Globally Optimal $O(n)$ Solution to the PnP Problem for General Camera Models[*]

Gerald Schweighofer and Axel Pinz
Institute of Electrical Measurement and Measurement Signal Processing,
Graz University of Technology [†]

Abstract

We present a novel and fast algorithm to solve the Perspective-n-Point problem. The PnP problem - estimating the pose of a calibrated camera based on measurements and known 3D scene, is recasted as a minimization problem of the Object Space Cost. Instead of limiting the algorithm to perspective cameras, we use a formulation for general camera models. The minimization problem, together with a quaternion based representation of the rotation, is transferred into a semi definite positive program (SDP). This transfer is done in $O(n)$ time and leads to an SDP of constant size. The solution of the SDP is a global minimizer of the PnP problem, which can be estimated in less than 0.15 seconds for 100 points.

1 Introduction

Estimating the pose of a calibrated camera has lots of applications in Computer Vision, Robotics and Augmented Reality. Previous attempts solve that problem either for a specific, small number of points (three points [4] or four points [9]). Others try to solve for an arbitrary number of points using iterative methods [12, 3, 10, 18], which minimize a given cost function. One drawback of such methods is the computational burden. To overcome that burden, non-iterative methods [16, 6, 2, 11] have been developed. These methods reformulate the problem in a way that it can be solved by a single (large) equation system ($O(n^2)$ [16] or $O(n^8)$ [2]).

Limitations for all of these methods are either (i) the fact that they gain speed by approximating the cost function (non-iterative ones), or (ii) lack in reaching a global minimum (iterative ones).

These limitations have led to a search for global optimization methods. Recent results were obtained by Agarwal et al. [1]. They proposed to use a *branch and bound* algorithm together with *second order cone* programs (SOCP) to estimate global solutions for triangulation and camera pose estimation. For pose estimation they modeled the camera as a 4×3 matrix without accounting for the constraints on the rotation matrix. The achieved runtimes vary from 42 seconds (6 points) to 250 seconds (100 points). Using the constraints of the rotation, Hartley et al. [8] proposed a *branch and bound* algorithm which solves iteratively an approximation of the original problem. Due to properties of

[*]This work was supported by the Austrian Science Foundation (FWF, project S9103-N13).
[†]E-Mail: gerald.schweighofer@tugraz.at, axel.pinz@tugraz.at

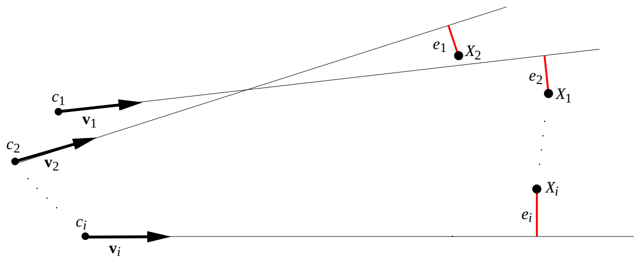


Figure 1: Object Space Cost e_i for General Camera Model.

the approximated rotation, this led to a SOCP too. In [13], Olsson et al. used the same *branch and bound* algorithm, but quaternions as representation for rotations. Reported timings range from 1.5 to 10 minutes (for 10 points) and from 2 to 14 minutes (for 4 points) depending on the approximation of the rotation.

In this paper we propose a non-iterative fast global optimal solution to the PnP problem. This is achieved based on the following three ideas: First, the problem is recasted as a minimization problem using the object space cost. This cost is used in many pose estimation algorithms [10, 18, 5]. In [11] and [18], comparisons of different pose estimation algorithms were performed. The conclusions of both papers are, that minimizing object space cost gives the best accuracy of the estimated pose. Second, the minimization can be transformed to a semi definite positive program (SDP) using the theory of sum of squares [15]. Third, this SDP can be efficiently solved, using publicly available tools like SeDuMi [19].

One main difference of the proposed approach compared to the other global optimization algorithms is the fact, that the time consuming *branch and bound* algorithm is avoided. Only one SDP need to be solved to estimate the global solution. This can be achieved in less then 0.15 seconds (for up to 100 points) and less than 0.3 seconds (for up to 1000 points) using *MATLAB* on a *Intel Core 2 Duo 1.6 GHz*.

In the remainder of the paper, we first introduce the general camera model and its appropriate cost function. Using that cost function we formulate the PnP problem as a minimization problem. After that we give a short introduction of sum of squares optimization, and show then how the PnP problem can be globally solved using that theory. Finally, we conclude with experiments.

2 Problem formulation - PnP for a General Camera Model

We follow the definition of a General Camera Model (GCM) described in [7], and use the same notation as [17]. Instead of measuring a position inside a sensor, a direction is measured. This direction is represented by a tuple ($\mathbf{c}$,$\mathbf{v}$). Based on these measurements the **Object Space Error for General Camera Models** is defined as [17]:

$$e_i(R,\mathbf{t},\mathbf{X}_i) = \|(I-V_i)(R\mathbf{X}_i+\mathbf{t}-\mathbf{c}_i)\|^2 \quad \text{with} \quad V_i = \frac{\mathbf{v}_i\mathbf{v}_i^T}{\mathbf{v}_i^T\mathbf{v}_i}. \tag{1}$$

As an example the costs e_i for points $\mathbf{X}_i$ are shown in Figure 1. The cost e_i measures the distance between a world point $\mathbf{X}_i$ and the projection of this point onto the *line of sight*. The index i represents one of n_i points.

Given world points $\mathbf{X}_i$ and their measurement in a calibrated camera ($\mathbf{v}_i$ and $\mathbf{c}_i$), the problem of pose estimation is defined as finding the pose (R and $\mathbf{t}$), for which the sum of all costs is a minimum:

$$\arg\min_{R,\mathbf{t}} E(R,\mathbf{t}) = \arg\min_{R,\mathbf{t}} \sum_{i=1}^{n_i} e_i(R,\mathbf{t},\mathbf{X}_i) = \arg\min_{R,\mathbf{t}} \sum_{i=1}^{n_i} \|(I-V_i)(R\mathbf{X}_i+\mathbf{t}-\mathbf{c}_i)\|^2 \quad \text{subject to } R \in SO(3) \tag{2}$$

Solving for an optimal translation $\mathbf{t}_{opt}$ is given by differentiating (2) w.r.t. the translation $\mathbf{t}$ and setting the result equal to zero [10, 5, 18]:

$$\mathbf{t}_{\text{opt}} = \left(\sum_i Q_i\right)^{-1} \sum_i Q_i(R\mathbf{X}_i+\mathbf{c}_i) \qquad \text{with} \qquad Q_i = (I-V_i)^T(I-V_i)\,. \tag{3}$$

Let us introduce the operator $C(\mathbf{X})$:

$$C(\mathbf{X}) = \begin{bmatrix} \mathbf{X}^T & 0_{1\times 3} & 0_{1\times 3} \\ 0_{1\times 3} & \mathbf{X}^T & 0_{1\times 3} \\ 0_{1\times 3} & 0_{1\times 3} & \mathbf{X}^T \end{bmatrix}, \tag{4}$$

where $0_{1\times 3}$ is a zero matrix of size 1×3. Using that operator and a vector based representation of the rotation matrix $\mathbf{r}(R) = [\mathbf{r}_1^T, \mathbf{r}_2^T, \mathbf{r}_3^T]^T$ ($R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$), the optimal translation $\mathbf{t}_{opt}$ is given by

$$\mathbf{t}_{\text{opt}} = T_{3\times 10}\begin{bmatrix} \mathbf{r} \\ 1 \end{bmatrix} \qquad \text{with} \qquad T_{3\times 10} = \left(\sum_i Q_i\right)^{-1} \sum_i Q_i\,[C(\mathbf{X}_i) \mid \mathbf{c}_i]. \tag{5}$$

Using this result and the vector based notation of the rotation $\mathbf{r}$, the original problem (Pose Estimation for a General Camera) (2) can be written as:

$$\begin{aligned} &\arg\min_{\mathbf{r}}\ \mathbf{r}^T M_r \mathbf{r} + M_c \mathbf{r} + M_{cc} \\ &\text{subject to } R \in SO(3), \end{aligned} \tag{6}$$

with

$$\begin{aligned} M &= \sum_i \left([C(\mathbf{X}_i) \mid \mathbf{c}_i] + T_{3\times 10}\right)^T Q_i \left([C(\mathbf{X}_i) \mid \mathbf{c}_i] + T_{3\times 10}\right) = \\ &= \begin{bmatrix} M_r & \frac{1}{2}M_c \\ \frac{1}{2}M_c^T & M_{cc} \end{bmatrix}. \end{aligned} \tag{7}$$

What makes the minimization problem in (6) complicated is the constraint "$R \in SO(3)$", which restricts the matrix R to be a valid rotation matrix.

In case of a single perspective camera all measurement rays intersect in the optical center of that camera. Therefore, without loss of generality, all $\mathbf{c}_i = 0$, which results in $M_c = 0_{9\times 1}$ and $M_{cc} = 0$. Such a minimization problem ($\arg\min_{\mathbf{r}}\ \mathbf{r}^T M_r \mathbf{r}$) was solved by

Ess et. al [5] ignoring the constraint on the rotation. Ignoring the constraint leads to a singular value decomposition (SVD) algorithm. One main drawback of that approach is that the result of the SVD is not a valid rotation, and therefore a second SVD is needed to *normalize* the result. Still, the result of this algorithm is not the optimal solution of the problem, due to this two step approach.

Schweighofer and Pinz [18] solved the same problem (2) incorporating the constraint on the rotation matrix using an iterative approach. They found, that there can exist up to two minima and therefore an iterative approach can end up in a local minimum instead of the global minimum.

In this paper we solve (6) with the constraint $R \in SO(3)$, which leads to a **global solution**. We do that using recent results in global optimization, namely the theory of sum of squares (SOS). In the next section we give a short overview of how SOS can be used for global optimization.

2.1 Sum of Squares for global minimization

Finding a global minimizer of a function $f(x)$ could be replaced by

$$\begin{aligned} &\text{maximize } \gamma \\ &\text{subject to } f(x) - \gamma >= 0. \end{aligned} \tag{8}$$

In (8) one needs to show that the constraint is positive for all x. In general this problem is NP-complete [15].

An easier, and computationally more tractable way is to show that $f(x) - \gamma$ is a sum of squares. To start we begin with the definition of a sum of squares (SOS) [15]: A multivariate polynomial $p(x_1, \dots, x_n) = p(x)$ is a sum of squares if there exist polynomials $f_1(x), \dots, f_m(x)$ such that

$$p(x) = \sum_{i=1}^{m} f_i^2(x). \tag{9}$$

Using that definition it is clear that the polynomial $p(x) \geq 0 \; \forall x$. To show that a polynomial $p(x)$ is an SOS it is sufficient to show that there exists a semi definite positive matrix Q such that

$$p(x) = \mathbf{Z}^T(x) Q \mathbf{Z}(x), \tag{10}$$

where $\mathbf{Z}(x)$ is a vector of monomials of x. The non-negativity constraint in (8) is replaced by an SOS constraint

$$\begin{aligned} &\text{maximize } \gamma \\ &\text{subject to } f(x) - \gamma = \mathbf{Z}^T(x) Q \mathbf{Z}(x) \\ &\qquad\qquad Q \geq 0, \end{aligned} \tag{11}$$

which, by comparing coefficients, gives a system of linear equations ($A\mathbf{q} = \mathbf{b}$), and therefore results in an SDP

$$\begin{aligned} &\text{maximize } \gamma \\ &\text{subject to } A\mathbf{q} = \mathbf{b} \\ &\qquad\qquad Q \geq 0, \end{aligned} \tag{12}$$

which could be easily solved using tools like SeDuMi [19].
Let us assume we want to solve the constrained optimization problem:

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } g_i(x) \geq 0, \text{ i=1,\dots,M} \\ &\qquad h_j(x) = 0, \text{ j=1,\dots,N} \end{aligned} \tag{13}$$

It was shown in [15, 14], that an upper bound to the global minimum could be found using the *Positivstellensatz*:

$$f(x) - \gamma = \sigma_0(x) + \sum_j \lambda_j(x) h_j(x) + \sum_i \sigma_i(x) g_i(x) + \sum_{i1,i2} \sigma_{i1,i2}(x) g_{i1}(x) g_{i1}(x) + \dots \tag{14}$$

If one finds polynomials $\lambda_j(x)$ and SOSs $\sigma_i(x)$ according to (14), then γ is a lower bound to the optimization problem of (13). Maximizing γ gives a lower bound, which gets tighter when the degree of (14) is increased. If the degree of the right hand side of (14) is high enough, or the lower bound $\gamma = f(x)$, then γ is the global minimum.

2.2 Pose estimation using SOS

The minimization problem of (6) can be rewritten as a multivariate minimization problem in four variables. First, we parameterize the rotation vector $\mathbf{r}$ using unit quaternions $\mathbf{q} = [q_1, q_2, q_3, q_4]$:

$$\mathbf{r} = \begin{matrix} [q_1^2 + q_2^2 - q_3^2 - q_4^2, 2q_2q_3 + 2q_1q_4, 2q_2q_4 - 2q_1q_3, 2q_2q_3 - 2q_1q_4, q_1^2 + q_3^2 - q_2^2 - q_4^2, \\ 2q_3q_4 + 2q_1q_2, 2q_2q_4 + 2q_1q_3, 2q_3q_4 - 2q_1q_2, q_1^2 + q_4^2 - q_2^2 - q_3^2]^T. \end{matrix} \tag{15}$$

Using that representation for rotations, the constraint "$R \in SO(3)$" can be written in constraints on the quaternions: First, the norm of $\mathbf{q}$ must be one ($\|\mathbf{q}\|^2 = 1$) to represent a valid rotation. Second, the ambiguity of quaternions must be resolved: $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation, so we add the constraint $q_1 \geq 0$. Ignoring the last constraint would lead to two equally valid solutions, which would distract the SDP solver. With this quaternion based representation, the pose estimation problem of (6) is recasted as

$$\begin{aligned} &\text{minimize } f(q) \\ &\text{subject to } q_1 \geq 0 \\ &\qquad \|\mathbf{q}\|^2 - 1 = 0, \end{aligned} \tag{16}$$

with $f(q) = \mathbf{r}^T M_r \mathbf{r} + M_c \mathbf{r} + M_{cc}$. Using the Positivstellensatz (14) and SOS decomposition we obtain

$$\begin{aligned} &\text{maximize } \gamma \\ &\text{subject to } f(q) - \gamma - \lambda(\|\mathbf{q}\|^2 - 1) - \sigma q_1 \text{ is SOS} \\ &\qquad \sigma \text{ is SOS}, \end{aligned} \tag{17}$$

with

$$\sigma = \left[1, \mathbf{q}^T\right] C_1 \begin{bmatrix} 1 \\ \mathbf{q} \end{bmatrix} \text{ and } \qquad \lambda = \left[1, \mathbf{q}^T\right] C_2 \begin{bmatrix} 1 \\ \mathbf{q} \end{bmatrix}. \tag{18}$$

Here C_1 is a 5×5 matrix with unknown coefficients. To fulfill the constraint "σ is SOS" C_1 has to be a semi-definite positive matrix. The matrix C_2 is an upper triangular matrix of size 5×5 with 15 unknown coefficients. Finally, the minimization problem of (17) can be written as an SDP like the one in (12). Using a tool for SOS programming [15] we developed a script, which converts the pose estimation problem (6) (M, M_c and M_{cc}) into an SDP (A, $\mathbf{b}$ and $\mathbf{c}$). The matrix A, and the vectors $\mathbf{b}$ and $\mathbf{c}$ for the SDP are too large to be shown here (size of A is 266×70), therefore the *MATLAB* code which does the conversion is available at http://www.emt.tugraz.at/$\sim$pinz/code. The SDP is solved using SeDuMi [19].

2.3 Special Case: Planar Scene

In cases of a single perspective camera all measurement rays intersect in the optical center of that camera. Therefore, without loss of generality all $\mathbf{c}_i = 0$, which results in $M_c = 0_{9\times1}$ and $M_{cc} = 0$. If the scene points $\mathbf{X}_i$ are co-planar we can assume, again without loss of generality, that the points are located in the z-plane: $\mathbf{X}_i = [X_{xi}, X_{yi}, 0]^T$. Evaluating the matrix M with such points results in an M where the 3-rd, 6-th and 9-th columns and rows are equal to zero.

This results in two different solutions for the global optimum. Let us assume that one of the global optima is reached at $R_1 = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ and $\mathbf{t}_1$. Then the second global optimum is at $R_2 = [-\mathbf{r}_1, -\mathbf{r}_2, \mathbf{r}_3]$ and $-\mathbf{t}_1$. Therefore, the two statements are equivalent:

$$\text{Two global minima} \begin{array}{ll} R1 = [\ \ \mathbf{r}_1, \ \ \mathbf{r}_2, \mathbf{r}_3], & \mathbf{t}_1 \\ R2 = [-\mathbf{r}_1, -\mathbf{r}_2, \mathbf{r}_3], & -\mathbf{t}_1 \end{array} \equiv \text{Points are co-planar } X_{zi} = 0\ . \tag{19}$$

In such a situation SeDuMi [19] would not converge to a single solution. It reports that no single solution could be found. To overcome this limitation we need to add a further constraint which separates the solutions from each other. Since the rotations of the two solutions differ only in the sign, we decided to use the sign of the first element $r_{1_1} = q_1^2 + q_2^2 - q_3^2 - q_4^2$ to distinguish the solutions from each other. Solving both problems

$$\begin{array}{ll} \text{minimize } f(q) & \qquad \text{minimize } f(q) \\ \text{subject to } q_1 \geq 0 & \qquad \text{subject to } q_1 \geq 0 \\ \qquad {q_1}^2 + {q_2}^2 - {q_3}^2 - {q_4}^2 \geq 0 & \qquad\qquad {q_1}^2 + {q_2}^2 - {q_3}^2 - {q_4}^2 \leq 0 \\ \qquad \|\mathbf{q}\|^2 - 1 = 0. & \qquad\qquad \|\mathbf{q}\|^2 - 1 = 0. \end{array} \tag{20}$$

results in two optimal solutions. In the first one the points are in front of the camera and in the second one the points are behind the camera. Both solutions have the same cost (2).

If one knows that the points are exactly co-planar only one of the two minimization problems in (20) has to be solved, because the second solution can be obtained from the first one. If the scene is not exactly planar, or one does not know whether the scene is planar or not, both programs have to be solved. In such a case the 3-rd, 6-th and 9-th column and row of M are not exactly zero and therefore the statement of (19) does not hold anymore. In such cases both systems of (20) are solved and the best one is taken as the global optimal solution. The code for this special (planar) case (size of A is 292×70) is also available at http://www.emt.tugraz.at/$\sim$pinz/code.

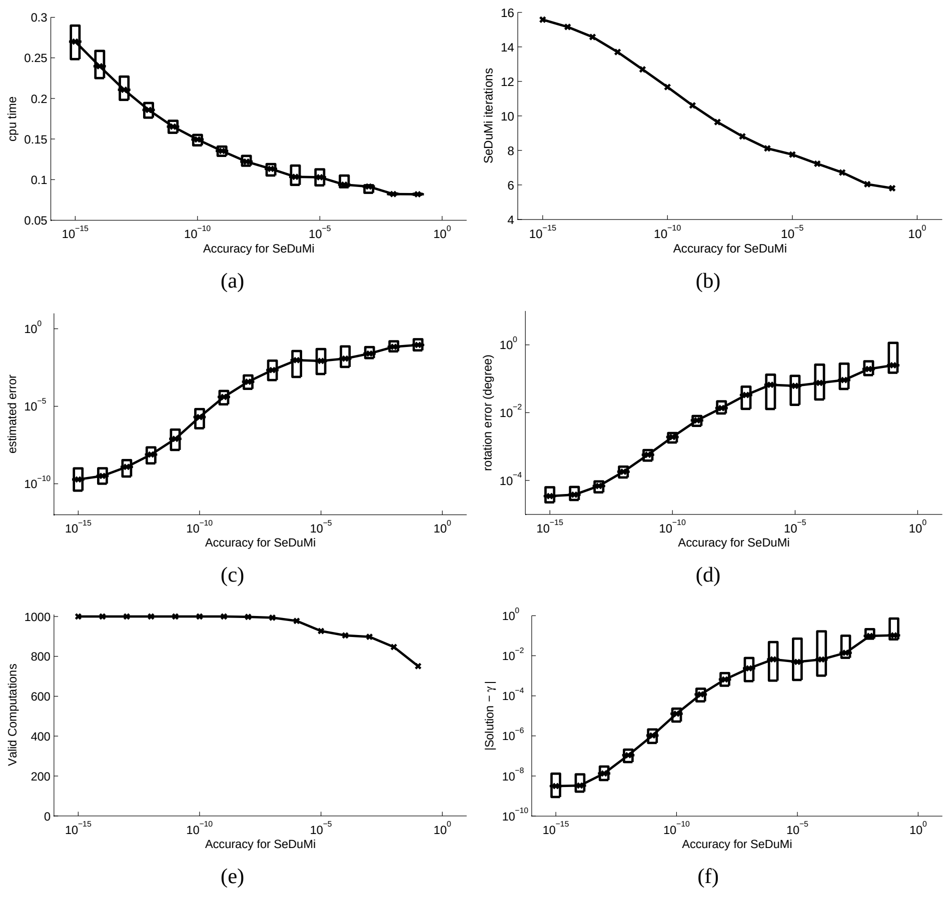


Figure 2: Numerical Results for global optimal pose estimation using SeDuMi. (a) consumed CPU time, (b) SeDuMi iterations, (c) error $E(R)$, (d) rotation error, (e) number of valid computations, (f) gap to global minimum.

3 Experiments

The first experiment was performed to analyze the numerical stability of the proposed algorithm. We generated 100 random points normally distributed in the x,y,z interval $[-0.5,0.5] \times [-0.5,0.5] \times [9.5,10.5]$. These points were observed with a general camera model ($\mathbf{c}_i$ were normally distributed in the x,y,z interval $[-0.5,0.5] \times [-0.5,0.5] \times [-0.5,0.5]$). From the measurements $\mathbf{v}_i$ we generate the matrices M_r, M_c and M_{cc} according to (7), which are then translated to an SDP program explained in section 2.2. The SDP program is then solved with SeDuMi [19]. SeDuMi is a publicly available iterative SDP solver. To stop the iterations one needs to select a stopping criterion. This criterion is based on the difference between successive iterations. We call this value *Accuracy of SeDuMi*. For different values of that accuracy, ranging from 10^{-1} to 10^{-15}, we repeated the experiment 1000 times.

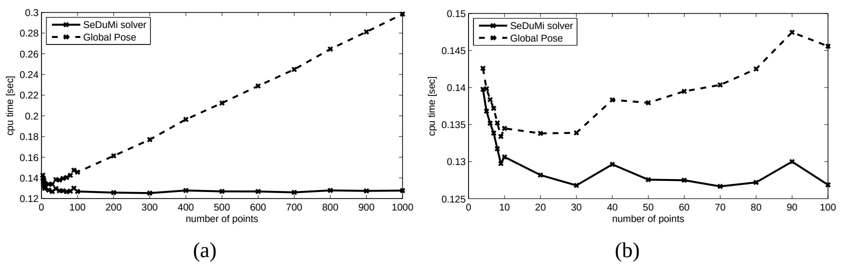


Figure 3: Consumed CPU time for different numbers of points. Computations are performed on a Intel Core 2 Duo 1.6 GHz, *MATLAB R2008a (64-bit), SeDuMi 1.2.* (a) 4 to 1000 points, (b) 4 to 100 points.

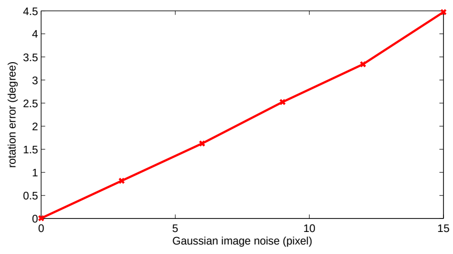
Fig.2 shows the median, the $1/4$ and $3/4$ quartiles. In Fig. 2(a) the computation time is reported, which is compared to the number of iterations SeDuMi needed to reach the preset accuracy in Fig. 2(b). To reach the highest accuracy only about 16 iterations need to performed and it takes only about 0.3 seconds to reach this accuracy.

In Fig. 2(c) we show the estimated error of the pose estimation problem $E(R)$ (2) and in Fig. 2(d) we show the rotation error. The rotation error is the difference of the estimated rotation w.r.t. ground truth. There is a linear relationship in the interval 10^{-7} to 10^{-13}. If we increase the accuracy for SeDuMi above 10^{-13} we reach the accuracy of the used data-types inside the implementation. If the accuracy is set too low (larger than 10^{-7}) SeDuMi failed to converge to a solution.

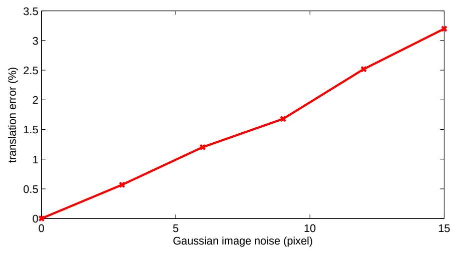
This failure of SeDuMi is also reported in Fig. 2(e). Here we see how often SeDuMi reported a valid solution for the 1000 experiments. For an accuracy better than 10^{-9} always a valid solution was reported. Therefore, we selected 10^{-10} as a good value for further experiments. Finally, in Fig. 2(f) we see how far the obtained solution is away from γ. This is a measure of how close we are to the global optimum.

The second experiment was performed to estimate the runtime of the algorithm. We set the accuracy for SeDuMi to 10^{-10} and repeated the above experiment for different numbers of points (from 4 to 1000). In Fig. 3(a) we see the spent CPU time. The dashed line shows the amount of CPU time for the complete algorithm (generating matrices M, M_c, M_{cc}, computing the matrices for the SDP A, b, c, solving the SDP), whereas the solid line shows the time, which was consumed by the SDP solver. In general we can say that the algorithm is of $O(n)$ time complexity, while reaching the global optimum. The time needed to solve the SDP problem is close to constant. This is because the size of the SDP problem is constant and does not vary with the number of points. On the other hand, the computation of the matrices M, M_c and M_{cc} depends linearly on the number of points.

A close look into the range of 4 to 100 points in Fig. 3(b) shows that solving the SDP requires more time for fewer points. This could be explained as follows: For three points, there are up to four valid global optimal solutions [4]. In that case the algorithm fails, due to the fact that it is only valid if there is one solution. If one adds a fourth point the up to four solutions are now disturbed by that point, but there are local minima close to these solutions. For the SDP solver, it takes a lot of time until it *detects* which of the *local possible* solutions is the global one. Adding more and more points makes it easier for the



(a)



(b)

Figure 4: Accuracy of the proposed algorithm for different noise levels. (a) rotation error. (b) translation error.

SDP solver to *decide* which one is the correct one.

In the third experiment we show the behavior of the algorithm w.r.t. noisy measurements for a central camera ($\mathbf{c}_i = 0$). We repeated the previous experiment for different Gaussian noise levels in the measurements. We notice a linear relationship between noise level and the accuracy of the rotation in degrees Fig. 4(a) and for the the accuracy of the translation (in percent: $100 \times |t - t_{\text{ground truth}}|/|t_{\text{ground truth}}|$) Fig. 4(b).

4 Conclusion

In this paper we have proposed a new globally optimal $O(n)$ algorithm for the PnP problem. The algorithm was developed using a cost function for general camera models. The minimization of that cost function can be written as a semi definite positive program, which is efficiently solved. The main difference to other global optimization algorithms is the avoiding of a branch and bound algorithm, which results in a fast computation. We believe that the method of expressing the minimization as a sum of squares problem also has a high potential for other Computer Vision algorithms, like structure from motion.

References

[1] Sameer Agarwal, Manmohan Chandraker, Fredrik Kahl, David Kriegman, and Serge Belongie. Practical global optimization for multiview geometry. In *European Conference on Computer Vision*, pages 592–605, 2006.

[2] Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. In *IEEE European Conference on Computer Vision*, volume 4, pages 282–296, 2002.

[3] H. Araújo, R.J. Carceroni, and C.M. Brown. A fully projective formulation to improve the accuracy of Lowe's pose-estimation algorithm. In *Computer Vision and Image Understanding*, volume 71, pages 227–238, 1998.

[4] D. DeMenthon and L.S. Davis. Exact and approximate solutions of the perspective-three-point problem. *IEEE Transactions on PAMI*, 14(11):1100–1105, Nov 1992.

[5] Andreas Ess, Alexander Neubeck, and Luc van Gool. Generalised linear pose estimation. In *British Machine Vision Conference (BMVC '07)*, 2007.

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

[7] Michael D. Grossberg and Shree K. Nayar. A general imaging model and a method for finding its parameters. In *IEEE International Conference on Computer Vision*, pages 1100–1105, 2001.

[8] R. Hartley and F. Kahl. Global optimization through searching rotation space and optimal estimation of the essential matrix. In *International Conference on Computer Vision*, 2007.

[9] M. L. Liu and K. H. Wong. Pose estimation using four corresponding points. *Pattern Recogn. Lett.*, 20(1):69–74, 1999.

[10] C.P. Lu, G.D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions on PAMI*, 22(6):610–622, 2000.

[11] Francesc Moreno-Noguer, Vincent Lepetit, and Pascal Fua. Accurate non-iterative o(n) solution to the pnp problem. *IEEE 11th International Conference on Computer Vision.*, pages 1–8, 14-21 Oct. 2007.

[12] Danis Oberkampf, Daniel F. DeMenthon, and Larry S. Davis. Iterative pose estimation using coplanar feature points. In *Computer Vision and Image Understanding*, volume 63, pages 495–511, 1996.

[13] C. Olsson, F. Kahl, and M. Oskarsson. Optimal estimation of perspective camera pose. *18th International Conference on Pattern Recognition*, 2:5–8, 2006.

[14] P.A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Math. Program*, (96):293–320, 2003.

[15] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*, 2004.

[16] Long Quan and Zhong-Dan Lan. Linear n-point camera pose determination. *IEEE Transactions on PAMI*, 21(8):774–780, 1999.

[17] Gerald Schweighofer and Axel Pinz. Fast and globally convergent structure and motion estimation for general camera models. In *17th British Machine Vision Conference*, pages 147 – 156, 2006.

[18] Gerald Schweighofer and Axel Pinz. Robust pose estimation from a planar target. *IEEE Transactions on PAMI*, 28:2024 – 2030, 2006.

[19] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Special issue on Interior Point Methods (CD supplement with software).