, T. S., & Blostein, S. D. (1987). Least-squares fitting of two 3-D points sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *9*(5), 698–700.

Chang, Y., & Rockwood, A. P. (1994). A generalized de Casteljau approach to 3D free-form deformation. In *ACM SIGGRAPH* (pp. 257–260).

DeMenthon, D., & Davis, L. S. (1995). Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, *15*, 123–141.

Dhome, M., Richetin, M., & Lapreste, J.-T. (1989). Determination of the attitude of 3d objects from a single perspective view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(12), 1265–1278.

Fiore, P. D. (2001). Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(2), 140–148.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications ACM*, *24*(6), 381–395.

Gao, X. S., Hou, X. R., Tang, J., & Cheng, H. F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(8), 930–943.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations*. *Johns Hopkins studies in mathematical sciences*. Baltimore: Johns Hopkins Press.

Haralick, R. M., Lee, D., Ottenburg, K., & Nolle, M. (1991). Analysis and solutions of the three point perspective pose estimation problem. In *Conference on computer vision and pattern recognition* (pp. 592–598).

Hartley, R., & Zisserman, A. (2000). *Multiple view geometry in computer vision*. Cambridge: Cambridge University Press.

Horaud, R., Conio, B., Leboulleux, O., & Lacolle, B. (1989). An analytic solution for the perspective 4-point problem. *Computer Vision, Graphics, and Image Processing*, *47*(1), 33–44.

Horaud, R., Dornaika, F., & Lamiroy, B. (1997). Object pose: The link between weak perspective, paraperspective, and full perspective. *International Journal of Computer Vision*, *22*(2), 173–189.

Horn, B. K. P., Hilden, H. M., & Negahdaripour, S. (1988). Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America*, *5*(7), 1127–1135.

Kipnis, A., & Shamir, A. (1999). Cryptanalysis of the HFE public key cryptosystem by relinearization. In *Advances in cryptology—CRYPTO'99* (Vol. 1666/1999, pp. 19–30). Berlin: Springer.

Kumar, R., & Hanson, A. R. (1994). Robust methods for estimating pose and a sensitivity analysis. *Computer Vision and Image Understanding*, *60*(3), 313–342.

Lepetit, V., & Fua, P. (2006). Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(9), 1465–1479.

Lowe, D. G. (1991). Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(5), 441–450.

Lu, C.-P., Hager, G. D., & Mjolsness, E. (2000). Fast and globally convergent pose estimation from video images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(6), 610–622.

McGlove, C., Mikhail, E., & Bethel, J. (Eds.) (2004). *Manual of photogrametry*. *American society for photogrammetry and remote sensing* (5th edn.).

Moreno-Noguer, F., Lepetit, V., & Fua, P. (2007). Accurate non-iterative $o(n)$ solution to the pnp problem. In *IEEE international conference on computer vision*. Rio de Janeiro, Brazil.

Oberkampf, D., DeMenthon, D., & Davis, L. S. (1996). Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, *63*, 495–511.

Quan, L., & Lan, Z. (1999). Linear N-point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *21*(7), 774–780.

Schweighofer, G., & Pinz, A. (2006). Robust pose estimation from a planar target. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(12), 2024–2030.

Sederberg, T. W., & Parry, S. R. (1986). Free-form deformation of solid geometric models. *ACM SIGGRAPH*, *20*(4).

Skrypnyk, I., & Lowe, D. G. (2004). Scene modelling, recognition and tracking with invariant image features. In *International symposium on mixed and augmented reality* (pp. 110–119). Arlington, VA.

Stewènius, H., Engels, C., & Nister, D. (2006). Recent developments on direct relative orientation. *International Society for Photogrammetry and Remote Sensing*, *60*, 284–294.

Triggs, B. (1999). Camera pose and calibration from 4 or 5 known 3D points. In *International conference on computer vision* (pp. 278–284).

Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(4).