Journal of Mathematical Imaging and Vision manuscript No.
(will be inserted by the editor)

CvxPnPL: A Unified Convex Solution to the Absolute Pose Estimation Problem from Point and Line Correspondences

Sérgio Agostinho · João Gomes · Alessio Del Bue

Received: date / Accepted: date

Abstract We present a novel certifiable convex method to estimate 3D pose from mixed combinations of 2D-3D point and line correspondences, the Perspective-n-Points-and-Lines problem (PnPL). We merge the contributions of each point and line into a unified Quadratically Constrained Quadratic Problem (QCQP) and then relax it into a Semidefinite Program (SDP) through Shor's relaxation. In this way, we jointly handle mixed configurations of points and lines in a single computational framework. Furthermore, the proposed relaxation allows us to recover a finite number of solutions under ambiguous configurations. In such cases, the 3D pose candidates are found by further enforcing geometric constraints on the solution space and then retrieving such poses from the intersections of multiple quadrics. Experiments provide results in line with the best performing state of the art methods while providing the flexibility of solving for an arbitrary number of points and lines, while the convex formulation provides a framework for a posteriori validation of globally optimal solutions. [1]

Keywords Absolute Pose Estimation · Point and Line Correspondences · Convex Optimization · Semidefinite Relaxation

S. Agostinho and J. Gomes
Instituto de Sistemas e Robótica/LARSys
Instituto Superior Técnico, Universidade de Lisboa
Portugal
E-mail: {sergio.agostinho@,jpg@isr.}tecnico.ulisboa.pt

A. Del Bue
Fondazione Istituto Italiano di Tecnologia
Genoa, Italy
E-mail: alessio.delbue@iit.it

[1] The code implementation of CvxPnPL is available at `https://github.com/SergioRAgostinho/cvxpnpl`.

1 Introduction

The problem of estimating the relative 3D pose between an object and a camera, given a number of 2D-3D correspondences, is well studied and it enabled a number of very successful applications in Robotics and Augmented Reality (AR) [26]. The absolute pose problem is challenging because under certain geometric configurations of 2D-3D points and/or lines, there can exist more than a unique valid 3D pose. In such cases, these multiple poses need to be retrieved and it is up to the user to rely on external information to disambiguate which one is correct. As such, it is understandable that few attempts were made to tackle the problem through convex optimization. A convex problem can be unbounded and have no global optimum, it can have a single global optimum as is often the case, or it can have infinite global optima e.g, the entire domain of a constant function is its argmin. The existence of a finite countable number of solutions implies that the problem is non-convex. However, we show that through a relaxation of the original problem, it is possible to address it with convex optimization and still retrieve multiple solution. The method makes use of point and line correspondences, to leverage collinearity and coplanarity constraints as in [32, 48]. We formulate our optimization problem as a Quadratically Constrained Quadratic Program (QCQP), which we further relax into a Semidefinite Program (SDP) using Shor's relaxation [28]. We experimentally verify that our relaxation is tight, has no duality gap and is certifiably optimal. Our method is the first convex formulation to solve the Perspective-n-Points-and-Lines (PnPL) problem from 2D-3D correspondences, being able to recover up to 4 ambiguous poses. We modified the formulation from Zhou et al. [48] to fully exploit all geometric information provided

by the point and line correspondences and, lastly, we present a modification to Kukelova's et al. [22] E3Q3 method to handle the further constrained case of the intersection of 6 quadrics with 3 unknowns. Our experimental results are in line with the most accurate state-of-the-art methods, with the benefit of providing guarantees of global optimality.

2 Related Work

The literature on pose estimation from 2D-3D correspondences is extensive and a comprehensive review is out of scope for this paper. Our method is designed for a central camera and non-minimal combinations of points and lines, despite being able to handle the minimal or planar points-only cases. For this reason we restrict the review to the following sub-topics.

Perspective-n-Points. The first approach to effectively estimate pose from 2D-3D correspondences was the DLT [1]. The DLT recovers both the camera intrinsics and pose, and as such, tends to achieve lower accuracy when compared to methods which make use of the intrinsic information of the camera. However, it could scale to an arbitrary number of correspondences with only a linear increase in computational complexity. Subsequent approaches to solve the minimal [11,14] and non-minimal [2] problems all suffered from poor scaling for a large number of points. Schweighofer and Pinz [37] proposed an $O(n)$ convex method to solve the PnP problem for general cameras: they formulated a SDP problem around a quaternion rotation and their method handled the planar case separately. EPnP [23] is also an $O(n)$ method that relies on a parametrization based on four control points and a linearization step to simplify the optimization problem. Subsequent works avoided the linearization step and tackled the polynomial problem directly [16,46], some targeting robustness and outlier rejection [24,13], others proposing a formulation for universal cameras [20]. Among all, OPnP [46] shows the most accurate results for the central camera case.

Perspective-n-Lines. The first works addressed the PnL problem in the minimal 3 lines case [10,12]. The minimal problem was revisited recently by Xu et al. [42], which observed that there could be a maximum of eight possible solutions. Ansar and Daniilidis [2] proposed one of the first PnL methods, but it struggles to scale with large number of correspondences. The same year, Hartley and Zisserman [15] proposed an adaptation of DLT for lines. Mirzae and Roumelioutis [27] estimated the camera rotation matrix from a system of polynomial equations whose solution is extracted from eigen-decomposition. Přibyl et al. [30,31] proposed a DLT inspired method which makes use of Plücker line parameterization to formulate the problem as a system of linear equations. Zhang et al. [45] proposed a robust method to estimate the pose from multiple line triplets. Lastly, Zhou et al. [47] addressed the PnL problem in terms of two algebraic distances to approximate Geometric distance. The method skips the use of a Gröbner basis solver by using the first order optimality conditions of its polynomial equation with a more stable hidden variable method.

Perspective-n-Points-and-Lines. The literature for mixed combinations of points and lines is briefer compared to the previous two modalities. The works of Ramaligan et al. [32] and Zhou et al. [48] address the minimal cases, while for non-minimal cases the DLT approach [15] can naturally be adapted to take contribution from points and lines. Subsequently, Kuang and Åström [21] proposed a method to jointly estimate the pose and focal length from points, lines and points with direction. Vakhitov et al. [40] proposed an adaptation to EPnP and OPnP and extended their support to lines.

Convex Relaxations on Rotation Matrices. Saunderson et al. [35] provided an in-depth analysis on the properties of the convex hull enclosing rotation matrices, and the works [34,18,7,8,33,43] showed that despite the non-convex nature of the SO(3) group, there exist successful relaxation strategies. Carlone et al. [9] solved the planar pose graph optimization problem resorting to an SDP relaxation. This relaxation is mostly tight and in the cases where it is not, it still provided a suboptimal reduced search space to extract a meaningful solutions. This work was later extended to the full 3D case by Rosen et al [33]. Our method draws inspiration from the recent work of Briales et al. [7,8] that enforce the orthogonal and the determinant constraints of a rotation matrix using only quadratic constraints; apply a QCQP relaxation; and achieve remarkable results for non-minimal 3D registration and relative pose determination between two views.

3 A Unified Formulation for Point and Line Correspondences

We wish to formulate an optimization problem, with respect to the 3D pose of a model, that combines geometric information from both points and lines. To do so, we employ the collinear and coplanarity constraints introduced by Ramalingam et al. [32]. Let us consider $\mathbf{p}_i$ as a 3D point from a set of n points, and the tuple $(\mathbf{l}_{\mathbf{p}1j}, \mathbf{l}_{\mathbf{p}2j})$ as two points parametrizing a 3D line from a set of m lines, defined in the object's reference frame.

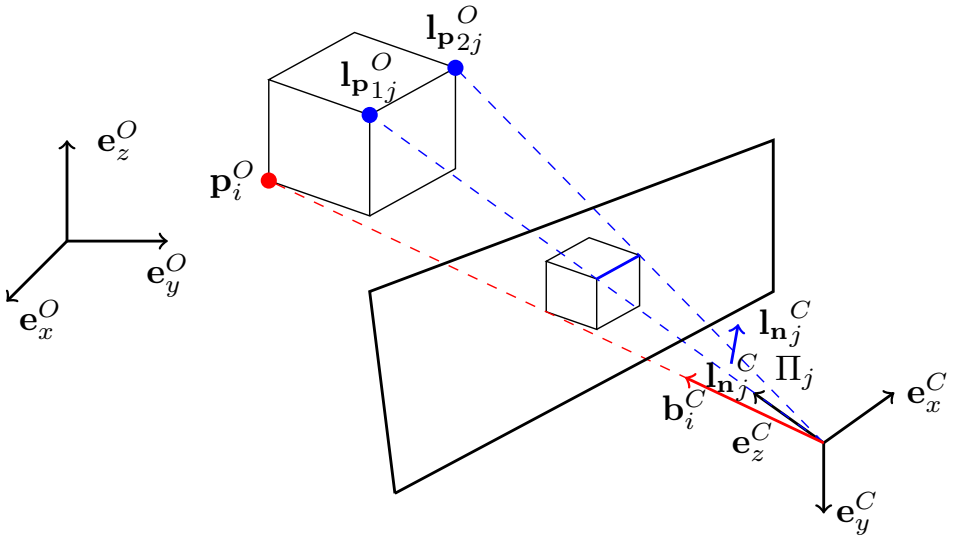

Fig. 1: Example of 2D-3D correspondences from a single point and line. Elements belonging to point correspondences are represented in red and elements belonging to line correspondences are represented in blue. The superscripts O and C denote that the element belongs to the object and camera's reference frames, respectively. Best viewed in color.

Refer to Figure 1 for a visualization of the problem geometry. The 3D model's pose with respect to the camera is given by the rotation matrix $\mathtt{R}$ and the translation vector $\mathbf{t} \in \mathbb{R}^3$. We also define $\mathbf{r}$ as the vectorization of $\mathtt{R}$, such that $\mathbf{r} = \mathrm{vec}(\mathtt{R})$.

Point Correspondences. The projection of a 3D point $\mathbf{p}_i$ onto the image plane will intersect a given pixel. A pixel in the image plane can be represented in homogeneous coordinates as $[u_i, v_i, 1]^\top$. Consider a camera whose intrinsic parameters are known and described by matrix $\mathtt{K}$. The corresponding bearing vector/ray for this pixel is calculated as

$$\vec{b}_i = \mathtt{K}^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}, \tag{1}$$

determined up to scale factor. Both $\mathbf{p}_i$ and $\mathbf{b}_i$ are necessarily collinear, allowing us to enforce $\mathbf{b}_i \times (\mathtt{R}\mathbf{p}_i + \mathbf{t}) = 0$, where the operator $\times$ represents the cross product. Then, let us write $\lfloor \mathbf{b}_i \rfloor_\times (\mathtt{R}\mathbf{p}_i + \mathbf{t}) = 0$, employing the equivalent skew symmetric matrix representation $\lfloor \mathbf{b} \rfloor_\times$ for the cross product. The matrix $\lfloor \mathbf{b} \rfloor_\times$ has rank 2, so despite each point correspondence contributing 3 equations, only 2 of them are linearly independent. Stacking the contributions from all n points and rearranging terms yields $\mathtt{C}_\mathtt{p}\mathbf{r} + \mathtt{N}_\mathtt{p}\mathbf{t} = 0$, where $\mathtt{C}_\mathtt{p}$ and $\mathtt{N}_\mathtt{p}$ are $3n \times 9$ and $3n \times 3$ matrices, respectively.[2]

Line Correspondences. Any 3D line that is not collinear with the origin of the camera's reference frame, forms a unique plane Π_j with it (see Figure 1). We denote by $\mathbf{l}_{\mathbf{n}j}$ the normal of Π_j. The line constraints are built from the fact that, in the camera's space, both $\mathbf{l}_{\mathbf{p}1j}$ and $\mathbf{l}_{\mathbf{p}2j}$ belong to Π_j and are therefore orthogonal to $\mathbf{l}_{\mathbf{n}j}$, satisfying

$$\mathbf{l}_{\mathbf{n}j} \cdot (\mathtt{R}\mathbf{l}_{\mathbf{p}ij} + \mathbf{t}) = 0 \quad : i = 1, 2. \tag{2}$$

Each line correspondence contributes two linearly independent equations, and upon stacking, it forms the system

$$\mathtt{C}_\mathtt{L}\mathbf{r} + \mathtt{N}_\mathtt{L}\mathbf{t} = 0, \tag{3}$$

where $\mathtt{C}_\mathtt{L}$ is a $2m \times 9$ matrix, and $\mathtt{N}_\mathtt{L}$ is $2m \times 3$.[2] Consider $\mathbf{b}_{\mathbf{l}j1}$ and $\mathbf{b}_{\mathbf{l}j2}$, the bearing vectors associated with two distinct points sampled from the 2D line projection. The normal $\mathbf{l}_{\mathbf{n}j}$ can be recovered from their cross product $\mathbf{l}_{\mathbf{n}j} = \lfloor \mathbf{b}_{\mathbf{l}j1} \rfloor_\times \mathbf{b}_{\mathbf{l}j2}$.

[2] The full derivation of this expression can be found in the Appendix.

3.1 Composing the Homogeneous System

Stacking together the contributions from points and lines requires us to further define the matrices as:

$$\mathtt{C} = \begin{bmatrix} \mathtt{C}_\mathtt{P} \\ \mathtt{C}_\mathtt{L} \end{bmatrix}, \quad \mathtt{N} = \begin{bmatrix} \mathtt{N}_\mathtt{P} \\ \mathtt{N}_\mathtt{L} \end{bmatrix}. \tag{4}$$

Both $\mathtt{C}$ and $\mathtt{N}$ are composed of $3n + 2m$ rows, of which $2n + 2m$ are linearly independent. It is a known linear algebra result that, given an optimal $\hat{\mathbf{r}}$, the optimal unconstrained solution $\hat{\mathbf{t}}$ to the overdetermined system

$$\mathtt{C}\mathbf{r} + \mathtt{N}\mathbf{t} = 0, \tag{5}$$

is given by $\hat{\mathbf{t}} = -(\mathtt{N}^T\mathtt{N})^{-1}\mathtt{N}^\top\mathtt{C}\hat{\mathbf{r}}$. Substituting back into Eq. (5), allows us to write the complete system as $\mathtt{A}\mathbf{r} = 0$, with

$$\mathtt{A} = (\mathtt{I}_{3n+2m} - \mathtt{N}(\mathtt{N}^\top\mathtt{N})^{-1}\mathtt{N}^\top)\mathtt{C}, \tag{6}$$

where $\mathtt{A}$ is a $(3n + 2m) \times 9$ matrix and $\mathtt{I}_{3n+m}$ is the identity matrix of size $3n + 2m$. The second row block of $\mathtt{A}$ is the residual of the projection of $\mathtt{C}$ onto the column space of $\mathtt{N}$.

4 Convex Formulation

The formulation in Eq. (5) does not consider the specific structure of $\mathbf{r} = \text{vec}(\mathtt{R})$. Proper rotation matrices are orthogonal and satisfy $\det(\mathtt{R}) = +1$. While the orthogonality is a quadratic constraint, the determinant is cubic. However, it was show in [39], that enforcing the right-hand convention in the columns or rows, ensures a positive determinant, allowing us to express this inherently cubic constraint in an equivalent quadratic form. As such, we formulate our problem as:

$$\min_{\mathbf{r}} \quad \|\mathtt{A}\mathbf{r}\|^2 \tag{7a}$$
$$\text{s.t.} \quad \mathtt{R}^\top\mathtt{R} = \mathtt{I}_3 \tag{7b}$$
$$\mathtt{R}\mathtt{R}^\top = \mathtt{I}_3 \tag{7c}$$
$$\mathtt{R}^{(i)} \times \mathtt{R}^{(j)} = \mathtt{R}^{(k)}, \tag{7d}$$

with $(i, j, k) \in \{(1, 2, 3), (2, 3, 1), (3, 1, 2)\}$ and $\mathtt{R}^{(i)}$ denoting the i-th column of $\mathtt{R}$. While the inclusion of Eq. (7c) might seem redundant, we show in Section 7.1 that it helps the solver to converge faster and achieve estimates with lower error. The previous problem can be written in a canonical form given by:[2]

$$\min_{\tilde{\mathbf{r}}} \quad \tilde{\mathbf{r}}^T\mathtt{Q}_0\tilde{\mathbf{r}} \tag{8a}$$
$$\text{s.t.} \quad \tilde{\mathbf{r}}^T\mathtt{Q}_i\tilde{\mathbf{r}} = 0 : i \in \{1, \dots, 21\} \tag{8b}$$
$$\tilde{\mathbf{r}}_{10} = 1, \tag{8c}$$

where $\tilde{\mathbf{r}}$ is the homogeneous vector $\tilde{\mathbf{r}} = [\mathbf{r}^\top 1]^\top$ of size 10. It is also important to highlight that $\mathtt{Q}_0$ and $\mathtt{Q}_\mathtt{i}$ belong to the subspace of symmetric matrices $\mathbb{S}_{10}$. The constraints in Eq. (8b) are non-convex, so in order to overcome this limitation, we relax the problem into a SDP employing Shor's relaxation [28,38]. This relaxation exploits the trace identity $\text{tr}(\tilde{\mathbf{r}}^\top\mathtt{Q}\tilde{\mathbf{r}}) = \text{tr}(\mathtt{Q}\tilde{\mathbf{r}}\tilde{\mathbf{r}}^\top)$, allowing to rewrite the QCQP as:

$$\min_{\mathtt{Z}} \quad \text{tr}(\mathtt{Q}_0\mathtt{Z}) \tag{9a}$$
$$\text{s.t.} \quad \text{tr}(\mathtt{Q}_i\mathtt{Z}) = 0 : i \in \{1, \dots, 21\} \tag{9b}$$
$$\mathtt{Z} \succcurlyeq 0 \tag{9c}$$
$$\mathtt{Z}_{10,10} = 1, \tag{9d}$$

where $\mathtt{Z} = \tilde{\mathbf{r}}\tilde{\mathbf{r}}^\top$ is a 10×10 matrix with rank 1 by construction. The rank constraint is not convex and therefore it is dropped, relaxing the original problem. One might wonder if it is worth to move from the original Problem (7) with nine unknowns to one (9) with 100 unknowns. This line of thinking overlooks the fact that (7) is not convex and as such, common optimization strategies are sensitive to initialization and cannot provide guarantees with respect to the optimality of the solutions returned, contrary to what is done in Section 6. As it turns out, this relaxation plays an important role when handling situations in which there are multiple solutions, a topic we further expand in Section 5. Problem (9) is part of the family of convex cone programs and we resort to the Splitting Conic Solver (SCS) [29] (cf. Appendix E) to compute the optimal estimate $\hat{\mathtt{Z}}$.

5 Rotation Recovery

We can reconstruct the optimal $\tilde{\mathbf{r}}_k^*$ from:

$$\tilde{\mathbf{r}} = \sum_{k=1}^{K} \alpha_k \mathbf{v}_{\lambda_k} \tag{10}$$
$$\tilde{r}_{10} = 1, \tag{11}$$

where K denotes the rank of $\hat{\mathtt{Z}}$ and $\mathbf{v}_{\lambda_k}$ are the eigenvectors associated with non-null eigenvalues. These eigenvectors are not vectorized rotations themselves, but they span a linear space which the solutions inhabit. While dealing with rank 1 is relatively straightforward, rank 2 and above requires enforcing geometric constraints as in Eqs. (7b), (7c) and (7d) to the solution space in order to retrieve admissible solutions. While it is not optimal to have different procedures dependent on the numerical observation of K, tackling each rank with the its own dedicated procedure allows solving the simpler cases quicker and with less computational resources, reserving the most computationally expensive procedures for the cases that really need it.

Back-substituting the linear constraint in Eq. (11) allows removing one of the unknowns, resulting in an expression of the form:

$$\tilde{\mathbf{r}} = \sum_{k=1}^{K-1} \alpha'_k \mathbf{v}'_k + \mathbf{v}'_0 \tag{12}$$

$$\begin{bmatrix}\mathbf{r}\\1\end{bmatrix} = \sum_{k=1}^{K-1} \alpha'_k \begin{bmatrix}\vdots\\0\end{bmatrix} + \begin{bmatrix}\vdots\\1\end{bmatrix}. \tag{13}$$

Imposing the rotation matrix constraints here results in a system of 21 quadratic polynomial equations with $K-1$ unknowns. This paper provides solutions up to rank 4, meaning that we are able to tackle points-only configurations with 3 or more points, but for lines-only or mixed configurations of points and lines, we require at least 4 elements to get a solution. The next paragraphs, with more details in the supplemental material, describe the procedure to recover such solutions.

Rank 1. When dealing with rank 1 matrix $\hat{\mathtt{Z}}$, we are in a situation where the relaxation is tight. The optimal $\hat{\mathbf{r}}$ can be recovered from the eigenvector associated with the largest eigenvalue as $\mathbf{v}_{\lambda_{\max}} = a\begin{bmatrix}\hat{\mathbf{r}}^\top & 1\end{bmatrix}^\top$. In practice, it is also advisable to reproject this solution to the orthogonal matrix space, so after applying vec^{-1}, we decompose it using SVD and retrieve $\hat{\mathtt{R}}$ as:

$$\mathtt{R}' = \mathrm{vec}^{-1}(\hat{\mathbf{r}}) \tag{14}$$

$$\mathtt{U}\mathtt{D}\mathtt{V}^\top = \mathrm{svd}(\mathtt{R}') \tag{15}$$

$$\hat{\mathtt{R}} = \mathtt{U}\mathtt{V}^\top. \tag{16}$$

Rank 2. Refer to Eq. (12) with $K=2$. After enforcing the linear constraint on the last element of $\tilde{\mathbf{r}}$, we have a single unknown, designated as a. After enforcing the quadratic constraints on the rotation we end up with a system of equations of the form $\mathtt{G}\begin{bmatrix}a^2 & a & 1\end{bmatrix}^\top = 0$,[2] where $\mathtt{G} \in \mathbb{R}^{21\times 3}$. In our experiments, we observed that $\mathtt{G}$ has rank 1. Finding the roots of a single quadratic equation will produce up to two solutions, in line with the hypothesis that the rank is indicative of the number of ambiguous solutions. We retrieve both values of a from the row-wise average of $\mathtt{G}$, denoted as $\bar{\mathbf{g}}^\top$, by finding the roots of the second order polynomial $\bar{g}_1 a^2 + \bar{g}_2 a + \bar{g}_3 = 0$.

Rank 3. The rank 3 case is composed of the two unknowns α'_1 and α'_2 that we shall refer as a and b. After the enforcing the rotation's quadratic constrains we obtain a system of equations of the form $\mathtt{G}\begin{bmatrix}a^2 & b^2 & ab & a & b & 1\end{bmatrix}^\top = 0$, with $\mathtt{G} \in \mathbb{R}^{21\times 6}$. In our experiments, we observed that matrix $\mathtt{G}$ has rank 3, indicating the presence of only 3 linearly independent equations. Consider the column-wise block representation of $\mathtt{G}$ such that $\mathtt{G} = \begin{bmatrix}\mathtt{G}_\mathtt{L} & \mathtt{G}_\mathtt{R}\end{bmatrix}$, with matrices $\mathtt{G}_\mathtt{L} \in \mathbb{R}^{21\times 3}$ and $\mathtt{G}_\mathtt{R} \in \mathbb{R}^{21\times 3}$. Given the rank 3 of $\mathtt{G}$ and resorting to the left pseudo inverse, we can write

$$\begin{bmatrix}a^2\\b^2\\ab\end{bmatrix} = \underbrace{-(\mathtt{G}_\mathtt{L}{}^\top \mathtt{G}_\mathtt{L})^{-1}\mathtt{G}_\mathtt{L}{}^\top \mathtt{G}_\mathtt{R}}_{\mathtt{D}\in\mathbb{R}^{3\times 3}} \begin{bmatrix}a\\b\\1\end{bmatrix}. \tag{17}$$

We pick rows 1 and 3 and after rearranging and substituting some terms, we end with the following expressions:

$$a^3 - (d_{11}+d_{32})a^2 + (d_{11}*d_{32} - d_{12}*d_{31} - d_{13})a + d_{13}*d_{32} - d_{12}*d_{33} = 0 \tag{18}$$

$$b = \frac{1}{d_{12}}(a^2 - d_{11}a - d_{13}), \tag{19}$$

where d_{ij} corresponds to the element in the i-th row and j-th column of matrix $\mathtt{D}$. Extracting the solution amounts to finding the roots of a 3th degree polynomial, yielding 3 solutions for a. Given a value for a, we can find the corresponding b through Eq. (19) and with both, retrieve the optimal $\tilde{\mathbf{r}}$.

Rank 4. In the rank 4 case, we will designate the three unknown α'_1, α'_2 and α_3 by the letters a, b and c. After enforcing the available constraints we obtain the following system $\mathtt{G}\begin{bmatrix}a^2 & b^2 & c^2 & ab & ac & bc & a & b & c & 1\end{bmatrix}^\top = 0$, where $\mathtt{G} \in \mathbb{R}^{21\times 10}$. In our experiments, we empirically observed that $\mathtt{G}$ is composed of 6 linearly independent equations. Similar to rank 3, we consider the column-wise block representation of $\mathtt{G}$ such that $\mathtt{G} = \begin{bmatrix}\mathtt{G}_\mathtt{L} & \mathtt{G}_\mathtt{R}\end{bmatrix}$, this time with matrices $\mathtt{G}_\mathtt{L} \in \mathbb{R}^{21\times 6}$ and $\mathtt{G}_\mathtt{R} \in \mathbb{R}^{21\times 4}$. Given the rank 6 of $\mathtt{G}$ and resorting to the left pseudo inverse, we can write

$$\begin{bmatrix}a^2\\b^2\\c^2\\ab\\ac\\bc\end{bmatrix} = \underbrace{-(\mathtt{G}_\mathtt{L}{}^\top \mathtt{G}_\mathtt{L})^{-1}\mathtt{G}_\mathtt{L}{}^\top \mathtt{G}_\mathtt{R}}_{\mathtt{D}\in\mathbb{R}^{6\times 4}} \begin{bmatrix}a\\b\\c\\1\end{bmatrix}. \tag{20}$$

To find our solution we adapt the E3Q3 method developed by Kukelova et al. in [22]. We pick rows 2, 3 and 6 and treat a as a constant. Doing so, allows us to write Eq. (20) as:

$$\begin{bmatrix}b^2\\c^2\\bc\end{bmatrix} = \begin{bmatrix}d_{22} & d_{23} & d_{21}a + d_{24}\\ d_{32} & d_{33} & d_{31}a + d_{34}\\ d_{62} & d_{63} & d_{61}a + d_{64}\end{bmatrix} \begin{bmatrix}b\\c\\1\end{bmatrix}. \tag{21}$$

After applying the identities $(b^2)c = (bc)b$, $(c^2)b = (bc)c$ and $(b^2)(c^2) = (bc)(bc)$, followed by double substitution yields the homogeneous system

$$\underbrace{\begin{bmatrix}m^{[1]}_{11}(a) & m^{[1]}_{12}(a) & m^{[1]}_{13}(a)\\ m^{[1]}_{21}(a) & m^{[1]}_{22}(a) & m^{[1]}_{23}(a)\\ m^{[1]}_{31}(a) & m^{[1]}_{32}(a) & m^{[2]}_{33}(a)\end{bmatrix}}_{M(a)} \begin{bmatrix}b\\c\\1\end{bmatrix} = 0. \tag{22}$$

The subscript $[\cdot]$ denotes the degree of the polynomial in a. Eq. (22) only has a non-trivial solution if the determinant of $M(a)$ is 0. This amounts to finding the roots of a 4th degree polynomial, yielding our 4 desired solutions. Recovering b and c amounts to substituting a in Eq. (22) for every solution and solving the overdetermined linear system for b and c. As the reader might notice, our particular selection of coefficients $((b, c), a)$ in Eq. (21) or rows in Eqs. (20) is not unique. The same procedure can be done with a different row and coefficient selection.

6 Certificate of a Global Optimal Solution

This section exploits known results of duality theory in establishing lower bounds for an optimization problem. For an introduction to duality we recommend Boyd and Vanderberg [3, Sec. 5]. An in-depth explanation of the technique used in this section can be found in [3, Sec. 5.5.1]. Let us denote by p^* and p^*_{SDP} the optimal values of the primal problem's cost function and its relaxation, and by d^* the optimal dual of the relaxation. The problem solved in Eq. (9) is a relaxation fulfilling $p^*_{SDP} \leq p^*$. This is a convex problem in which strong duality holds, as noted by Briales et al. [8] for a similar problem, and the following condition is verified $d^* = p^*_{SDP} \leq p^*$. If the relaxation is indeed tight, this implies $d^* = p^*_{SDP} = p^*$. When such condition is verified, we a have a certificate that the solution found is a global optimum of the original problem. To solve problem (9), we resort to the off-the-shelf solver SCS [29], that besides computing an optimal solution $\mathtt{Z}$, also provides the dual objective d^*. After retrieving all rotations as detailed in Section 5, we can compute p by substituting $\mathbf{r}$ back in Eq. (7a). If $d^* = p$, this implies $p = p^*$, since $d^* \leq p^* \leq p$. We consider valid solutions, ones that verify $|d^* - p| < \epsilon \approx 10^{-9}$.

7 Experimental Results

We present separate results for the angular and translations errors when solving the PnPL problem. The error metrics used are given by

$$\Delta\mathtt{R} = \hat{\mathtt{R}}^\top \mathtt{R}_{gt}, \tag{23}$$

$$\Delta\mathbf{t} = \frac{\|\hat{\mathbf{t}} - \mathbf{t}_{gt}\|}{\|\mathbf{t}_{gt}\|}, \tag{24}$$

for rotation and translation errors, respectively, with the subscript gt denoting the ground truth. Given the residual rotation $\Delta\mathtt{R}$, the angular error is retrieved from the absolute value of the angle, once $\Delta\mathtt{R}$ is converted to its axis-angle representation. The translation error is computed in normalized form, to prevent that situations where the object's origin $\mathbf{t}_{gt}$ is located far away from the camera origin, dominate the translation error statistics.

Table 1: Average runtime in milliseconds of each method over the benchmark performed in Figure 2. These metrics were acquired on an Intel® Xeon® CPU E5-2697 v4 @ 2.30GHz × 16 logical cores. The keyword `baseline` refers to our baseline method, which enforces orthonormality of both rows and columns. The keyword `stripped` refers to the modification of the baseline method in which the orthonormality of rows constraint was discarded.

Method	PnP	PnL	PnPL
`baseline`	18ms	24ms	22ms
`stripped`	23ms	33ms	30ms

7.1 Synthetic Data Evaluation

We generate a simulation environment where we instantiate the necessary numbers of points and lines to test each configuration. To define a 3D line, we parameterize it as a tuple of two points. All these points are randomly generated inside an origin centered, axis aligned, 3D cube of edge size 0.6, which represents the model's frame of reference. We then apply a random 3D transformation which guarantees that the origin of the model will lie somewhere in $[-0.5, 0.5] \times [-0.5, 0.5] \times [0.4, 2.0]$. To project the point onto the image plane, we adopt the same camera intrinsics as the Kinect v1. Ultimately, we are trying to replicate similar conditions to those present in the LINEMOD dataset [17], that contains scenes targeting object pose estimation tasks. We apply Gaussian noise of various levels to the projected pixels to simulate noise in the camera. With every single run, we instantiate new random elements, a new random pose and apply random pixel noise to the projections. We then submit all methods to the same realization to ensure that we have a direct comparison in every single run. This ensures that even if a specific realization is degenerate or simply challenging, all methods are subjected to it.

Inclusion of the Redundant Orthonormality Constraint. The inclusion of both orthonormality of rows and columns constraints is theoretically redundant in our problem formulation. However, experimental evaluation whose results are displayed in Figure 2 shows that including both constraints helps achieve more precise estimates

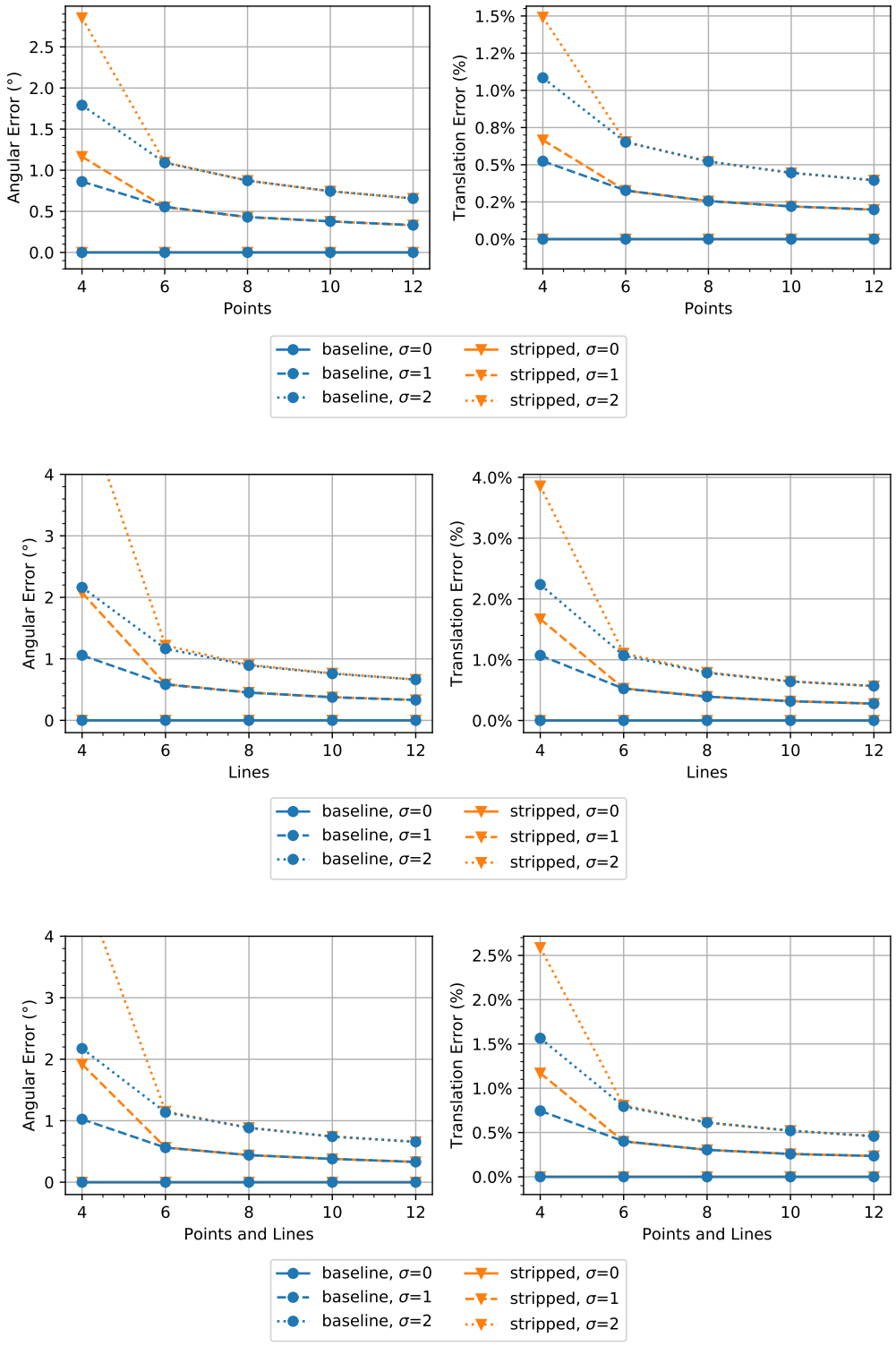

Fig. 2: Comparison between including both row and column orthonormality constraints (baseline) or only the former (stripped) for a PnP scenario with 4+ points (top), PnL scenario with 4+ lines (middle) and a PnPL scenario with 4+ points and lines (bottom). The median angular and translation errors are shown on the left and right, respectively, for different numbers of elements in the scenario, employing Gaussian pixel noise with standard deviation σ computed over 10^4 runs.

with few points and lines, when noise is present. Furthermore, in Table 1 we observe that adding the redundant constraints helps the SDP solver converging to a solution faster. This also suggests that the poorer estimate precision without the additional constraint might be explained by early stopping occurrences caused from reaching the maximum number of iterations allowed, a parameter set to 2500 iterations in this benchmark. The results also validate that working with point correspondences constrains the problem further, resulting in higher convergence speed in comparison with line correspondences.

Justifying the SDP Inclusion. Looking at Eq. (7a), one might wonder if there is an actual need to formulate the problem as a SDP in order to extract a valid rotation. After all, the optimal rotation $\mathbf{r}$ needs to lie in the null space of $\mathtt{A}$. While that is true for noiseless cases, in the presence of noise, the null space collapses. One might still argue that the right singular vector associated with the smallest singular value of $\mathtt{A}$ should still provide a reasonable solution, once reshaped and projected to the orthogonal matrix space. In Figure 3 we compare the results between both approaches. For simplicity, we only consider non-ambiguous cases where the null space of $\mathtt{A}$ has a dimensionality of 1. It is clear from the figure that the advantage of the SDP formulation stands out under noisy conditions.

Scalability with the Number of Correspondences. Linear scalability with the number of correspondences is one of the golden standards for perspective-n-points/lines algorithms. The cost matrix $\mathtt{Q}_0 \in \mathbb{R}^{10\times 10}$ irrespective of the number of correspondences. Even though computing $\mathtt{Q}_0$ involves a number of operations that scale with the number of correspondences, these tend to be negligible in the global execution time for the usual amount of points involved in typical PnP/L scenarios. Our experiments in Figure 4, in scenarios with an increasing number of points, show exactly that. Up to roughly 100 points, the SDP solver dominates the execution time. Fewer points render the problem more challenging to optimize and convergence to a minimum is slower. As the the number of points increases, the problem becomes geometrically more constrained and the solver is quicker to reach the solution. After a certain stage, the remaining numerical operations start dominating the execution time and the linear scaling becomes noticeable.

Rank of $\mathtt{Z}$. The rank of the solution matrix $\mathtt{Z}$, dictates the subsequent procedure that is applied in order to retrieve an optimal rotation. In Figures 5 and 6, we display the values estimated for $\mathrm{rank}(\mathtt{Z})$ in the synthetic PnP, PnL and PnPL scenarios with 4+ points and/or lines. Assuming λ_k denotes the k-th eigenvalue of $\mathtt{Z}$, with $k = 1, \ldots, 10$, the $\mathrm{rank}(\mathtt{Z}) = \sum_{k=1}^{10} \mathbf{1}_{\lambda_k > \tau}(\lambda_k)$, where $\mathbf{1}_{\mathcal{A}}(x)$ is the indicator function of set $\mathcal{A}$ such that

$$\mathbf{1}_{\mathcal{A}}(x) = \begin{cases} 1 & \text{if } x \in \mathcal{A} \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

For these experiments, the threshold is set to the value $\tau = 10^{-3}$. The important difference between both figures is the maximum number of allowed iterations for the convex solver: 2500 in the former case and 2500000 in the latter. Apart from this experiment, we set the maximum number of iteration to 2500 in all other experiments. Because 4+ points and/or lines is in general a non-minimal configuration, the mode is $\mathrm{rank}(\mathtt{Z}) = 1$. It is also evident from both figures, that increasing the number of iterations allows the solver to further reduce the solution rank. We can also observe that as the number of points and/or lines increases, these constraint the problem better and lower the ceiling for the number of iterations required. Finally, Figure 5 also suggests that while our method benefits from accurate estimation of $\mathrm{rank}(\mathtt{Z})$, this is not critical for its success. As long as the rank estimate is above or equal to its actual value, we are still able to retrieve the desired pose solution. We present some additional results supporting this observation in Appendix F.

Comparison with Other Methods. We compare our method with a number of other available approaches in the scenario of 4+ points, 4+ lines and the combination of 4+ points and lines. We evaluate all methods for different levels of Gaussian pixel noise and number of elements in the scene. We benchmark against Vakhitov et al. [40] who developed EPnPL and OPnPL as extensions of the original EPnP [23] and OPnP [46], to support mixed combinations of points and lines. For points-only scenarios, the existing number of PnP methods in the literature is considerable, so we opted to focus on the more popular ones such as EPnP, UPnP [20] and OPnP.[3] For the lines-only scenarios, we compare against Mirzaei and Roumeliotis [27], RPnL [45], EPnPL and OPnPL.[4] Finally, for the mixed scenarios we only compare directly with EPnPL and OPnPL. The condensed results for all these scenarios are displayed in Fig. 7. We can verify that our method achieves results in line with the most precise state of the art methods.

[3] EPnP implementation from OpenCV [6], UPnP implementation from OpenGV [19] and OPnP implementation from Vakhitov et al. [40].

[4] All PnL implementations are available from Vakhitov et al. [40].

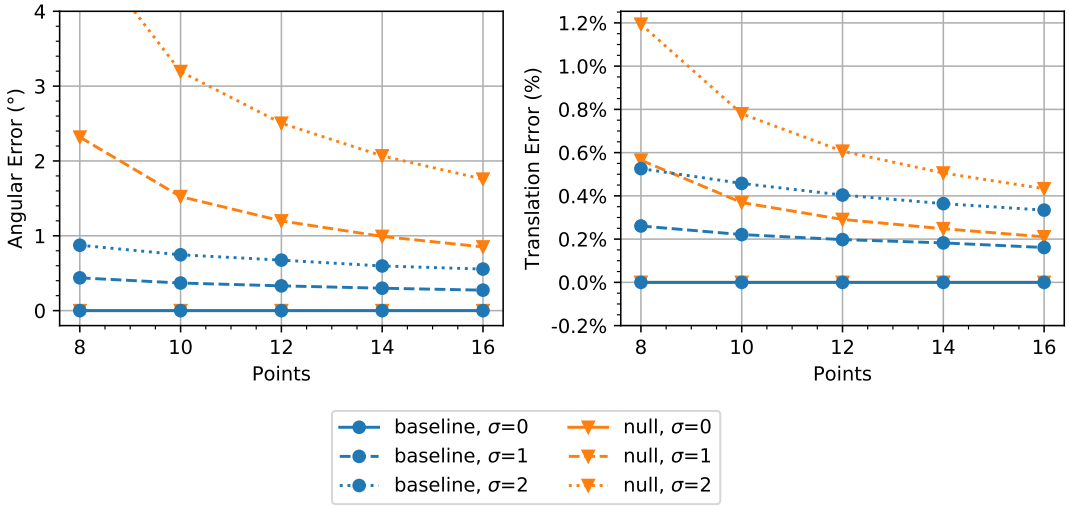

Fig. 3: Comparison between cvxpnpl (baseline) and null space search strategy described in Section 7.1 (null) for a PnP scenario with 8+ points. The median angular and translation errors are shown on the left and right, respectively, for different numbers of elements in the scenario, employing Gaussian pixel noise with standard deviation σ computed over 10000 runs.

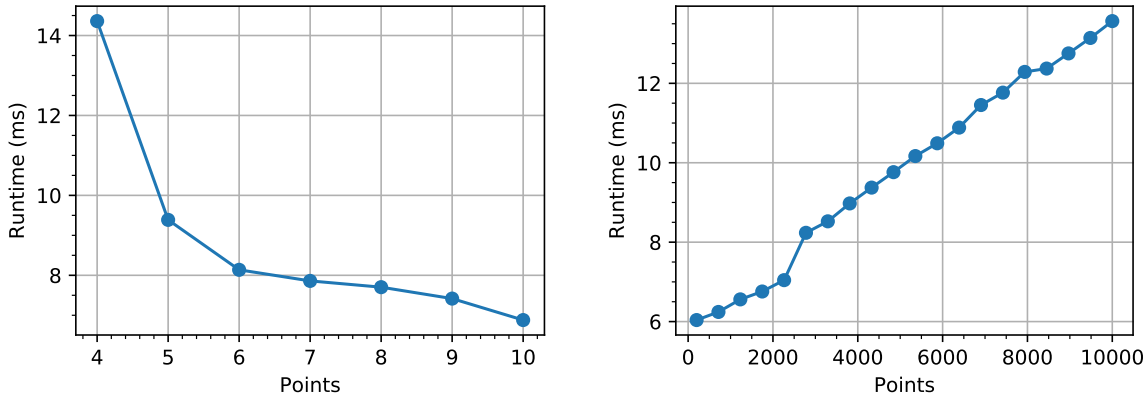

Fig. 4: Median runtime performance in ms for two PnP scenarios with increasing number of points, over 10^4 runs.

7.2 Real Data Evaluation

To test with real data we opted to establish ground truth 2D-3D correspondences, following the strategy employed in [5] to infer for each pixel belonging to an object, its normalized 3D object coordinate. The unnormalized coordinate of the pixel in the 3D object is recovered from the lengths of the 3D bounding box dimensions of the object. By leveraging popular object pose estimation datasets, we have access to real objects CAD models and images of these objects under known 3D poses. With both these elements we can accurately extract dense 2D-3D correspondences in each image. In order to only select meaningful points and lines from the image, we use SIFT [25] and LSD [41] to generate point and line-segment proposals. These are respectively pruned and trimmed in order to only consider proposals lying inside valid object masks. We evaluated all methods in all sequences from the LINEMOD dataset [17], including the richly annotated `benchvise` sequence created in Brachmann et al. [5], coined as the Occlusion dataset.

In Table 2 we present a quantitative comparison of all methods over all sequences employing points, lines or both. Whenever points are used, OPnP and OPnPL achieve the best results. However, contrary to the

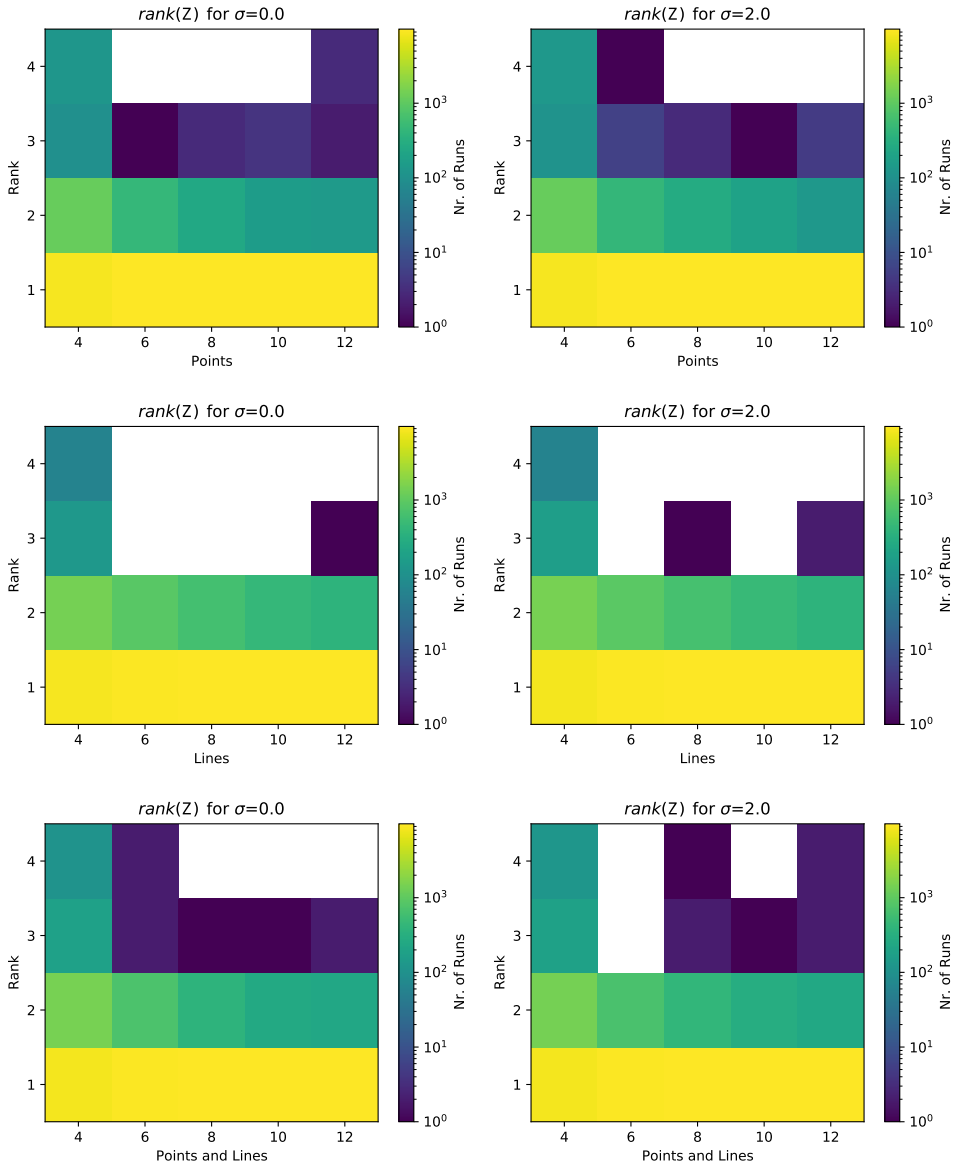

Fig. 5: A visual representation of the different values of rank(Z) over 10000 runs for a PnP scenario with 4+ points (top), PnL scenario with 4+ lines (middle) and a PnPL scenario with 4+ points and lines (bottom), when the convex solver is allowed a maximum number of 2500 iterations. We display the rank value under noiseless conditions (left) and with added Gaussian pixel noise with standard deviation $\sigma = 2$. The color coding is log-scaled (figure best seen in colour). The absence of color (white) in some locations, signifies that there were no runs for a given number of points/lines (horizontal axis) that produced a solution with that rank (vertical axis).

synthetic experiments, when relying only on line segments, CvxPnPL achieves the lowest pose error in all sequences. In Figure 8, we show two images in which we employed the objects' pose estimates to render their masks. The poses were estimated resorting to both points and lines. We also superimpose the point and line detections used to compute correspondences. We note that while certain parts of an object contour can be approximated by straight lines, this is usually just a subset of the contour. To avoid noisy detections, we also discard line segments that span less than 5 pixels that are outside known object masks.

Table 2: Median pose error of all methods over a number of sequences. The tags `Line` and `Occl` are used to designate respectively the LINEMOD and Occlusion datasets. The angular error is expressed in degrees (°) and the translation error is normalized by the ground truth translation vector norm and expressed in parts per thousand (‰). For a fair comparison, we only consider frames in which all methods produce a pose estimate. (top) Pose estimated resorting only to points. (middle) Pose estimated resorting only to lines. (bottom) Pose estimated resorting to both points and lines.

Points	CvxPnPL °/‰	EPnP °/‰	OPnP °/‰	UPnP °/‰
Line ape	0.284 / 1.702	0.299 / 1.716	**0.260** / **1.631**	0.275 / 1.665
Line benchvise	0.091 / 1.280	0.093 / 1.288	**0.088** / **1.276**	0.090 / 1.279
Line bowl	0.143 / 1.497	0.141 / 1.527	**0.134** / **1.461**	0.137 / 1.471
Line cam	0.102 / 1.289	0.103 / 1.293	**0.095** / **1.274**	0.099 / 1.278
Line can	0.103 / 1.306	0.108 / 1.315	**0.099** / **1.299**	0.102 / 1.301
Line cat	0.133 / 1.377	0.139 / 1.401	**0.127** / **1.358**	0.129 / 1.369
Line cup	0.181 / 1.570	0.174 / 1.583	**0.164** / **1.475**	0.165 / 1.482
Line driller	0.106 / 1.368	0.112 / 1.403	**0.103** / **1.361**	0.107 / 1.362
Line duck	0.261 / 1.701	0.258 / 1.636	**0.230** / **1.551**	0.241 / 1.580
Line eggbox	0.137 / 1.368	0.138 / 1.371	**0.128** / **1.341**	0.133 / 1.355
Line glue	0.168 / **1.532**	0.182 / 1.554	**0.163** / 1.547	0.178 / 1.609
Line holepuncher	0.161 / 1.447	0.158 / 1.447	**0.143** / **1.391**	0.147 / 1.403
Line iron	0.124 / 1.324	0.128 / 1.352	**0.119** / **1.320**	0.124 / 1.325
Line lamp	0.098 / 1.328	0.100 / 1.341	**0.095** / **1.319**	0.097 / 1.325
Line phone	0.134 / 1.370	0.138 / 1.392	**0.129** / **1.354**	0.132 / 1.381
Occl	0.175 / 1.640	0.178 / 1.654	**0.161** / **1.557**	0.169 / 1.610
All	0.145 / 1.428	0.148 / 1.443	**0.136** / **1.395**	0.141 / 1.412

Lines	CvxPnPL °/‰	EPnPL °/‰	Mirzaei °/‰	OPnPL °/‰	Pluecker °/‰	RPnL °/‰
Line ape	**0.247** / **1.478**	0.303 / 1.628	0.797 / 6.057	0.304 / 1.679	2.467 / 243.141	0.855 / 3.026
Line benchvise	**0.111** / **1.299**	0.128 / 1.368	0.548 / 4.295	0.119 / 1.307	0.516 / 42.638	0.628 / 3.648
Line bowl	**0.177** / **1.419**	0.250 / 2.307	0.768 / 6.226	0.206 / 1.563	2.262 / 163.411	0.805 / 5.333
Line cam	**0.119** / **1.301**	0.133 / 1.354	0.522 / 3.685	0.125 / 1.344	0.583 / 58.059	0.767 / 2.814
Line can	**0.105** / **1.287**	0.116 / 1.317	0.449 / 3.327	0.109 / 1.288	0.442 / 39.037	0.569 / 2.205
Line cat	**0.150** / **1.404**	0.175 / 1.572	0.612 / 5.742	0.159 / 1.457	0.810 / 120.878	0.647 / 4.320
Line cup	**0.209** / **1.460**	0.261 / 1.775	0.559 / 5.064	0.221 / 1.713	2.894 / 232.480	0.755 / 4.501
Line driller	**0.119** / **1.349**	0.163 / 1.592	0.919 / 5.785	0.131 / 1.488	1.166 / 79.323	0.766 / 7.538
Line duck	**0.220** / **1.399**	0.240 / 1.545	0.682 / 4.320	0.239 / 1.434	1.220 / 165.782	0.746 / 2.707
Line eggbox	**0.141** / **1.322**	0.163 / 1.396	0.598 / 3.735	0.149 / 1.357	0.881 / 85.035	0.672 / 3.669
Line glue	**0.171** / **1.482**	0.282 / 2.051	0.705 / 6.180	0.205 / 1.608	1.958 / 210.189	0.560 / 4.067
Line holepuncher	**0.182** / **1.419**	0.349 / 1.989	0.796 / 4.736	0.204 / 1.512	4.119 / 254.884	0.950 / 4.552
Line iron	**0.125** / **1.316**	0.158 / 1.481	0.773 / 4.757	0.136 / 1.340	0.905 / 61.470	0.862 / 3.984
Line lamp	**0.101** / **1.339**	0.131 / 1.526	0.546 / 5.207	0.107 / 1.375	0.632 / 45.365	0.629 / 7.175
Line phone	**0.147** / **1.364**	0.211 / 1.612	0.696 / 4.565	0.161 / 1.408	1.166 / 112.223	0.783 / 4.207
Occl	**0.184** / **1.592**	0.252 / 1.922	0.702 / 12.181	0.198 / 1.704	1.585 / 184.307	0.845 / 4.755
All	**0.150** / **1.386**	0.191 / 1.592	0.655 / 6.048	0.162 / 1.451	1.058 / 108.239	0.745 / 4.141

Points and Lines	CvxPnPL °/‰	DLT °/‰	EPnPL °/‰	OPnPL °/‰
Line ape	0.193 / **1.418**	2.681 / 9.329	0.201 / 1.538	**0.185** / 1.420
Line benchvise	0.086 / 1.258	0.304 / 1.941	0.088 / 1.277	**0.085** / **1.255**
Line bowl	**0.108** / 1.328	0.883 / 5.896	0.125 / 1.479	0.108 / **1.322**
Line cam	0.088 / 1.262	0.520 / 2.417	0.089 / 1.271	**0.081** / **1.249**
Line can	0.085 / 1.258	0.394 / 2.014	0.088 / 1.273	**0.082** / **1.254**
Line cat	0.107 / 1.311	0.804 / 3.712	0.114 / 1.347	**0.104** / **1.303**
Line cup	0.144 / 1.386	1.549 / 9.045	0.142 / 1.419	**0.126** / **1.347**
Line driller	0.091 / 1.285	0.347 / 2.523	0.109 / 1.363	**0.088** / **1.282**
Line duck	0.161 / 1.343	1.562 / 5.890	0.170 / 1.408	**0.152** / **1.320**
Line eggbox	0.104 / 1.284	0.725 / 2.647	0.112 / 1.296	**0.098** / **1.276**
Line glue	**0.126** / **1.382**	1.239 / 4.600	0.170 / 1.571	0.130 / 1.411
Line holepuncher	0.124 / 1.339	1.172 / 6.522	0.137 / 1.416	**0.118** / **1.322**
Line iron	0.098 / 1.277	0.457 / 2.018	0.111 / 1.314	**0.098** / **1.276**
Line lamp	0.083 / 1.281	0.273 / 2.302	0.089 / 1.323	**0.083** / **1.272**
Line phone	0.105 / 1.301	0.590 / 2.728	0.119 / 1.346	**0.102** / **1.289**
Occl	0.143 / 1.494	1.423 / 6.542	0.160 / 1.596	**0.135** / **1.441**
All	0.115 / 1.324	0.834 / 4.029	0.126 / 1.385	**0.111** / **1.314**

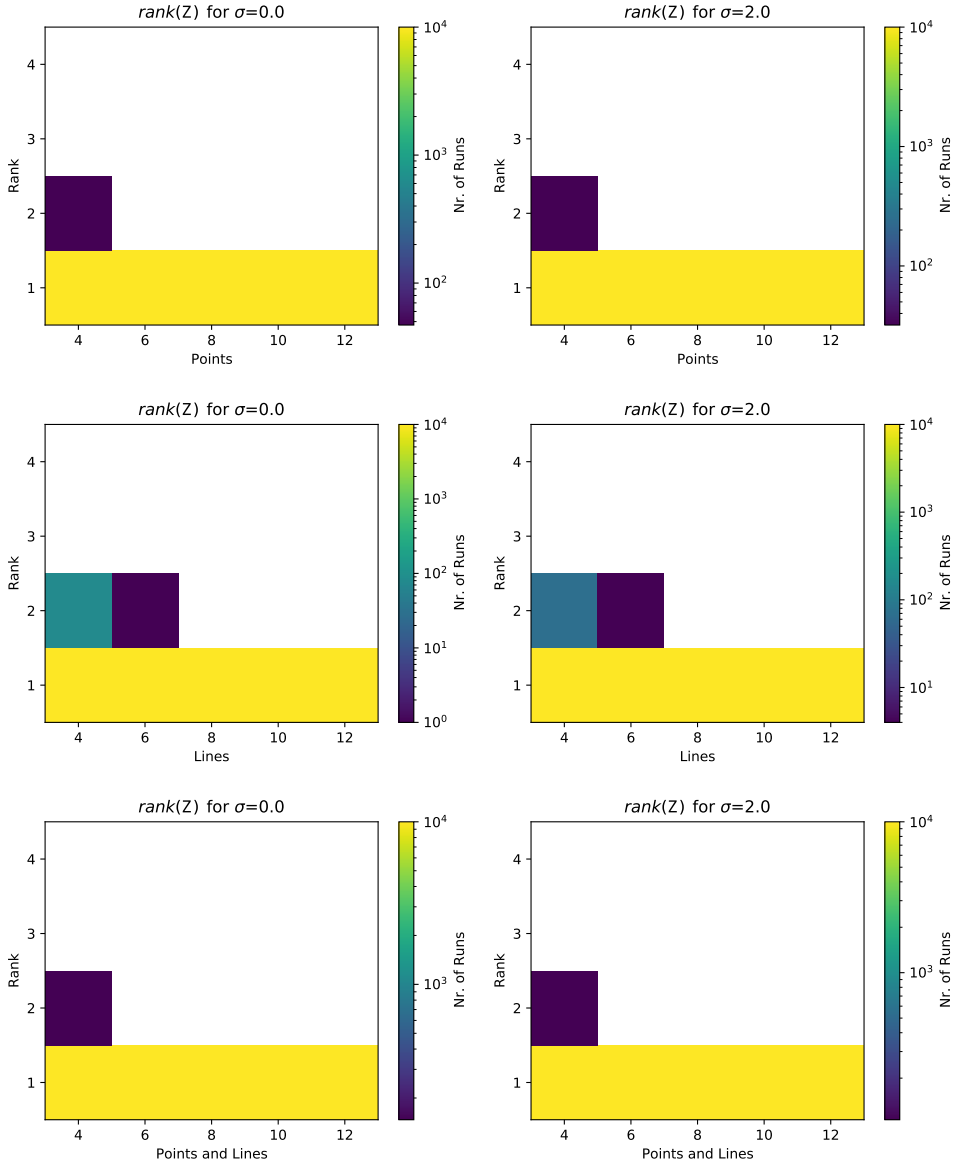

Fig. 6: A visual representation of the different values of rank(Z) over 10000 runs for a PnP scenario with 4+ points (top), PnL scenario with 4+ lines (middle) and a PnPL scenario with 4+ points and lines (bottom), when the convex solver is allowed a maximum number of *2500000* iterations, a 1000x increase in iterations compared to Figure 5. We display the rank value under noiseless conditions (left) and with added Gaussian pixel noise with standard deviation $\sigma = 2$. The color coding is log-scaled (figure best seen in colour). The absence of color (white) in some locations, signifies that there were no runs for a given number of points/lines (horizontal axis) that produced a solution with that rank (vertical axis).

8 Conclusion

We introduced the first convex approach to the central absolute problem from mixed point and line correspondences. We formulated our optimization problem as a QCQP and relaxed it into a SDP. Through empirical observation, we established that the rank of the relaxed solution has strong connection to the number of ambiguous poses in the problem, being equal in most situations. We then derived approaches for retrieving all poses in situations up to rank 4. We showed that our method is competitive with the best state-of-the-art algorithms under synthetic conditions and qualitatively validated its performance on a real dataset.

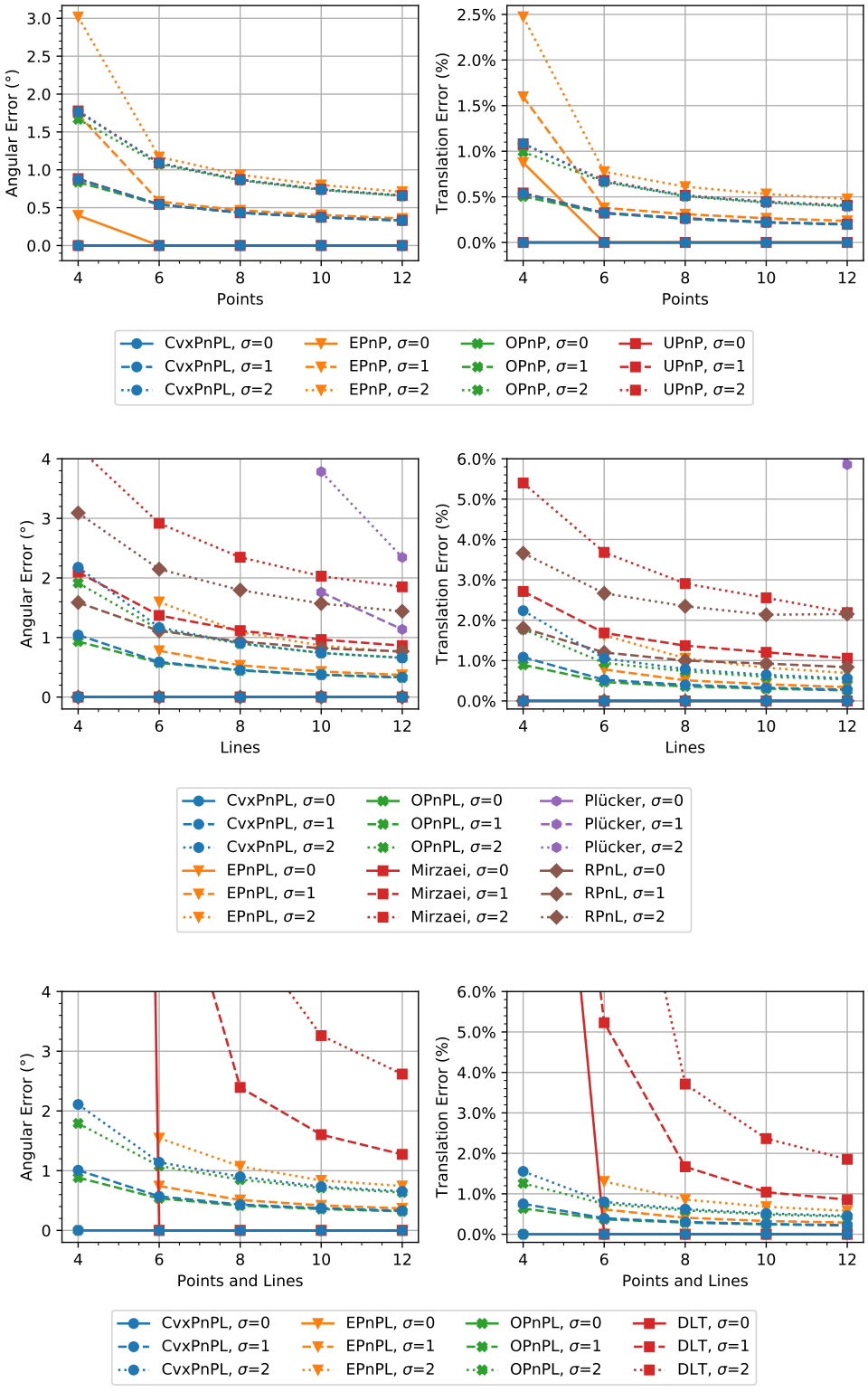

Fig. 7: Comparison between different methods for a PnP scenario with 4+ points (top), PnL scenario with 4+ lines (middle) and a PnPL scenario with 4+ points and lines (bottom). The median angular and translation errors are shown on the left and right, respectively, for different numbers of elements in the scenario, employing Gaussian pixel noise with standard deviation σ computed over 10000 runs (figure best seen in colour).

Fig. 8: Two views from the Occlusion dataset, showing the ability of our method to handle real scenes. The key points and lines used as correspondences are represented in white (figure best seen in colour).

Acknowledgements The authors would like to thank Jacopo Cavazza and all present at the Optimization Methods in Geometric Vision seminar at the 2019 NII Shonan Meetings, for their suggestions and insightful discussions.

References

1. Abdel-Aziz, Y., Karara, H.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In: Proceedings of the Symposium on Close-Range Photogrammetry, pp. 1–18. ASP (1971)
2. Ansar, A., Daniilidis, K.: Linear pose estimation from points or lines. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(5), 578–589 (2003)
3. Boyd, S., Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge university press (2004)
4. Boyd, S., Parikh, N., Chu, E.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Now Publishers Inc (2011)
5. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6D object pose estimation using 3D object coordinates. In: European Conference on Computer Vision (ECCV), pp. 536–551. Springer (2014)
6. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
7. Briales, J., Gonzalez-Jimenez, J.: Convex global 3D registration with lagrangian duality. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
8. Briales, J., Kneip, L., Gonzalez-Jimenez, J.: A certifiably globally optimal solution to the non-minimal relative pose problem. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
9. Carlone, L., Calafiore, G.C., Tommolillo, C., Dellaert, F.: Planar pose graph optimization: Duality, optimal solutions, and verification. IEEE Transactions on Robotics **32**(3), 545–565 (2016)
10. Chen, H.H.: Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. In: Third International Conference on Computer Vision, pp. 374–378. IEEE (1990)
11. DeMenthon, D., Davis, L.S.: Exact and approximate solutions of the perspective-three-point problem. IEEE Transactions on Pattern Analysis & Machine Intelligence **14**(11), 1100–1105 (1992)
12. Dhome, M., Richetin, M., Lapreste, J.T., Rives, G.: Determination of the attitude of 3D objects from a single perspective view. IEEE Transactions on Pattern Analysis and Machine Intelligence **11**(12), 1265–1278 (1989)
13. Ferraz, L., Binefa, X., Moreno-Noguer, F.: Very fast solution to the PnP problem with algebraic outlier rejection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
14. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(8), 930–943 (2003)
15. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge university press (2003)
16. Hesch, J.A., Roumeliotis, S.I.: A direct least-squares (DLS) method for PnP. In: 2011 International Conference on Computer Vision, pp. 383–390. IEEE (2011)
17. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: Asian Conference on Computer Vision, pp. 548–562. Springer (2012)
18. Khoo, Y., Kapoor, A.: Non-iterative rigid 2D/3D point-set registration using semidefinite programming. IEEE Transactions on Image Processing **25**(7), 2956–2970 (2016)
19. Kneip, L., Furgale, P.: Opengv: A unified and generalized approach to real-time calibrated geometric vision. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2014)
20. Kneip, L., Li, H., Seo, Y.: Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In: European Conference on Computer Vision (ECCV), pp. 127–142. Springer (2014)
21. Kuang, Y., Astrom, K.: Pose estimation with unknown focal length using points, directions and lines. In: IEEE International Conference on Computer Vision (ICCV) (2013)
22. Kukelova, Z., Heller, J., Fitzgibbon, A.: Efficient intersection of three quadrics and applications in computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

23. Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o (n) solution to the pnp problem. International Journal of Computer Vision **81**(2), 155 (2009)
24. Li, S., Xu, C., Xie, M.: A robust o (n) solution to the perspective-n-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(7), 1444–1450 (2012)
25. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)
26. Marchand, E., Uchiyama, H., Spindler, F.: Pose estimation for augmented reality: a hands-on survey. IEEE Transactions on Visualization and Computer Graphics **22**(12), 2633–2651 (2016)
27. Mirzaei, F.M., Roumeliotis, S.I.: Globally optimal pose estimation from line correspondences. In: 2011 IEEE International Conference on Robotics and Automation, pp. 5581–5588. IEEE (2011)
28. Nesterov, Y., Wolkowicz, H., Ye, Y.: Semidefinite programming relaxations of nonconvex quadratic optimization. In: Handbook of Semidefinite Programming, pp. 361–419. Springer (2000)
29. O'Donoghue, B., Chu, E., Parikh, N., Boyd, S.: Conic optimization via operator splitting and homogeneous self-dual embedding. Journal of Optimization Theory and Applications **169**(3), 1042–1068 (2016). URL `http://stanford.edu/~boyd/papers/scs.html`
30. Přibyl, B., Zemčík, P., Čadík, M.: Camera pose estimation from lines using plücker coordinates. In: British Machine Vision Conference (2015)
31. Přibyl, B., Zemčík, P., Čadík, M.: Absolute pose estimation from line correspondences using direct linear transformation. Computer Vision and Image Understanding **161**, 130–144 (2017)
32. Ramalingam, S., Bouaziz, S., Sturm, P.: Pose estimation using both points and lines for geo-localization. In: 2011 IEEE International Conference on Robotics and Automation, pp. 4716–4723. IEEE (2011)
33. Rosen, D.M., Carlone, L., Bandeira, A.S., Leonard, J.J.: Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. The International Journal of Robotics Research **38**(2-3), 95–125 (2019)
34. Rosen, D.M., DuHadway, C., Leonard, J.J.: A convex relaxation for approximate global optimization in simultaneous localization and mapping. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 5822–5829. IEEE (2015)
35. Saunderson, J., Parrilo, P.A., Willsky, A.S.: Semidefinite descriptions of the convex hull of rotation matrices. SIAM Journal on Optimization **25**(3), 1314–1343 (2015)
36. Schacke, K.: On the kronecker product. Master's thesis, University of Waterloo (2004)
37. Schweighofer, G., Pinz, A.: Globally optimal o (n) solution to the pnp problem for general camera models. In: British Machine Vision Conference, pp. 1–10 (2008)
38. Shor, N.Z.: Quadratic optimization problems. Soviet Journal of Computer and Systems Sciences **25**, 1–11 (1987)
39. Tron, R., Rosen, D.M., Carlone, L.: On the inclusion of determinant constraints in lagrangian duality for 3d slam. In: Robotics: Science and Systems (RSS) in the workshop "The Problem of Mobile Sensors" (2015)
40. Vakhitov, A., Funke, J., Moreno-Noguer, F.: Accurate and linear time pose estimation from points and lines. In: European Conference on Computer Vision (ECCV), pp. 583–599. Springer (2016)
41. Von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: Lsd: A fast line segment detector with a false detection control. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(4), 722–732 (2010)
42. Xu, C., Zhang, L., Cheng, L., Koch, R.: Pose estimation from line correspondences: A complete analysis and a series of solutions. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1209–1222 (2017)
43. Yang, H., Carlone, L.: A polynomial-time solution for robust registration with extreme outlier rates. arXiv preprint arXiv:1903.08588 (2019)
44. Ye, Y., Todd, M.J., Mizuno, S.: An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. Mathematics of operations research **19**(1), 53–67 (1994)
45. Zhang, L., Xu, C., Lee, K.M., Koch, R.: Robust and efficient pose estimation from line correspondences. In: Asian Conference on Computer Vision, pp. 217–230. Springer (2012)
46. Zheng, Y., Kuang, Y., Sugimoto, S., Åström, K., Okutomi, M.: Revisiting the pnp problem: A fast, general and optimal solution. In: IEEE International Conference on Computer Vision (ICCV) (2013)
47. Zhou, L., Yang, Y., Abello, M., Kaess, M.: A robust and efficient algorithm for the pnl problem using algebraic distance to approximate the reprojection distance. In: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (2019)
48. Zhou, L., Ye, J., Kaess, M.: A Stable Algebraic Camera Pose Estimation for Minimal Configurations of 2D/3D Point and Line Correspondences. In: Asian Conference on Computer Vision (2018)

Appendix A Useful Vectorization Properties

See [36] for an insightful discussion on the following identities:

$$\operatorname{vec}(\mathtt{AXB}) = (\mathtt{B}^\top \otimes \mathtt{A}) \operatorname{vec}(\mathtt{X}) \tag{26}$$

$$\operatorname{tr}(\mathtt{A}^\top \mathtt{X}^\top \mathtt{BY}) = \operatorname{vec}(\mathtt{X})^\top (\mathtt{A} \otimes \mathtt{B}) \operatorname{vec}(\mathtt{Y}), \tag{27}$$

where the operator $\otimes$ represents the Kronecker product.

Appendix B Geometric Constraints

This section contains the detailed derivations showing how to compose the equations systems from §3.

B.1 Point Correspondences

Consider $\mathbf{b}_i$ as the bearing vector associated with the 3D point $\mathbf{p}_i$ (refer to Figure 1). The transformation from the model/object space to the camera is parameterized by the rotation matrix $\mathtt{R}$ and the translation vector $\mathbf{t}$. The collinearity constraint of a point allows us to write,

$$\lfloor \mathbf{b}_i \rfloor_\times (\mathtt{R}\mathbf{p}_i + \mathbf{t}) = 0, \tag{28}$$

where $\lfloor \mathbf{b} \rfloor_\times$ is the skew symmetric matrix representation of the 3D vector $\mathbf{b}$

$$\lfloor \mathbf{b} \rfloor_\times = \begin{bmatrix} 0 & -b_z & b_y \\ b_z & 0 & -b_x \\ -b_y & b_x & 0 \end{bmatrix}. \tag{29}$$

We can represent Eq. (28) isolating the vectorized representation of $\mathtt{R}$,

$$\mathbf{r} = \operatorname{vec}(\mathtt{R}), \tag{30}$$

as

$$\lfloor \mathbf{b}_i \rfloor_\times (\mathtt{R}\mathbf{p}_i + \mathbf{t}) = 0 \tag{31}$$

$$\lfloor \mathbf{b}_i \rfloor_\times \mathtt{R}\mathbf{p}_i + \lfloor \mathbf{b}_i \rfloor_\times \mathbf{t} = 0 : \text{(distributive prop.)} \tag{32}$$

$$\underbrace{(\mathbf{p}_i^\top \otimes \lfloor \mathbf{b}_i \rfloor_\times)}_{\mathtt{C}_{\mathtt{p}i}} \mathbf{r} + \underbrace{\lfloor \mathbf{b}_i \rfloor_\times}_{\mathtt{N}_{\mathtt{p}i}} \mathbf{t} = 0. : \text{(applying (26))} \tag{33}$$

Matrices $\mathtt{C}_{\mathtt{p}i}$ and $\mathtt{N}_{\mathtt{p}i}$ are 3×9 and 3×2, respectively. Each point correspondence contributes 3 equations and upon stacking all n points yields,

$$\mathtt{C}_\mathtt{p} = \begin{bmatrix} \mathtt{C}_{\mathtt{p}1} \\ \vdots \\ \mathtt{C}_{\mathtt{p}n} \end{bmatrix}, \quad \mathtt{N}_\mathtt{p} = \begin{bmatrix} \mathtt{N}_{\mathtt{p}1} \\ \vdots \\ \mathtt{N}_{\mathtt{p}n} \end{bmatrix}, \tag{34}$$

which are $3n \times 9$ and $3n \times 3$ sized matrices, respectively. This ultimately results in the homogeneous system of equations

$$\mathtt{C}_\mathtt{p}\mathbf{r} + \mathtt{N}_\mathtt{p}\mathbf{t} = 0. \tag{35}$$

B.2 Line Correspondences

Consider a 3D line defined by two points, which we represent by the tuple $(\mathbf{l}_{\mathbf{p}1j}, \mathbf{l}_{\mathbf{p}2j})$. Also consider $\mathbf{l}_{\mathbf{n}j}$, the normal to the plane formed between the 3D line and the origin of the camera. From the coplanarity constraints of the line, we have

$$\mathbf{l}_{\mathbf{n}j}^\top (\mathtt{R}\mathbf{l}_{\mathbf{p}ij} + \mathbf{t}) = 0 \quad : i = 1, 2. \tag{36}$$

and from Eq. (36),

$$\mathbf{l}_{\mathbf{n}j}^\top (\mathtt{R}\mathbf{l}_{\mathbf{p}ij} + \mathbf{t}) = 0 \tag{37}$$

$$\mathbf{l}_{\mathbf{n}j}^\top \mathtt{R}\mathbf{l}_{\mathbf{p}ij} + \mathbf{l}_{\mathbf{n}j}^\top \mathbf{t} = 0 : \text{(distributive property)} \tag{38}$$

$$(\mathbf{l}_{\mathbf{p}ij}^\top \otimes \mathbf{l}_{\mathbf{n}j}^\top)\mathbf{r} + \mathbf{l}_{\mathbf{n}j}^\top \mathbf{t} = 0 : \text{(applying (26))} \tag{39}$$

$$\underbrace{(\mathbf{l}_{\mathbf{p}ij} \otimes \mathbf{l}_{\mathbf{n}j})^\top}_{\mathbf{c}_{\mathbf{l}ij}} \mathbf{r} + \mathbf{l}_{\mathbf{n}j}^\top \mathbf{t} = 0. : \text{(single out } ^\top) \tag{40}$$

The vector $\mathbf{c}_{\mathbf{l}ij} \in \mathbb{R}^9$. Stacking the contributions from all m lines yields

$$\mathtt{C}_\mathtt{l} = \begin{bmatrix} \mathbf{c}_{\mathbf{l}11}^\top \\ \mathbf{c}_{\mathbf{l}21}^\top \\ \vdots \\ \mathbf{c}_{\mathbf{l}1m}^\top \\ \mathbf{c}_{\mathbf{l}2m}^\top \end{bmatrix}, \quad \mathtt{N}_\mathtt{l} = \begin{bmatrix} \mathbf{l}_{\mathbf{n}1}^\top \\ \mathbf{l}_{\mathbf{n}1}^\top \\ \vdots \\ \mathbf{l}_{\mathbf{n}m}^\top \\ \mathbf{l}_{\mathbf{n}m}^\top \end{bmatrix}, \tag{41}$$

which are matrices of size $2m \times 9$ and $2m \times 3$, respectively. The full system of equations assumes the form

$$\mathtt{C}_\mathtt{l}\mathbf{r} + \mathtt{N}_\mathtt{l}\mathbf{t} = 0. \tag{42}$$

Appendix C The QCQP Reformulation

The reformulation from QCQP problem (7) into its canonical form (8) is required in order to leverage conventional convex solvers. In the next sections, we address this reformulation for each element of the problem. Out of convenience, we resort to the homogenized representation of $\mathbf{r}$,

$$\tilde{\mathbf{r}} = \begin{bmatrix} \mathbf{r} \\ 1 \end{bmatrix}, \tag{43}$$

as it eases bundling all constant and linear terms with respect to $\mathbf{r}$, into a quadratic form with respect to $\tilde{\mathbf{r}}$.

C.1 Cost Function

The cost function is defined by

$$\|A\mathbf{r}\|^2 = (A\mathbf{r})^\top (A\mathbf{r}) \quad : (\|\mathbf{v}\|^2 = \mathbf{v}^\top \mathbf{v} \text{ for } \mathbf{v} \in \mathbb{R}^n) \tag{44}$$

$$= \mathbf{r}^\top A^\top A \mathbf{r} \quad : \text{(transpose of product)} \tag{45}$$

$$= \tilde{\mathbf{r}}^\top \underbrace{\begin{bmatrix} A^\top A & 0_{9\times 1} \\ 0_{1\times 9} & 0 \end{bmatrix}}_{Q_0} \tilde{\mathbf{r}}. \quad : \text{(replace } \mathbf{r} \text{ for } \tilde{\mathbf{r}}) \tag{46}$$

C.2 Orthogonality of Rows

The constraint associated with the orthogonality of rows provides 6 linearly independent equations,

$$\mathtt{R}\mathtt{R}^\top = \mathtt{I}_3, \tag{47}$$

where the matrix $\mathtt{I}_3$ represents the identity of size 3×3. We introduce the unit vector $\mathbf{e}_i \in \mathbb{R}^3$, whose component i is set to 1 and the remainder to 0, as well as the 3×3 matrix

$$\mathtt{E}_{ij} = \mathbf{e}_i \mathbf{e}_j^\top, \tag{48}$$

which is composed of a unique element 1 at row i and column j, while all the remaining ones are 0. Equipped with these new definitions, we are able to express the constraint in (47) with respect to each individual component. In general, for a given matrix $\mathtt{A} \in \mathbb{R}^{3\times 3}$ we have that:

$$\mathbf{e}_i^\top \mathtt{A} \mathbf{e}_j = a_{ij}. \tag{49}$$

Employing the same property we have

$$\begin{aligned}
\mathbf{e}_i^\top(\mathtt{R}\mathtt{R}^\top - \mathtt{I}_3)\mathbf{e}_j &= 0 && (50)\\
\mathbf{e}_i^\top \mathtt{R}\mathtt{R}^\top \mathbf{e}_j - \mathbf{e}_i^\top \mathtt{I}_3 \mathbf{e}_j &= 0 && : \text{(distributive prop.)} \quad (51)\\
\mathbf{e}_i^\top \mathtt{R}\mathtt{R}^\top \mathbf{e}_j - \delta_{ij} &= 0 && : \text{(Kronecker delta)} \quad (52)\\
\mathrm{tr}(\mathbf{e}_i^\top \mathtt{R}\mathtt{R}^\top \mathbf{e}_j) - \delta_{ij} &= 0 && : \text{(trace of a scalar)} \quad (53)\\
\mathrm{tr}(\mathtt{R}^\top \mathbf{e}_j \mathbf{e}_i^\top \mathtt{R}) - \delta_{ij} &= 0 && : \text{(trace cyclic prop.)} \quad (54)\\
\mathrm{tr}(\mathtt{R}^\top \mathtt{E}_{ji} \mathtt{R}) - \delta_{ij} &= 0 && : \text{(substitution of } \mathtt{E}_{ji}) \quad (55)\\
\mathrm{tr}(\mathtt{I}_3 \mathtt{R}^\top \mathtt{E}_{ji} \mathtt{R}) - \delta_{ij} &= 0 && : \text{(identity matrix)} \quad (56)\\
\mathbf{r}^\top(\mathtt{I}_3 \otimes \mathtt{E}_{ji})\mathbf{r} - \delta_{ij} &= 0 && : \text{(applying (27))} \quad (57)\\
\tilde{\mathbf{r}}^\top \underbrace{\begin{bmatrix} \mathtt{I}_3 \otimes \mathtt{E}_{ji} & 0_{9\times 1} \\ 0_{1\times 9} & -\delta_{ij} \end{bmatrix}}_{\mathtt{Q}_{\mathtt{r}ij}} \tilde{\mathbf{r}} &= 0. && : \text{(with respect to } \tilde{\mathbf{r}}) \quad (58)
\end{aligned}$$

Iterating for all indexes, we compose 6 constraints:

$$\tilde{\mathbf{r}}^\top \mathtt{Q}_{\mathtt{r}ij} \tilde{\mathbf{r}} = 0 \quad : i = 1,\ldots,3; j = i,\ldots,3. \tag{59}$$

C.3 Orthogonality of Columns

The derivation is very similar to Sec. C.2. This time we start from the following constraint:

$$\begin{aligned}
\mathbf{e}_i^\top(\mathtt{R}^\top \mathtt{R} - \mathtt{I}_3)\mathbf{e}_j &= 0 && (60)\\
\mathbf{e}_i^\top \mathtt{R}^\top \mathtt{R} \mathbf{e}_j - \mathbf{e}_i^\top \mathtt{I}_3 \mathbf{e}_j &= 0 && : \text{(distributive prop.)} \quad (61)\\
\mathbf{e}_i^\top \mathtt{R}^\top \mathtt{R} \mathbf{e}_j - \delta_{ij} &= 0 && : \text{(Kronecker delta)} \quad (62)\\
\mathrm{tr}(\mathbf{e}_i^\top \mathtt{R}^\top \mathtt{R} \mathbf{e}_j) - \delta_{ij} &= 0 && : \text{(trace of a scalar)} \quad (63)\\
\mathrm{tr}(\mathbf{e}_j \mathbf{e}_i^\top \mathtt{R}^\top \mathtt{R}) - \delta_{ij} &= 0 && : \text{(trace cyclic prop.)} \quad (64)\\
\mathrm{tr}(\mathtt{E}_{ji} \mathtt{R}^\top \mathtt{R}) - \delta_{ij} &= 0 && : \text{(substitution of } \mathtt{E}_{ji}) \quad (65)\\
\mathrm{tr}(\mathtt{E}_{ij}^\top \mathtt{R}^\top \mathtt{R}) - \delta_{ij} &= 0 && : \text{(transpose } \mathtt{E}_{ji}) \quad (66)\\
\mathrm{tr}(\mathtt{E}_{ij}^\top \mathtt{R}^\top \mathtt{I}_3 \mathtt{R}) - \delta_{ij} &= 0 && : \text{(identity matrix)} \quad (67)\\
\mathbf{r}^\top(\mathtt{E}_{ij} \otimes \mathtt{I}_3)\mathbf{r} - \delta_{ij} &= 0 && : \text{(applying (27))} \quad (68)\\
\tilde{\mathbf{r}}^\top \underbrace{\begin{bmatrix} \mathtt{E}_{ij} \otimes \mathtt{I}_3 & 0_{9\times 1} \\ 0_{1\times 9} & -\delta_{ij} \end{bmatrix}}_{\mathtt{Q}_{\mathtt{c}ij}} \tilde{\mathbf{r}} &= 0. && : \text{(with respect to } \tilde{\mathbf{r}}) \quad (69)
\end{aligned}$$

Iterating for all indexes composes 6 constraints as:

$$\tilde{\mathbf{r}}^\top \mathtt{Q}_{\mathtt{c}ij} \tilde{\mathbf{r}} = 0 \quad : i = 1,\ldots,3; j = i,\ldots,3. \tag{70}$$

Appendix D Composing the Linear System of Quadrics

In § 5 we state that, under minimal configurations, the solution space of all admissible solutions is given by the linear decomposition

$$\mathbf{r} = \sum_{k=1}^{K-1} \alpha'_k \mathbf{v}'_k + \mathbf{v}'_0, \tag{71}$$

where K is the rank of matrix $\mathtt{Z}$. Consider now the case $K = 4$, which is the highest rank we address in this paper. The three unknowns α'_1, α'_2 and α'_3 will be designated by the letters a, b and c and we will drop the $'$ superscript on the vectors for convenience. We resort to the inverse of the vectorizing operator vec^{-1} to reformulate Eq. (71). Defining

$$\mathtt{V}_i = \mathrm{vec}^{-1}(\mathbf{v}_i) : i = 0,\ldots,3, \tag{72}$$

we can rewrite it as

$$\mathtt{R} = a\mathtt{V}_1 + b\mathtt{V}_2 + c\mathtt{V}_3 + \mathtt{V}_0. \tag{73}$$

Once more, for $\mathtt{R}$ to be valid, it needs to respect the constraints in Eqs. (7b), (7c) and (7d). In the next sections we will show how to rewrite this combined set of 21 constraints as a linear system

$$\mathtt{A}\left[a^2, b^2, c^2, ab, ac, bc, a, b, c, 1\right]^\top = 0, \tag{74}$$

where $\mathtt{A}$ is a matrix of size 21×10.

D.1 Reformulating the Quadratic System of Equations as a Linear System with Respect to Quadratic Terms

One important aspect of this formulation is recognizing how to convert the quadratic system to its "linear" form, with respect to quadratic terms. Consider the vector

$$\mathbf{v} = \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix}, \tag{75}$$

and the quadratic expression

$$\mathbf{v}^\top \mathtt{P} \mathbf{v} = 0, \tag{76}$$

where $\mathtt{P} \in \mathbb{R}^{4\times4}$. We can reformulate Eq. (76) to its "linear" form as

$$\begin{bmatrix} P_{11} \\ P_{22} \\ P_{33} \\ P_{12}+P_{21} \\ P_{13}+P_{31} \\ P_{23}+P_{32} \\ P_{14}+P_{41} \\ P_{24}+P_{42} \\ P_{34}+P_{43} \\ P_{44} \end{bmatrix}^\top \begin{bmatrix} a^2 \\ b^2 \\ c^2 \\ ab \\ ac \\ bc \\ a \\ b \\ c \\ 1 \end{bmatrix} = 0. \tag{77}$$

In the next sections we will focus once more on describing all constraints in their natural quadratic form, always with the outlook that the previous reformulation can be applied and that each quadratic constraint will contribute a row in the final linear system in Eq. (74).

D.2 Orthogonality of Columns

We start by writing Eq. (71) in a more compact linear form

$$\mathbf{r} = \mathtt{V}\mathbf{q}, \tag{78}$$

where

$$\mathtt{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \mathbf{v}_0 \end{bmatrix} \tag{79}$$

$$\mathbf{q} = \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix}. \tag{80}$$

The first important step is to define an operation which allows us to select each column of $\mathtt{R}$. Using the index i to designate the desired column, we can write

$$\mathbf{r}_{\mathbf{c}i} = (\mathbf{e}_i \otimes \mathtt{I}_3)^\top \mathbf{r}. \tag{81}$$

With this mechanism in place, orthogonality of the columns reads

$$\mathbf{r}_{\mathbf{c}_i}^\top \mathbf{r}_{\mathbf{c}j} = \delta_{ij} \quad \text{for} \quad i = \{1,2,3\}, j = \{i,\ldots,3\}. \tag{82}$$

Substituting the appropriate terms

$$\mathbf{r}_{\mathbf{c}_i}^\top \mathbf{r}_{\mathbf{c}j} - \delta_{ij} = 0 \tag{83}$$

$$\mathbf{r}^\top (\mathbf{e}_i \otimes \mathtt{I}_3)(\mathbf{e}_j \otimes \mathtt{I}_3)^\top \mathbf{r} - \delta_{ij} = 0 : \text{(sub. (81))} \tag{84}$$

$$\mathbf{q}^\top \mathtt{V}^\top (\mathbf{e}_i \otimes \mathtt{I}_3)(\mathbf{e}_j \otimes \mathtt{I}_3)^\top \mathtt{V}\mathbf{q} - \delta_{ij} = 0 : \text{(sub. (78))} \tag{85}$$

$$\mathbf{q}^\top \mathtt{V}^\top (\mathbf{e}_i \otimes \mathtt{I}_3)(\mathbf{e}_j \otimes \mathtt{I}_3)^\top \mathtt{V}\mathbf{q} \tag{86}$$

$$-\mathbf{q}^\top \begin{bmatrix} 0_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & \delta_{ij} \end{bmatrix} \mathbf{q} = 0. \tag{87}$$

Considering all indices

$$\mathtt{P}_{\mathtt{c}ij} = \mathtt{V}^\top (\mathbf{e}_i \otimes \mathtt{I}_3)(\mathbf{e}_j \otimes \mathtt{I}_3)^\top \mathtt{V} - \begin{bmatrix} 0_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & \delta_{ij} \end{bmatrix} \tag{88}$$

$$\mathbf{q}^\top \mathtt{P}_{\mathtt{c}ij} \mathbf{q} = 0 \quad \text{for} \quad i = \{1,2,3\}, j = \{i,\ldots,3\}, \tag{89}$$

which contributes 6 equations to the system in Eq. (74).

D.3 Orthogonality of Rows

Similarly to the previous section, we start by defining a selector operator which allows us to isolate the rows of $\mathtt{R}$ from $\mathbf{r}$. Not surprisingly, we can verify that this operator can be built by commuting both terms in the Kronecker product.

$$\mathbf{r}_{\mathbf{r}i} = (\mathtt{I}_3 \otimes \mathbf{e}_i)^\top \mathbf{r}. \tag{90}$$

With this operator in place, orthogonality of the rows reads

$$\mathbf{r}_{\mathbf{r}_i}^\top \mathbf{r}_{\mathbf{r}j} = \delta_{ij} \quad \text{for} \quad i = \{1,2,3\}, j = \{i,\ldots,3\}. \tag{91}$$

Substituting the appropriate terms

$$\mathbf{r}_{\mathbf{r}_i}^\top \mathbf{r}_{\mathbf{r}j} - \delta_{ij} = 0 \tag{92}$$

$$\mathbf{r}^\top (\mathtt{I}_3 \otimes \mathbf{e}_i)(\mathtt{I}_3 \otimes \mathbf{e}_i)^\top \mathbf{r} - \delta_{ij} = 0 : \text{(sub. (90))} \tag{93}$$

$$\mathbf{q}^\top \mathtt{V}^\top (\mathtt{I}_3 \otimes \mathbf{e}_i)(\mathtt{I}_3 \otimes \mathbf{e}_i)^\top \mathtt{V}\mathbf{q} - \delta_{ij} = 0 : \text{(sub. (78))} \tag{94}$$

$$\mathbf{q}^\top \mathtt{V}^\top (\mathtt{I}_3 \otimes \mathbf{e}_i)(\mathtt{I}_3 \otimes \mathbf{e}_i)^\top \mathtt{V}\mathbf{q} \tag{95}$$

$$-\mathbf{q}^\top \begin{bmatrix} 0_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & \delta_{ij} \end{bmatrix} \mathbf{q} = 0. \tag{96}$$

Considering all indices

$$\mathtt{P}_{\mathtt{r}ij} = \mathtt{V}^\top (\mathtt{I}_3 \otimes \mathbf{e}_i)(\mathtt{I}_3 \otimes \mathbf{e}_i)^\top \mathtt{V} - \begin{bmatrix} 0_{3\times3} & 0_{3\times1} \\ 0_{1\times3} & \delta_{ij} \end{bmatrix} \tag{97}$$

$$\mathbf{q}^\top \mathtt{P}_{\mathtt{r}ij} \mathbf{q} = 0 \quad \text{for} \quad i = \{1,2,3\}, j = \{i,\ldots,3\}, \tag{98}$$

which contributes 6 additional equations to the system in Eq. (74).

D.4 Determinant - Right Hand Convention

Reusing the column selector operator from Eq. (81), the right-hand convention specifies that

$$\mathbf{r}_{\mathbf{c}i} \times \mathbf{r}_{\mathbf{c}j} = \mathbf{r}_{\mathbf{c}k}, \tag{99}$$

for $(i,j,k) \in \{(1,2,3),(2,3,1),(3,1,2)\}$. We resort to the unit vector $\mathbf{e}_l$ to specify the constraint with respect to each individual component. We will also exploit the cross product identity,

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = \mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}). \tag{100}$$

Starting with

$$\mathbf{e}_l^\top (\mathbf{r}_{\mathbf{c}i} \times \mathbf{r}_{\mathbf{c}j} - \mathbf{r}_{\mathbf{c}k}) = 0 \tag{101}$$
$$\mathbf{e}_l^\top (\mathbf{r}_{\mathbf{c}i} \times \mathbf{r}_{\mathbf{c}j}) - \mathbf{e}_l^\top \mathbf{r}_{\mathbf{c}k} = 0 : \text{(dist. prop.)} \tag{102}$$
$$\mathbf{r}_{\mathbf{c}j}^\top (\mathbf{e}_l \times \mathbf{r}_{\mathbf{c}i}) - \mathbf{e}_l^\top \mathbf{r}_{\mathbf{c}k} = 0 : \text{(using (100))} \tag{103}$$
$$\mathbf{r}_{\mathbf{c}j}^\top \lfloor \mathbf{e}_l \rfloor_\times \mathbf{r}_{\mathbf{c}i} - \mathbf{e}_l^\top \mathbf{r}_{\mathbf{c}k} = 0 : \text{(skew sym.)} \tag{104}$$
$$\mathbf{r}^\top (\mathbf{e}_j \otimes \mathtt{I}_3) \lfloor \mathbf{e}_l \rfloor_\times (\mathbf{e}_i \otimes \mathtt{I}_3)^\top \mathbf{r} \tag{105}$$
$$-\mathbf{e}_l^\top (\mathbf{e}_k \otimes \mathtt{I}_3)^\top \mathbf{r} = 0 : \text{(sub. (81))} \tag{106}$$
$$\mathbf{q}^\top \mathtt{V}^\top (\mathbf{e}_j \otimes \mathtt{I}_3) \lfloor \mathbf{e}_l \rfloor_\times (\mathbf{e}_i \otimes \mathtt{I}_3)^\top \mathtt{V} \mathbf{q} \tag{107}$$
$$-\mathbf{e}_l^\top (\mathbf{e}_k \otimes \mathtt{I}_3)^\top \mathtt{V} \mathbf{q} = 0 : \text{(sub. (78))} \tag{108}$$
$$\mathbf{q}^\top \mathtt{V}^\top (\mathbf{e}_j \otimes \mathtt{I}_3) \lfloor \mathbf{e}_l \rfloor_\times (\mathbf{e}_i \otimes \mathtt{I}_3)^\top \mathtt{V} \mathbf{q} \tag{109}$$
$$-\mathbf{q}^\top \begin{bmatrix} 0_{3\times 3} & 0_{3\times 1} \\ \mathbf{e}_l^\top (\mathbf{e}_k \otimes \mathtt{I}_3)^\top & 0 \end{bmatrix} \mathbf{q} = 0. \tag{110}$$

Considering all indices, we have

$$\mathtt{P}_{\mathtt{d}ijkl} = \mathtt{V}^\top (\mathbf{e}_j \otimes \mathtt{I}_3) \lfloor \mathbf{e}_l \rfloor_\times (\mathbf{e}_i \otimes \mathtt{I}_3)^\top \mathtt{V} \tag{111}$$
$$- \begin{bmatrix} 0_{3\times 3} & 0_{3\times 1} \\ \mathbf{e}_l^\top (\mathbf{e}_k \otimes \mathtt{I}_3)^\top & 0 \end{bmatrix} \tag{112}$$
$$\mathbf{q}^\top \mathtt{P}_{\mathtt{d}ijkl} \mathbf{q} = 0 \tag{113}$$

with $(i,j,k) = \{(1,2,3),(2,3,1),(3,1,2)\}, l = \{1,2,3\}$, for a total of 9 equations.

Appendix E The Splitting Conic Solver

The Splitting Conic Solver (SCS) [29] makes use of a first-order method for solving large-scale primal-dual convex cone programs. The method relies on the Alternating Direction Method of Multipliers (ADMM) [4] to solve the homogeneous self-dual embedding. The homogeneous self-dual embedding [44] is an equivalent feasibility problem to the original primal-dual cone program, that consists in finding a nonzero point resultant from the the intersection of a cone and an affine set. At each iteration the solver solves a system of linear equations and projects a point onto the cone. The solver returns solutions to the primal and dual problems, that we leverage to certify the optimality of the solutions returned by CvxPnPL. When not available, it alternatively provides a certificate of infeasibility or unboundedness. The method is particularly suited for large-scale problems where interior point methods are too slow.

Table 3: Experiment demonstrating the insensitivity of CvxPnPL to different rank thresholds for a **(top)** PnP, a **(middle)** PnL and a **(bottom)** PnPL scenario with 4+ points or lines. The median angular and translation errors are shown in each table cell on the left and right, respectively, for different numbers of elements in the scenario, employing Gaussian pixel noise with standard deviation σ computed over 10^4 runs. This is a similar experimental setup to Section 7.1, with the threshold of 0.001 corresponding to our baseline.

Perspective-n-Points					
Noise σ	Nr. of Points	thr=0.001 rot/tr (°/%)	thr=1e-06 rot/tr (°/%)	thr=1e-09 rot/tr (°/%)	thr=1e-12 rot/tr (°/%)
0.0	4	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	6	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	8	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	10	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	12	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
2.0	4	1.727 / 1.080	1.751 / 1.086	1.765 / 1.092	1.761 / 1.085
	6	1.099 / 0.638	1.102 / 0.641	1.103 / 0.641	1.100 / 0.639
	8	0.849 / 0.515	0.849 / 0.516	0.849 / 0.516	0.849 / 0.515
	10	0.735 / 0.448	0.735 / 0.448	0.735 / 0.448	0.735 / 0.448
	12	0.653 / 0.398	0.654 / 0.398	0.654 / 0.398	0.652 / 0.398
Perspective-n-Lines					
Noise σ	Nr. of Lines	thr=0.001 rot/tr (°/%)	thr=1e-06 rot/tr (°/%)	thr=1e-09 rot/tr (°/%)	thr=1e-12 rot/tr (°/%)
0.0	4	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	6	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	8	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	10	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	12	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
2.0	4	2.166 / 2.253	2.201 / 2.302	2.210 / 2.288	2.211 / 2.254
	6	1.164 / 1.032	1.177 / 1.038	1.183 / 1.044	1.182 / 1.038
	8	0.877 / 0.769	0.883 / 0.772	0.885 / 0.774	0.885 / 0.772
	10	0.756 / 0.651	0.760 / 0.654	0.760 / 0.655	0.760 / 0.654
	12	0.657 / 0.557	0.658 / 0.558	0.658 / 0.559	0.659 / 0.558
Perspective-n-Point-and-Lines					
Noise σ	Pts. or Lines	thr=0.001 rot/tr (°/%)	thr=1e-06 rot/tr (°/%)	thr=1e-09 rot/tr (°/%)	thr=1e-12 rot/tr (°/%)
0.0	4	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	6	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	8	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	10	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
	12	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000	0.000 / 0.000
2.0	4	2.056 / 1.526	2.079 / 1.527	2.105 / 1.546	2.103 / 1.533
	6	1.122 / 0.798	1.130 / 0.808	1.131 / 0.810	1.130 / 0.806
	8	0.880 / 0.619	0.882 / 0.621	0.882 / 0.623	0.882 / 0.622
	10	0.746 / 0.523	0.747 / 0.524	0.747 / 0.524	0.748 / 0.524
	12	0.658 / 0.463	0.659 / 0.463	0.659 / 0.463	0.658 / 0.463

Appendix F Rank Threshold Selection Criterion and Sensitivity

In this section, we provide some additional results supporting our statement on CvxPnPL not requiring accurate rank estimates, that are dependent on the rank threshold. This threshold is used decide whether a given eigenvalue counts towards the rank of $\mathtt{Z}$. CvxPnPL is able to gracefully handle rank overestimates as is shown

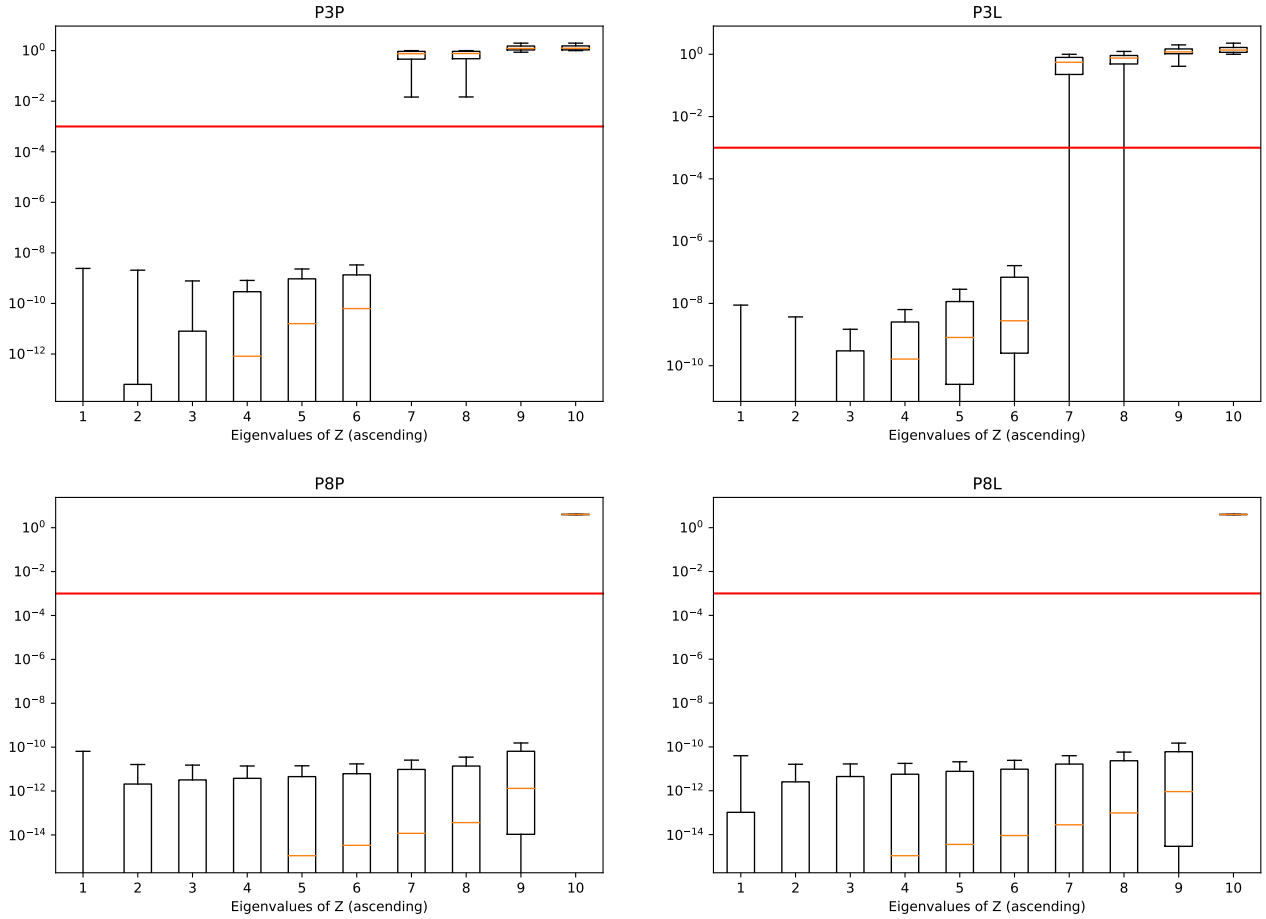

Fig. 9: Distribution of the ten eigenvalues of $\mathtt{Z}$ for minimal (P3L and P3L) and non-minimal (P8P and P8L) scenarios under noiseless conditions, over 1000 runs. The red horizontal line represents the rank threshold used of 0.001. The box upper and lower bounds plus the orange line, mark the three quartiles of the distribution, and the whiskers mark the 5th and 95th percentiles.

in Table 3. However, by setting an extremely low rank threshold, one is throwing away the reduced computational benefit from handling lower rank cases. We picked the rank threshold of 0.001 supported by numerical experiments similar to the setuo in Section 7.1, but using minimal and non-minimal PnP and PnL scenarios. As show in Figure 9, minimal scenarios with 3 points or lines have multiple solutions and produce higher eigenvalues for $\mathtt{Z}$, while for non-minimal scenarios, e.g. 8 points or lines scenarios, there's a single valid solution and lower eigenvalues. CvxPnPL can only handle situations up to rank 4 so the threshold is set to ensure that the 4 largest "non-null" eigenvalues are comfortably above it.