Fast certifiable relative pose estimation with gravity prior

Mercedes Garcia-Salguero[a,*], Javier Gonzalez-Jimenez[a]

[a]*Machine Perception and Intelligent Robotics (MAPIR) Group, System Engineering and Automation Department, University of Malaga, Campus de Teatinos, 29071 Malaga, Spain*

Abstract

Redundant and complementary information from different types of sensors boosts the robustness of autonomous systems, making them more reliable and safer. In particular, inertial measurement units (IMUs) are increasingly being integrated with cameras for that purpose, since the information provided by the IMU helps to simplify some visual problems and improves the accuracy of the results. In the context of estimating the motion of a camera, which is the problem we address in this work, the gravity vector delivers by the IMU reduces the unknown rotation to only one degree of freedom instead of three, hence simplifying the relative pose problem (RPp). Despite this simplification, the RPp is still nonconvex, therefore the quality (optimality) of the solution returned by iterative solvers cannot be guaranteed. These suboptimal solutions may have serious consequences for applications that have this solver as a key block, and may even cause their complete failure.

In this paper, we contribute a certifiable solver for the RPp with gravity prior. We propose an iterative certifier that does not assume any condition on the problem, and returns an optimality certification even for an overconstrained formulation with 28 constraints in less than 1.5 milliseconds. Since the certifier doesn't obtain the solution to the problem, we also provide a fast, iterative on-manifold estimation of the relative pose, which is shown to return solutions with lower costs than other nonminimal solvers in less time. We make the code available at https://www.github.com/mergarsal

Keywords: relative pose problem; optimality certificate; redundant constraints; gravity prior;

1. Introduction

Autonomous vehicles, such as self-driving cars, robots or unmanned aerial systems, are required to perform to a very high level of safety. These vehicles are very complex systems with many critical components that must be designed to avoid or, at least, to auto detect any defect or malfunctioning. A common and necessary component in most of these systems is the computation of the incremental pose of the vehicle

*Corresponding author

Email addresses: mercedesgarsal@uma.es (Mercedes Garcia-Salguero), javiergonzalez@uma.es (Javier Gonzalez-Jimenez)

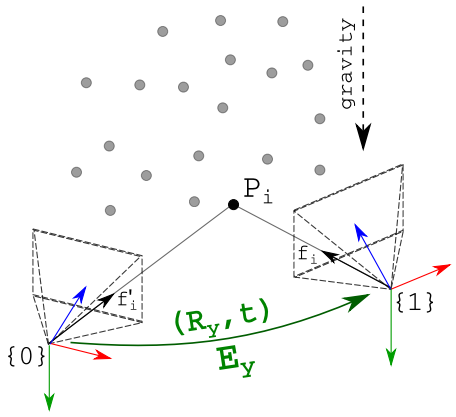


Figure 1: In this work we aim to estimate the relative pose $(\boldsymbol{R}_{\mathrm{Y}}, \boldsymbol{t})$ between cameras $\{0, 1\}$ from N pair-wise observations $\{\boldsymbol{f}_i, \boldsymbol{f}'_i\}_{i=1}^{N}$, being $\boldsymbol{R}_{\mathrm{Y}}$ a rotation around the Y-axis.

from visual data, *i.e*. the relative pose of an onboard camera(RPp). Specifically, it is a key block of the Simultaneous Localization And Mapping (SLAM) pipeline [1, 2, 3]. Cameras are the most common sensor in SLAM, since they are compact, cheap, energy efficient, and suitable for other tasks (e.g. object detection and recognition, scene classification, ...). While many of the existing implementations of SLAM rely only on visual data (observations from the poses), it is a fact that the integration of visual and inertial data from inertial measurement units (IMUs), if available, leads to simpler formulations and more accurate solutions. From the IMU we can obtain, among others, the gravity vector, which gives us the angles of the cameras with respect to this axis, hence reducing the three degrees of freedom of the rotation from three to one, see figure 1.

The gold-standard approach to RPp is Bundle Adjustment (BA), where the pose and the 3D coordinates of the points are jointly estimated. Since BA is iterative, a good initialization is required to converge to the global optimum. The common approach to obtain this initial guess is to leverage a simplification of the problem where the 3D coordinates of the points are not computed [4, 5]. Even if the axis of rotation is known from the IMU, this problem turns to be also nonconvex, that is, the optimality of the solution cannot be guaranteed. This is not a minor issue since such solution will be further used by other higher level components of the autonomous system to determine the global position of the vehicle or the pose of an obstacle in its path. Thus, being able to certify the quality of the solution at an early stage is key for a safe operation of the system.

Whereas for the general case (that using only the images) several nonminimal optimal solvers are available [6, 7, 8], for the particular case where the axis of rotation is known, only one solver is reported in the literature [9]. The latter is a polynomial solver whereas the generic ones rely on Interior-Point Methods (IPM), both being slow in practice. Faster certifiable algorithms can be devised as it was done in, *e.g.* [10, 11, 12] (see section 2). The driven idea behind these approaches is to estimate the

solution by any efficient iterative method and *then* to certify the solution. Most of these approaches develops a closed-form expression for the first stage of the certifier, which puts a limit on the number of variable and constraints, precluding the use of over-constrained (tighter) formulations, and assumes the Linear Independence Constraint Qualification (LICQ), which leads to the full rank of the Jacobian of the constraints. As with the general case, the LICQ doesn't hold for the RPp with known gravity vector even for the smallest formulations.

Contribution: In this work we aim to solve efficiently the RPp with known gravity vector with an optimality certificate. We state the problems in terms of the essential matrix $\boldsymbol{E}$ and minimize the squared normalized epipolar error $\epsilon_i^2 = (\boldsymbol{f}_i^T \boldsymbol{E} \boldsymbol{f}_i')^2$.

To make the solver efficient, we opt for estimating the essential matrix by an iterative algorithm that enforces the known condition about the rotation. Since the algorithm is iterative, we cannot guarantee that the returned solution is the global optimum. Thus, we also contribute an optimality certifier based on duality theory that does not require LICQ nor limit the number of variables and constraints of the problem. Since the certifier depends on the definition of the space, we provide four different formulations for the set of essential matrices with known rotation axis, with increasing number of constraints, following the work in [13], which leads to two overconstrained formulations.

An exhaustive evaluation with both synthetic and real data is performed, where we show that the on-manifold estimation is faster and more stable than other nonminimal solvers and attains solutions with lower cost, even when the known axis of rotation is noisy. Further, for small formulations our certifier is able to return an optimality certificate in less than one millisecond, and for an overconstrained formulation with 28 constraints, it returns the certificate in 1.5 milliseconds.

We make the code publicly available at `https://github.com/mergarsal`.

Notation: Bold, upper-case letters denote matrices *e.g.* $\boldsymbol{E}, \boldsymbol{H}$, whereas bold, lower-case letters denote (column) vectors *e.g.*, $\boldsymbol{t}, \boldsymbol{f}_i$ and normal font letters *e.g.*, a, b denote real scalar. $\mathbb{R}^{n\times m}$ is the set of $n\times m$ real-valued matrices, $\mathbb{S}^n \subset \mathbb{R}^{n\times n}$ the set of symmetric matrices of dimension $n\times n$ and $\mathbb{S}^n_+$ the cone of positive semidefinite (PSD) matrices of dimension $n \times n$. A PSD matrix will be also denoted by $\succeq$, *i.e.*, $\boldsymbol{S} \succeq 0 \Leftrightarrow \boldsymbol{S} \in \mathbb{S}^n_+$. The 3×3 skew-symmetric matrix $[\boldsymbol{t}]_\times$ is the equivalent matrix form for the cross-product with a 3D vector $\boldsymbol{t} = [t_1, t_2, t_3]^T$, *i.e.*, $\boldsymbol{t} \times (\bullet) = [\boldsymbol{t}]_\times(\bullet)$ with

$$[\boldsymbol{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} \tag{1}$$

The operator vec($\boldsymbol{E}$) vectorises the given matrix $\boldsymbol{E} \in \mathbb{R}^{m\times n}$ column-wise. The kronecker product is denoted as $\otimes$. We will denote the trace of a matrix as $\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^n a_{ii}$, $\boldsymbol{A} = [a_{ij}] \in \mathbb{R}^{n\times n}$. We identify 3D rotations with points in the rotation group $\mathrm{SO(3)} \doteq \{\boldsymbol{R} \in \mathbb{R}^{3\times 3} | \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}_3,\ \det(\boldsymbol{R}) = 1\ \}$ and define the 2-sphere as $\mathcal{S}^2 \doteq \{\boldsymbol{t} \in \mathbb{R}^3 | \boldsymbol{t}^T\boldsymbol{t} = 1\}$.

2. Related work

We summarize here previous works regarding the relative pose estimation with known axis of rotation and those where optimality certifiers were proposed.

2.1. Minimal solver

Given the observations of five generic points we can compute the relative pose between the calibrated cameras that observed them, see *e.g*. [14, 15, 16]. Nevertheless, when the axis of rotation is known the number of correspondences required for estimating the pose is reduced to three. Thus general solvers are not guaranteed to return a solution with the desired rotation when the observations are corrupted by noise. Introducing this information simplifies the problem and guarantees that the rotation has the desired form. Fraundorfer *et al*. [17] propose a minimal solver for this problem that employs only three pair-wise correspondences. Ding *et al*. in [18] eliminate the rotation and translation parameters from the original constraints, obtaining a reduced set of quadratic and cubic constraints. Sweeney *et al*. in [19] propose a minimal solver by re-writing the problem as a generalized eigenvalue problem, avoiding the resolution of the polynomial system.

However, and although these solvers can be integrated into robust paradigms such as RANSAC to detect and discard outliers, they come with well-known drawbacks. First, they usually require to compute a Gröebner basis and the roots of a high-order polynomial, which may become unstable and slow. Second, since they only employ the minimum number of correspondences they are more sensitive to noise.

2.2. Nonminimal solver

In real-world applications many correspondences are usually found and thus nonminimal solvers that incorporate all the observations are leveraged instead. A simple approach applies a linearization of the problem, known as Direct Linear Transform (DLT), which is faster and more stable than minimal solvers. However, the solution comes without guarantees to be a valid real pose. A common method that overcomes this is to state the problem as a nonconvex optimization on the desired domain. In general, this approach may encounter many local minima and therefore the optimality of the solution is not guaranteed, though in practice, they have shown to achieve good performance, see *e.g*. [20, 21, 22, 23, 24]. Interestingly, Ding *et al*. in [9] propose for the relative pose problem with known rotation axis an alternative approach which decouples rotation and translation following the work in [6]. A Gröbner basis is then derived to solve this nonminimal solver, guaranteeing that the solution is the global optimum. The proposal though becomes unstable for some problem instances, as we observe in the experiments of Section 5.

2.3. Optimality certificate

Given the nonconvexity of the problem, solvers that are able to certify the optimality of the solution are highly desired. Branch-and-bound techniques have been proposed for the general relative pose problem, see *e.g*. [6, 25]. Due to their exploratory nature, they may have exponential complexity in the worst case, which prevents their application in practice. An alternative technique leverages a convex relaxation of the problem on the semidefinite positive cone, which can be solved in polynomial time by off-the-shelf solvers such as SeDuMi [26] or SDPT3 [27], and under some circumstances they may return the optimal solution to the original, nonconvex problem. For

the general formulation (only with visual data) this was the approach followed in [7, 8]. The main drawback of these works is that the adopted relaxations may not remain tight for all problem instances, meaning that the returned solution is not feasible for the original problem. Empirically, it has been shown that increasing the number of constraints tightens the convex relaxations, although it also increases the computational time required by the solvers. Previous works [28, 29, 30] have focused on developing fast and stable solvers for problems with large number of constraints and variables, although they are restricted to some domains (for example, rotations) or assume that the problems met other conditions that hold generally, for example, the Linear Independence Constraint Qualification (LICQ), strict complementarity (primal and dual solution have no common nullspace) and maximum complementarity (optimal primal and dual solutions has maximal rank among all optimal solutions). We later show that all these conditions don't necessarily hold for the relative pose problem as formulated in this work, which hinders the application of the above-mentioned solvers to our problem and motivates our work.

Certifiable algorithm

An alternative to the above-referred global optimization methods consists of finding a solution by any classic iterative method and then certifying whether such solution is the global optimum [10, 12, 31, 11, 32]. Given the efficiency of iterative solvers, these optimality certifiers acquire special relevance in real-world applications. Most of these proposals compute a part of the certifier (the Lagrange multipliers) in closed-form and check whether the associated Hessian is positive semidefinite (PSD). This last step is the most time consuming part, specially if the Hessian matrix is dense and/or has large size. Additionally, in order to derive a closed-form for the multipliers, the problem must also fulfill some conditions. First, the number of constraints must be non-greater than the number of variables, which limits the number of constraints and precludes in most cases the introduction of redundant constraints. Second, the coefficient matrix associated with the lagrange multipliers (the Jacobian of the constraints evaluated at the potential optimal solution) must be full rank (LICQ). In our previous work [32] where the general RPp was addressed, we showed that LICQ does not hold. To attain LICQ, we drop one of the constraints of the feasible set, relaxing the original problem. This is possible because only one constraint must be removed, given a low number of combinations (six) which do not deteriorate the performance of the certifier. However, if more constraints have to be removed (as for the formulations employed in this work), the number of combinations explodes, making the approach impractical.

3. Formulation for the relative pose problem

The RPp consists of estimating the rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ between two cameras given a set of N pair-wise observations $(\boldsymbol{f}_i, \boldsymbol{f}'_i), i = 1, \ldots, N$ originated by N 3D points. When these points are in general position, the relative pose can be estimated by computing the essential matrix $\boldsymbol{E} \in \mathbb{R}^{3\times3}$ [33]. In this work we assume the gravity vector is known (for example, provided by an IMU), which allows us to simplify the relative rotation between the two views and reduces the degrees of freedom from three to one. The simplified rotation has as axis of rotation the gravity one (the Y-axis, see

figure 3a) and thus it has the form:

$$\boldsymbol{R}_{\mathrm{Y}} = \begin{pmatrix} c & 0 & s \\ 0 & 1 & 0 \\ -s & 0 & c \end{pmatrix}, \tag{2}$$

where $c, s \in \mathbb{R}$ such that $c^2 + s^2 = 1$. For simplicity, we call this kind of rotation *Y-rotation*, and denote the set of these rotations by $\mathcal{R}^{\mathrm{Y}}$ with

$$\mathcal{R}^{\mathrm{Y}} \doteq \{\boldsymbol{R} \in \mathbb{R}^{3\times 3} \mid \boldsymbol{R} = \boldsymbol{R}_{\mathrm{Y}}, c^2 + s^2 = 1, c, s \in \mathbb{R}\}, \tag{3}$$

Then, the essential matrix $\boldsymbol{E} = [\boldsymbol{t}]_{\mathrm{x}}\boldsymbol{R}$ for a Y-rotation has the explicit pattern

$$\boldsymbol{E}_{\mathrm{Y}} = \begin{pmatrix} e_1 & e_2 & e_3 \\ e_4 & 0 & e_5 \\ -e_3 & e_6 & e_1 \end{pmatrix}, \tag{4}$$

with only six unknowns $e_1, \ldots, e_6$ and 3 degrees of freedom. Section A in the *Supplementary material* shows that any essential matrix with the form in eq. (4) has an associated Y-rotation. We define the set of *real* essential matrices with the form in eq. (4) by $\mathcal{E}^{\mathrm{Y}}$,

$$\mathcal{E}^{\mathrm{Y}} \doteq \{\boldsymbol{E} \in \mathbb{R}^{3\times 3} \mid \boldsymbol{E} = [\boldsymbol{t}]_{\mathrm{x}}\boldsymbol{R}, \boldsymbol{t} \in \mathcal{S}^2, \boldsymbol{R} \in \mathcal{R}^{\mathrm{Y}}\}. \tag{5}$$

The RPp is stated as an optimization problem that minimizes the sum of normalized squared epipolar errors $\epsilon_i^2 = (\boldsymbol{f}_i^T \boldsymbol{E} \boldsymbol{f}_i')^2$ over $\mathcal{E}^{\mathrm{Y}}$,

$$f_{\mathrm{E}}^{\star} = \min_{\boldsymbol{E}\in\mathcal{E}^{\mathrm{Y}}} \mathrm{vec}(\boldsymbol{E})^T \boldsymbol{C}_{\mathrm{E}} \mathrm{vec}(\boldsymbol{E}) \tag{O}$$

with $\boldsymbol{C}_{\mathrm{E}} = \sum_{i=1}^{N} (\boldsymbol{f}_i \otimes \boldsymbol{f}_i')(\boldsymbol{f}_i \otimes \boldsymbol{f}_i')^T \in \mathbb{S}_+^9$.

Our goal is to solve problem O and obtain an optimality certificate that guarantees that the estimated solution is the global optimum. For that, we leverage concepts from duality theory, specifically the dual problem, see [34]. To derive this auxiliary problem we need to reformulate problem O, *i.e.* the cost function and the set of constraints, as quadratic functions on the variables, hence obtaining a Quadratically Constrained Quadratic Problem (QCQP). This reformulation doesn't simplify the problem, but allows us to derive the dual problem in a simpler way. Since the cost function is already quadratic in the entries of the variable $\boldsymbol{E}$ (the unknown), we will focus now on the definition of the set $\mathcal{E}^{\mathrm{Y}}$.

3.1. Definitions of the essential matrix set

While other characterizations of this set exist, in this work we leverage those collected in [13]. Adapting these sets to essential matrices with the form in eq. (4) is straightforward by considering the pattern of $\boldsymbol{E}$, that is, considering the null and repeated entries.

The different parameterizations are related to two important elements associated with the relative pose: the translation vectors $\boldsymbol{t}, \boldsymbol{q} \in \mathbb{R}^3$ (the epipoles). The vector $\boldsymbol{t}$ is the left nullspace of the associated $\boldsymbol{E}$ and the vector $\boldsymbol{q} = \boldsymbol{R}^T\boldsymbol{t}$ is the right nullspace considering that $\boldsymbol{E} = \boldsymbol{R}[\boldsymbol{q}]_{\mathrm{x}} = [\boldsymbol{t}]_{\mathrm{x}}\boldsymbol{R}$.

Right formulation: We exploit the unitary condition of the rotation matrix $\boldsymbol{R}$ to obtain the characterization:

$$\mathcal{E} \doteq \{\boldsymbol{E} \in \mathbb{R}^{3\times3} | \boldsymbol{E}^T\boldsymbol{E} = [\boldsymbol{q}]_\times[\boldsymbol{q}]_\times^T, \boldsymbol{q} \in \mathcal{S}^2\}, \tag{6}$$

where $\boldsymbol{q}$ is the right nullspace of $\boldsymbol{E}$. For our problem, this set, denoted by RIGHT, provides seven constraints and requires nine variables.

Left formulation: This description also exploits the unitary condition of the rotation matrix $\boldsymbol{R}$:

$$\mathcal{E} \doteq \{\boldsymbol{E} \in \mathbb{R}^{3\times3} | \boldsymbol{E}\boldsymbol{E}^T = [\boldsymbol{t}]_\times[\boldsymbol{t}]_\times^T, \boldsymbol{t} \in \mathcal{S}^2\} \tag{7}$$

where $\boldsymbol{t}$ is the left nullspace of $\boldsymbol{E}$. This set, denoted by LEFT, provides seven constraints and requires nine variables. We combine LEFT and RIGHT, discard linear dependent constraints and obtain a third set, denoted by BOTH, with 13 (redundant) constraints and 12 variables.

Adjugate formulation: The following set of constraints forces the singular values of a 3×3 real matrix $\boldsymbol{E}$ to be $\sigma_1 = \sigma_2 = 1$ and $\sigma_3 = 0$:

$$\boldsymbol{q}^T\boldsymbol{q} = 1 \qquad \boldsymbol{t}^T\boldsymbol{t} = 1 \tag{8}$$

$$\boldsymbol{E}\boldsymbol{q} = \boldsymbol{0}_{3\times1} \qquad \boldsymbol{E}^T\boldsymbol{t} = \boldsymbol{0}_{3\times1} \tag{9}$$

$$\mathrm{tr}(\boldsymbol{E}\boldsymbol{E}^T) = 2 \qquad \mathrm{Adj}\,(\boldsymbol{E}) = \boldsymbol{q}\boldsymbol{t}^T, \tag{10}$$

where $\boldsymbol{t}, \boldsymbol{q}$ are the left and right nullspaces of $\boldsymbol{E}$, respectively and $\mathrm{Adj}\,(\boldsymbol{E})$ is the adjugate matrix of $\boldsymbol{E}$. We join all these definitions to obtain the last set ADJ with 28 constraints and 12 variables.

3.2. Lagrangian dual problem

These sets are all equivalent yet provide different convex relaxations and performance in terms of certification. To derive these relaxations from the original problem QCQP, we first re-write it in the generic form:

$$f^\star = \min_{\boldsymbol{X}\in\mathcal{X}} \mathrm{vec}(\boldsymbol{X})^T\boldsymbol{C}\mathrm{vec}(\boldsymbol{X}) \tag{QCQP}$$

where $\mathcal{X}$ is a generic feasible set defined by m quadratic constraints on the entries of the matrix $\boldsymbol{X}$ (the unknown), equivalently $\boldsymbol{x} = \mathrm{vec}(\boldsymbol{X}) \in \mathbb{R}^n$. The constraints are defined as $\mathrm{vec}(\boldsymbol{X})^T\boldsymbol{A}_i\mathrm{vec}(\boldsymbol{X}) = \boldsymbol{x}^T\boldsymbol{A}_i\boldsymbol{x} = c_i \in \mathbb{R}$, with $\boldsymbol{A}_i \in \mathbb{S}^n$ for $i = 1, \ldots, m$.

Two well-known convex relaxations have been leveraged in the literature for problems with the form in QCQP: the Shor's relaxation and the dual problem. The first one relaxes the problem as a semidefinite problem (SDP) that can be solved by off-the-shelf solvers in polynomial time in the number of variables and constraints. Under a pure c++ implementation previous works [8, 13] have reported low computational times circa $6 - 12$ milliseconds per problem instance.

The dual problem for problems like QCQP is also a SDP, but allows us to derive an optimality certifier under some conditions. This certifier tends to be faster than the above-mentioned approaches and achieves good performance in terms of certification,

that is, it can certify most of the optimal solutions. To derive it, we first define the (generic) dual problem for the (generic) primal QCQP

$$d^\star = \max_{\boldsymbol{\lambda}} \boldsymbol{\lambda}^T \boldsymbol{c}, \text{ subject to } \boldsymbol{S} = \boldsymbol{C} - \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i \geq 0. \tag{D}$$

Here, $\boldsymbol{c} = [c_1, \ldots, c_m]^T \in \mathbb{R}^m$ and $\boldsymbol{S} \in \mathbb{S}^n$ is the Hessian of the Lagrangian. The feasible set of the dual D are all the vectors $\boldsymbol{\lambda} \in \mathbb{R}^m$, called Lagrange multipliers, that make their associated Hessian $\boldsymbol{S}$ positive semidefinite (PSD).

From duality theory [34] we know that the optimal cost of the dual problem represents a lower bound on the cost of the primal QCQP for any pair of primal ($\boldsymbol{x}$)-dual($\boldsymbol{\lambda}$) feasible points for QCQP-D, formally: $d(\boldsymbol{\lambda}) \leq d^\star \leq f^\star \leq f(\boldsymbol{x})$.

4. Optimality certification

For some problem instances the inequality $d^\star \leq f^\star$ is tight, meaning that the optimal costs of both problems are the same, and we say that *strong duality* holds. While for the problems here considered we cannot guarantee a priori that strong duality holds, we empirically observe it in section 5. This relation is the driven idea behind the optimality certificate: Given a potentially optimal solution $\boldsymbol{x}^\star$ for the primal O, we try to find a feasible dual point $\boldsymbol{\lambda}^\star$ for its dual with same cost $f^\star = d^\star$. If such solution exists, then the provided primal solution is the global optimum. Notice that we have assumed that a primal solution for the problem is provided. How this solution has been estimated is not relevant as long as it is feasible for the primal problem. We propose in this work to leverage the machinery of Riemmanian optimization to estimate both the primal and dual solutions. To estimate the primal solution, we follow the approach in [32] and we include at the end of this Section only the main aspects. We refer the reader to this reference and the *Supplementary material* section E for further details. The remaining of this Section is devoted to the estimation of the optimality certificate in its general form.

The condition that both costs ($d^\star = f^\star$) are the same can be re-written in terms of the variables of the problem as $f^\star - d^\star = \boldsymbol{x}^{\star T}\boldsymbol{C}\boldsymbol{x}^\star - \boldsymbol{\lambda}^{\star T}\boldsymbol{c} = \boldsymbol{x}^{\star T}\boldsymbol{C}\boldsymbol{x}^\star - \sum_{i=1}^m \boldsymbol{\lambda}^\star{}_i \boldsymbol{x}^{\star T}\boldsymbol{A}_i\boldsymbol{x}^\star = \boldsymbol{x}^{\star T}(\boldsymbol{C} - \sum_{i=1}^m \boldsymbol{\lambda}^\star{}_i \boldsymbol{A}_i)\boldsymbol{x}^\star = \boldsymbol{x}^{\star T}\boldsymbol{S}^\star\boldsymbol{x}^\star = 0$, since the optimum $\boldsymbol{x}^\star$ is feasible for prob. QCQP and thus, $\boldsymbol{x}^{\star T}\boldsymbol{A}_i\boldsymbol{x}^\star = c_i$ for all $i = 1, \ldots, m$. Since the dual point $\boldsymbol{\lambda}^\star$ is feasible, we know that its associated Hessian $\boldsymbol{S}^\star$ is PSD. Thus, we have that $\boldsymbol{x}^{\star T}\boldsymbol{S}^\star\boldsymbol{x}^\star = 0 \Leftrightarrow \boldsymbol{S}^\star\boldsymbol{x}^\star = (\boldsymbol{C} - \sum_{i=1}^m \boldsymbol{\lambda}^\star{}_i \boldsymbol{A}_i)\boldsymbol{x}^\star = \boldsymbol{0}_{n\times 1}$ for the optimal primal-dual $(\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star)$ points. This last relation allows us to re-formulate the condition that the dual gap is zero by the equivalent condition that the primal solution $\boldsymbol{x}^\star$ must lay on the nullspace of the Hessian $\boldsymbol{S}^\star$, *i.e.* $\boldsymbol{S}^\star\boldsymbol{x}^\star = \boldsymbol{0}_{n\times 1}$.

We want to point out that we are not solving the relaxation from scratch, but as other proposals have done previously (see section 2), we assume that strong duality holds and try to obtain a certificate of it. As we indicate before, if the number of constraints is not greater than the number of variables and if the LICQ holds, then a closed-form for the Lagrange multipliers can be derived. While these requirements have been shown to be met for other problems, they don't hold for *any* of the formulations considered in

this work. For the general RPp in [32] we proposed to remove one of the constraints to attain LICQ, hence relaxing the original problem. We avoid that approach here since it doesn't perform well when applied to the redundant formulations (the relaxations become too loose to be used in practice). Instead we propose an alternative approach that also allows to certify solutions but in an iterative manner, admitting more constraint than variables and problems without LICQ. It also handles problems where the set of dual optimal solutions is not a singleton and where strict and maximal complementarity between optimal solutions don't hold.

The certification problem can be reformulated as a feasibility problem for a primal solution $\boldsymbol{x}$ as

$$\text{find } \boldsymbol{\lambda} \text{ such that } \boldsymbol{S} = \boldsymbol{C} - \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i \succeq 0 \text{ and } \boldsymbol{S}\boldsymbol{x} = \boldsymbol{0}_{n\times 1}. \tag{F}$$

Inspired by previous works [35, 29], we opt here for minimizing the constraints $\boldsymbol{S} = \boldsymbol{C} - \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i$ and $\boldsymbol{S}\boldsymbol{x} = \boldsymbol{0}_{n\times 1}$, thus obtaining a relaxation of the form

$$\begin{aligned} f_{\mathrm{F}}^{\star} = \min_{\boldsymbol{S}\in\mathbb{R}^{n\times n},\boldsymbol{\lambda}\in\mathbb{R}^{m}} & \|\boldsymbol{S}\boldsymbol{x}\|_2^2 + \left\|\boldsymbol{S} - \boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i\right\|_{\mathrm{F}}^2 \\ \text{subject to } & \boldsymbol{S} \succeq 0. \end{aligned} \tag{R-F}$$

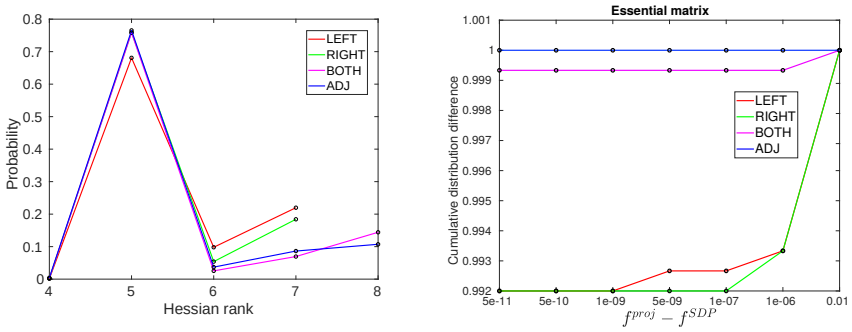
Notice that if a solution exists for problem F, then the optimal cost of R-F is zero, and it is necessarily $f_{\mathrm{F}}^{\star} > 0$ otherwise.

Since the problem R-F is convex and always feasible (see *Supplementary material* Section B), we can rely on the Burer-Monteiro low-rank approach [36] to solve it. For that, we decompose the matrix $\boldsymbol{S} \in \mathbb{S}_{+}^{n}$ as the outer product of $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$, *i.e.* $\boldsymbol{S} = \boldsymbol{Y}\boldsymbol{Y}^T$, and the rank of $\boldsymbol{S}$ is bounded above by $k < n$. With this parameterization, the resulting matrix $\boldsymbol{Y}\boldsymbol{Y}^T$ is already symmetric and PSD. Problem R-F is then re-written in terms of $\boldsymbol{Y}$ as

$$f_{\mathrm{F}}^{\star} = \min_{\boldsymbol{Y}\in\mathbb{R}^{n\times k},\boldsymbol{\lambda}\in\mathbb{R}^{m}} \left\|\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\right\|_{\mathrm{F}}^2 + \left\|\boldsymbol{Y}\boldsymbol{Y}^T - \boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i\right\|_{\mathrm{F}}^2 \tag{R-Y}$$

Before continuing, we must say that problem R-Y is non-convex, and thus, we cannot guarantee that the iterative algorithm returns the global optimum. However, as indicated above, we seek the zero-cost solution and thus, any solution with a different cost cannot certify optimality, either because it's suboptimal (a local minimum of the problem) or because any of the assumptions made are incorrect. Note that we cannot apply the same logic to the RPp directly, since the cost of the global optimum is not known a priori (in the presence of noise).

Further, notice that we don't assume any value for the rank of the matrix $\boldsymbol{S}$, and thus the dimension of $\boldsymbol{Y}$. This rank is associated with the tightness of the relaxation, the concept of *strict complementary* and *maximal complementarity*, see *e.g.* [37]. For most problems with strong duality we have strict complementary, that is, the rank of $\boldsymbol{S}$ is $n - 1$. Further, off-the-shelf solvers, such as SeDuMi or SDPT3, are central path following algorithms that return *maximally complementarity solutions* [38, Sec. 4.1.2],



(a) (b)

Figure 2: Fig. (a) Rank of the Hessian during the evaluation of the certifier in Section 5 for the different formulations (see legend) for the essential matrix. The rank of the cost matrix $\boldsymbol{C}$ is, in all cases, at least five. Fig. (b) Cumulative distribution of the difference between the cost attained by our proposal and the cost attained by solving the feasibility problem R-Y with off-the-shelf tools.

i.e. primal and dual solutions with maximal rank among all optimal solutions, as De Klerk *et al*. in [39] proved in Theorem 2.1. This behavior makes these two conditions difficult to detect and/or identify in practice.

For the RPp as formulated in this work these assumptions don't hold. First, due to the pattern of the primal problem QCQP, there exist optimal solutions with rank larger than one. Provided strong duality holds, this necessarily implies that there exists at least one Hessian with rank less than $n-1$ but also with rank $n-1$ (different PSD Hessian matrices), and therefore, there exists also more than one dual optimal solution. Since we can have a rank-1 primal solution and a dual solution with rank less than $n-1$, the strict *and* maximal complementarity conditions fail. Notice that all these conditions tend to hold for most problem instances, and together with LICQ, are assumed in most solvers. However, and since our interest is focused on optimality certification, *any* of these dual solutions is valid for certification and we don't need to assume any of these conditions. In fact, we observe that our proposal, on the other hand, empirically tends to the dual solution $\boldsymbol{S}$ with low rank. Notice that this is a desired feature, since the complexity of problem R-Y depends on the rank k of $\boldsymbol{Y}$, as we will elaborate later. This also implies that the optimality certificate returned by our algorithm will not fulfill the strict complementarity condition: nevertheless, since the solution is optimal for the dual problem, it is equally valid and sufficient for certification.

To the best of our knowledge, the minimum rank of the Hessian for a given problem instance is not known a priori. However for the different formulations of the RPp as considered in this work these ranks are stable as we show in figure 2a [1], and tend to five for *all* the formulations, even though the size of the (symmetric) Hessian matrix is nine for LEFT and RIGHT, and twelve for BOTH and ADJ. In order to provide users with a generic tool, we believe an automatic detection of this rank will be useful for the

[1] These values are extracted from our evaluation in section 5

potential users and/or for other problems. The idea behind this automatic detection is similar to the Riemmanian staircase proposed in [36]. We start by solving the problem R-Y with rank $1 \leq k < n$. If during the optimization we detect a rank deficient solution $\boldsymbol{Y}$, we reduce the rank and re-run the optimization using the previous solution as initial guess. This reduction is required to maintain the completeness of the space, see *e.g.* [40], as we will explain later. If the optimization with rank k ends and the cost is not zero, we try to increase the rank by one and re-run the optimization. To estimate this $(k+1)$-rank matrix we deploy the line search algorithm in [36] applied to the previous Hessian of the Lagrangian and, in case of failure, the initialization of the algorithm. Alg 2 includes the pseudo-code of this logic. We provide further details in *Supplementary material* section C. The main drawback of this staircase-like certifier is that it implies more than one iteration of the optimization in problem R-Y with different rank(s). Empirically, though, we observe that for the Relative pose problem the number of iterations is low.

Initialization for the iterative certifier: The initial rank is thus really important since an accurate estimation means fewer iterations of the certifier. We propose to relax the domain of $\boldsymbol{S}$ in problem R-F as $\mathbb{S}^n_+ \subset \mathbb{S}^n$. Among all the solutions to this relaxation, we are interested in the one with minimum norm $\|\boldsymbol{\lambda}\|_2$. For this particular solution, the Hessian $\boldsymbol{S}$ can be seen as a perturbation of the data matrix $\boldsymbol{C}$. Then, we seek the minimum norm solution to the linear system (potentially under-determined) $\boldsymbol{J}(\boldsymbol{x})\boldsymbol{\lambda} = \boldsymbol{b}(\boldsymbol{x})$, where $\boldsymbol{J}(\boldsymbol{x}) = [\boldsymbol{A}_1\boldsymbol{x}, \dots, \boldsymbol{A}_m\boldsymbol{x}] \in \mathbb{R}^{n\times m}$ is the Jacobian of the constraints and $\boldsymbol{b}(\boldsymbol{x}) = \boldsymbol{C}\boldsymbol{x} \in \mathbb{R}^{n\times 1}$, all of them depending on the data matrix, the constraints, and the solution. The minimum norm solution is found via the pseudo-inverse of the Jacobian $\boldsymbol{J}(\boldsymbol{x})$. Since LICQ doesn't hold for the RPp as formulated in this manuscript, this Jacobian is rank-deficient for any feasible solution for problem QCQP for *all* the considered formulations. This deficiency has to be taken into account while computing the pseudo-inverse, and in practice, we apply a threshold $1e-09$ to the singular values to avoid numerical instabilities. A feasible solution for R-F can be obtained by projecting this initial guess $\hat{\boldsymbol{S}}$ through an eigenvalue decomposition. If $\hat{\boldsymbol{S}} \in \mathbb{S}^n$ is the Hessian evaluated at $\boldsymbol{\lambda}$ with minimum 2-norm, not necessarily PSD, and $\hat{\boldsymbol{S}} = \boldsymbol{U}\,\mathrm{diag}(\mu_1, \dots, \mu_n)\boldsymbol{U}^T$ is its spectral decomposition, we can obtain the closest PSD matrix in the Frobenius sense as $\boldsymbol{S} = \boldsymbol{U}\,\mathrm{diag}((\mu_1)_+, \dots, (\mu_m)_+)\boldsymbol{U}^T \in \mathbb{S}^n_+$, with the operator $(a)_+ = \max(0, a), a \in \mathbb{R}$. Notice that this initialization can be also employed to obtain solutions with rank $k+1$ when the line search algorithm is not able to estimate it.

Solving problem (R-Y): We solve problem R-Y by leveraging the Riemmanian machinery and separating the domain, the solver and the problem. As solver we employ the trust-region method as it was done in previous works, see *e.g.* [41, 32]. For the domain, it is necessary to recall that we seek the m-D vector $\boldsymbol{\lambda}$ and the $n \times k$ matrix $\boldsymbol{Y}$, knowing that $\boldsymbol{S} = \boldsymbol{Y}\boldsymbol{Y}^T$. Under this relation, we can notice that there exists an equivalence between points $\boldsymbol{Y}, \boldsymbol{W} \in \mathbb{R}^{n\times k}$ as $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{O}$ with $\boldsymbol{O} \in \mathrm{O}(k)$ and so $\boldsymbol{S} = \boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{W}\boldsymbol{W}^T$. To respect this aspect, we leverage the geometry defined in [42, 43] which takes into account the equivalence. The operators associated with it are found in *Supplementary material* section D. Under this definition, the space is not complete and the geometry breaks down when $\boldsymbol{Y}$ is rank deficient [40, 42], which again motivates our interest in finding the correct rank for the solution. Last, the quadratic model (cost,

gradient and vector-product Hessian) for the generic certifier as in R-Y can be written in terms of the variables. We provide them in the *Supplementary material* section D. The initial guess for $\boldsymbol{Y}$ is given by $\boldsymbol{Y} = \boldsymbol{U}\,\mathrm{diag}((\mu_1)_+, \ldots, (\mu_k)_+)$ W.L.O.G. since the employed characterization of the set contemplates the equivalence.

The most consuming operation associated with the chosen geometry is the projection onto the tangent space of an ambient point [44]. This involves the solution of the Lyapunov equation of size k, being k the rank of $\boldsymbol{Y}$. This makes our algorithm specially suitable for problems with low rank Hessian $\boldsymbol{S}$, *i.e.* when strict complementarity does not hold even for tight relaxations. For example, the Hessian for our formulation ADJ tends to the rank of the minimal LEFT and RIGHT, making the certifier only slightly slower, see Table 3b. For problems where the rank of the Hessian is larger, this operation may slow the certifier. A naive approach to avoid this operation relies on dropping (overlooking) the relation between equivalent points in the manifold and optimizing over the $n \times k$ Euclidean space for the variable $\boldsymbol{Y}$. The main advantage of this approach is that there is no need of detecting the deficiency of the solution nor estimating a solution with larger/smaller rank during the optimization, thus avoiding increasing/decreasing the rank. That is, we can project the solution from the initialization to obtain a PSD matrix, and feed the naive problem in R-Y with the $n \times (n-1)$ matrix. We perform only one iteration but the final solution may not have rank $n-1$. The main drawback regards the convergence since equivalent points from the point of view of $\boldsymbol{S}$ are not considered as such with this simplification. This may cause the algorithm to "jump" from one equivalent point to another. In spite of this, in our experiments the time required by both approaches were similar and provide similar results regarding the final cost. Our implementation allows the user to choose the domain of $\boldsymbol{Y}$.

Implementation details: The full algorithm of the proposed iterative certifier is summarized in Alg. 1. We discuss now some aspects regarding the implementation. First, to avoid triggering the rank deficient condition at the first iteration, we compute the initial guess for problem R-Y by applying a tolerance $\tau_{\text{eigen-init}} = 1e-04$ to the eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \mu_m$ divided by the maximum eigenvalue μ_n, *i.e.* we zeroed all the ratii μ_i/μ_m under $\tau_{\text{eigen-init}}$. This gives us the initial rank of the potential solution. This initial threshold $\tau_{\text{eigen-init}}$ is larger than the one that detects rank deficient points $\tau_{\text{eigen}} = 1e-06$ during the optimization. Second, we stop the optimization of the second problem if the cost is below a given threshold τ_{cost}. Since the cost is the sum of the squared entries of $\boldsymbol{Sx}$ and $\boldsymbol{S} - (\boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i)$, we normalize it by the size n of the vector $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$

$$f^{\star}_{\text{F-N}} = \frac{1}{n^2}\|\boldsymbol{Sx}\|_2^2 + \frac{1}{n^2}\left\|\boldsymbol{S} - \boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i\right\|_{\text{F}}^2. \tag{11}$$

Thus, we stop the optimization if the cost $f^{\star}_{\text{F}}$ without normalization reaches the value $\tau_{\text{cost}} = 1e-09$ or the normalized cost $f^{\star}_{\text{F-N}}$ goes under $\tau_{\text{cost-norm}} = 1e-11$. Empirically we observe that non-certified solutions have associated costs close to $1e-06$. The algorithm stops when the rank of the matrix $\boldsymbol{Y}$ is larger than $n-1$. Further, if the line search algorithm is unable to find a new solution, we leverage the initial guess and continue the process. Alg 2 shows the process to increase the rank of the solution $\boldsymbol{Y}$.

Algorithm 1: Iterative optimality certifier

Data: Solution to problem QCQP $\boldsymbol{x}$; data matrix $\boldsymbol{C}$; set of constraints $\{\boldsymbol{A}_i\}_i$; initial lagrange multipliers $\boldsymbol{\lambda}_0$ (optional)
Result: lagrange multipliers $\boldsymbol{\lambda}$; Hessian $\boldsymbol{S}$; optimality certificate isOpt

```
// Obtain initial guess
```
$(\boldsymbol{\lambda}_0, \boldsymbol{S}_0) \leftarrow$ Solve system $\boldsymbol{J}(\boldsymbol{x})\boldsymbol{\lambda} = \boldsymbol{b}(\boldsymbol{x})$ for $\boldsymbol{\lambda}_0$ with minimum 2-norm;
Decompose solution $\boldsymbol{S}_0$ to obtain $\boldsymbol{Y}_0$ with rank k;
// $j = 0$
repeat
  $(\boldsymbol{\lambda}_{j+1}, \boldsymbol{Y}_{j+1}) \leftarrow$ Solve problem R-Y with $(\boldsymbol{\lambda}_j, \boldsymbol{Y}_j)$
  if *cost below threshold* **then**
    `// Solution is optimal`
    isOpt = True ;
    $\boldsymbol{\lambda}^\star \leftarrow \boldsymbol{\lambda}_{j+1}$;
    $\boldsymbol{S}^\star \leftarrow \boldsymbol{S}_{j+1}$;
  else if *we can continue* **then**
    `// Test another solution`
    if $\boldsymbol{Y}_{j+1}$ *rank deficient* **then**
      `// Reduce rank`
      $k \leftarrow k - 1$;
      Update $\boldsymbol{Y}_{j+1}$ with rank k;
    else
      `// Increase rank`
      $k \leftarrow k + 1$;
      Update $\boldsymbol{Y}_{j+1}$ with rank k;
  else
    `// We couldn't test more solutions`
    isOpt = UNKNOWN
until *convergence or max. iters or max. rank*;

Estimation of the essential matrix: As with the certifier, we propose to estimate the essential matrix by optimizing on its manifold. Since the process is similar to the general case devised in [32], we provide here only the basic information and refer the reader to the *Supplementary material* section E for more information. For the solver we employ again the trust-region method.

Domain: Regarding the domain, we employ the classic representation based on the rotation and translation up-to-scale. For the essential matrix following the definition in eq. (4), we have that $\boldsymbol{E}_{\mathrm{Y}} = [\boldsymbol{t}]_{\times}\boldsymbol{R}_{\mathrm{Y}}$, where $\boldsymbol{R}_{\mathrm{Y}}$ is a 2D rotation as in eq. (2). Hence, our domain for this problem is $\mathcal{S}^2 \times \mathrm{SO}(2)$. It is well-known that in the general case some issues may arise with this simplification, since each essential matrix corresponds to four different configurations of translation and rotation [33]. Two different rotations are associated with the same essential matrix: one of them is the "true" rotation and the other one is the result of the rotation followed by a reflection w.r.t. the translation

Algorithm 2: Increasing rank of solution $\boldsymbol{Y}_{og}$

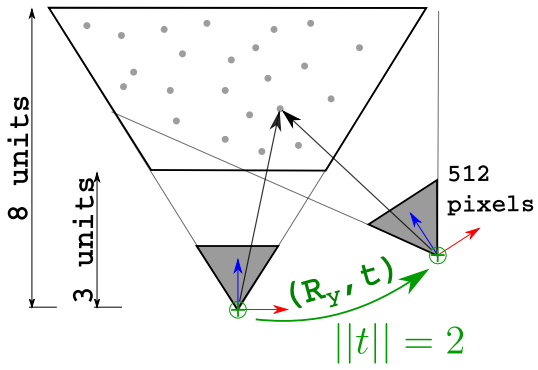
Data: Solution to problem R-Y with rank r; initial Hessian $\boldsymbol{S}_0$
Result: Solution $\boldsymbol{Y}_{new}$ with rank $r+1$
// Run linear-search algorithm
$\alpha \leftarrow 16e-6$;
// Compute cost for original solution
$f_{og} = cost(\boldsymbol{Y}_{og})$;
// Compute search direction
$\boldsymbol{v}_{min} \leftarrow$ searchDirection($\boldsymbol{Y}_{og}$) (see *Supplementary material* C);
$i \leftarrow 0$;
repeat
  // Obtain new solution along direction
  $\boldsymbol{Y}_i \leftarrow$ retract($\boldsymbol{Y}_{og}, \alpha \boldsymbol{v}_{min}$) ;
  // Compute cost
  $f_i \leftarrow cost(\boldsymbol{Y}_i)$;
  if $f_i < f_{og}$ *and* $rank(\boldsymbol{Y}_i) \equiv r+1$ **then**
    // Accept this solution
    $\boldsymbol{Y}_{new} \leftarrow \boldsymbol{Y}_i$;
    break ;
  $\alpha \leftarrow \alpha/2$;
  $i \leftarrow i+1$;
until *while* $i <$ *maxIters and solution not improved*;
if *Linear-search failed, use initial Hessian* $\boldsymbol{H}_0$ **then**
  // Decompose initial Hessian $\boldsymbol{H}_0$
  $(\boldsymbol{U}, \boldsymbol{D}^2) \leftarrow$ eigenDecomposition($\boldsymbol{H}_0$) ;
  $\boldsymbol{Y}_{new} \leftarrow \boldsymbol{U}_{[:,1:r+1]} \boldsymbol{D}_{[:,1:r+1]}$;

vector. In this work we restrict the rotation space to 2D, and hence the reflected rotation does not appear since it will be not a Y-rotation in general. In practice, we did not observe any additional solution to the problem nor a decrease on the performance of the optimization. A theoretical proof of the existence or not of any symmetric solution is out of the scope of the present manuscript.

Model: The employed solver requires second-order information about the model, that is, gradient and (vector-product) Hessian in terms of the variables of the problem. Since we consider the domain as the product of manifolds, we can define the gradient and Hessian separately for each variable. We leverage the full model and extend the 2D rotation to a 3D rotation. This allows us to re-use the model derived in [32] with minor modifications.

5. Evaluation

In this last section we evaluate the proposed iterative certifier and the on-manifold variable estimation, and compare the latter with the state-of-the-art works in different



(a)

Form.	Mean (median)
LEFT	1.1674 (0.548)
RIGHT	1.0872 (0.509)
BOTH	1.2415 (0.632)
ADJ	1.4336 (0.768)

(b)

Figure 3: Fig. 3a shows the configuration of the points, cameras and parameters for the considered problem instances used in the synthetic evaluation. Table 3b reports the computational times (in milliseconds) required by the certifier for each formulation for the configurations with pure Y-rotation $\boldsymbol{R}_{\mathrm{Y}}$.

situations. We consider the minimal solver denoted by MIN-E by [19] and the non-minimal, optimal solver denoted by OPT-E in [9]. From all the proposed options in that work, we select the one which is not based on Sturm sequences since it appears to be more stable although slower. The minimal solver MIN-E was taken from the SfM library Theia [4] (by the same main author), and the solver for the non-minimal OPT-E was kindly provided by the authors. Since the minimal solver MIN-E requires only three points, we sample five random subsets from the whole set and keep the solution with the lowest cost. Additionally, we will provide also the results given by the initialization DLT-E employed for the on-manifold estimation. This initialization is similar to the classic DLT method, but we impose the constraints in the set (4) by operating over the corresponding columns and rows of the full 9×9 matrix to obtain a 6×6 matrix.

5.1. Evaluation on synthetic data

We place the first camera frame at the origin (identity orientation and zero translation) and generate a set of random 3D points within a frustum with depth ranging from three to eight units measured from the first camera frame and inside its Field of View (FoV) of 100 degrees. Then, we generate a random pose for the second camera such that the relative rotation is a Y-rotation. The relative translation depends on the set of experiments which we define later. We enforce that all the 3D points lie within the second camera's FoV (also 100 degrees). Figure 3a depicts a simple scene. We create the correspondences with a pin-hole camera model and add Gaussian noise with deviation σ (in pixels). The noise is applied on the normalized image plane, considering a focal length of 512 pixels for both cameras. After the perturbation, the observations are normalized to have Euclidean norm one. We consider three different configurations in terms of the relative translation: (a) general translation, where we assign to the translation a random vector with Euclidean norm two; (b) forward translation, where the second camera moves two units in the Z-direction; and (c) lateral translation, similar

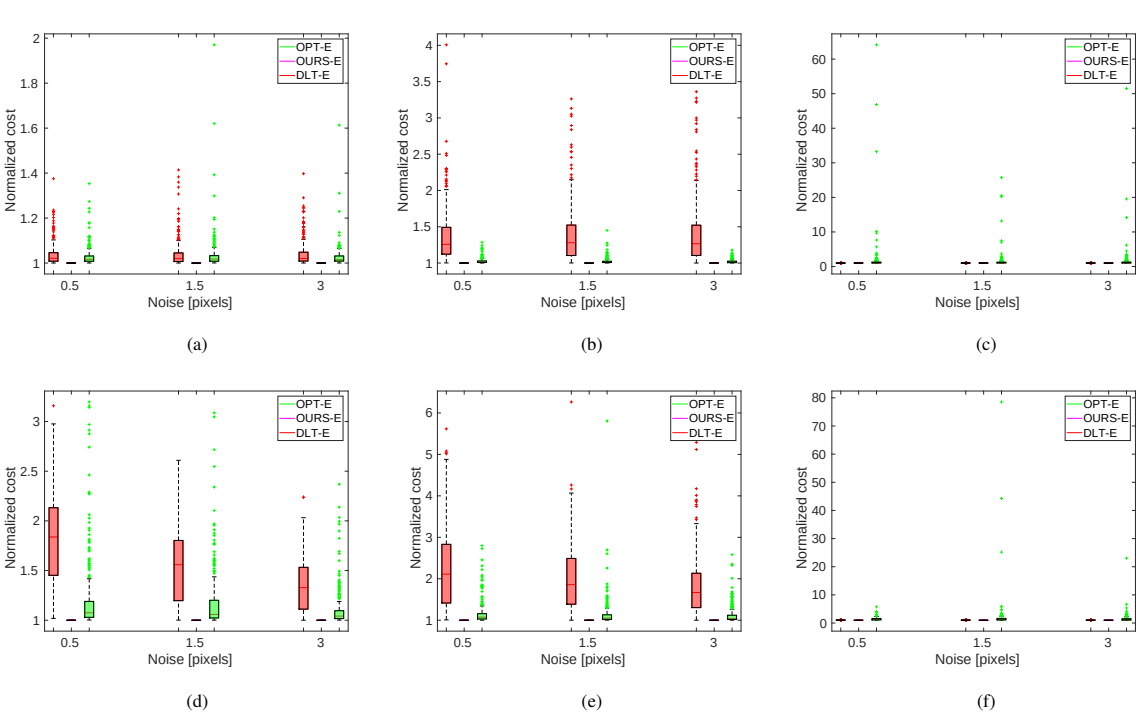


Figure 4: **Essential matrix: Normalized cost**: Cost of the solutions normalized by the one for OURS-E (in magenta). We don't include the results for MIN-E for being too large in comparison (see *Supplementary material* section G where these results are included). Values above one indicates that the cost for the solver was larger than that for OURS-E. Top row shows the experiments with pure Y-rotation and bottom row the experiments where a perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis). The number of correspondences is fixed to $N = 50$. Results with zero noise are not shown for being numerically zero for all the solvers. Note the scale difference between general and lateral motion and forward motions (right column), and the number of outliers returned by OPT-E.

to a classic stereo configuration, the second camera is placed along the X-direction at two units of distance from the first one.

We will also consider perturbation on the relative rotation by pre-multiplying a random rotation with angle $\theta_{\boldsymbol{R}} \leq 0.01[rad]$ to the original Y-rotation. For the different setups we also vary the number of correspondences from $N \in \{10, 15, 50, 100, 200\}$ and noise $\sigma \in \{0.0, 0.5, 1.5, 3.0\}$. Since we maintain the focal length fixed, varying the noise changes the Signal-to-noise ratio. For each type of configuration and set of parameters, we generate 300 random problem instances.

5.1.1. Results for the variable estimation

While the rotation error is an intuitive measurement of the quality of the solution, the cost attained by the solutions indicates which one is actually closer to the true global optimum, which is the main goal of this work. To limit the length of the main

Solver	Synthetic data (μs)	Real data (μs)
DLT-E	38.9434 (29)	78.3212(71)
OURS-E	77.6539 (73)	140.733(124)
MIN-E	10.9514 (10)	7.0121(6)
OPT-E	214.2631 (207)	332.0242(169)

Table 1: Computational times in microseconds required as mean(median) by the solvers for (a) synthetic data (first column); and (b) real data (second column).

manuscript, we only include here the results with the costs and include in the *Supplementary material* section F these rotation errors for the considered configurations. The cost attained by each solver normalized by OURS-E for all the configurations for problem instances with $N = 50$ correspondences as a function of the noise (X-axis) is shown in figure 4. *Supplementary material* section G includes more results, including a comparison with the minimal MIN-E, which are not included in the manuscript due to space limits. Values above one indicate that the cost attained by the solver was larger than the one for OURS-E. For all our experiments, this was always the case even with OPT-E except in six problem instances with forward motion and noisy $\boldsymbol{R}_{\mathrm{Y}}$: 3 for $N = 10$ and 3 for $N = 15$ (results are included in *Supplementary material*). For those cases, OURS-E returns solutions with larger cost than OPT-E, but lower than MIN-E. Further, for the general and forward motion the normalized costs of our initial guess DLT-E and the solver OPT-E are similar. For lateral motions, though, OPT-E performs better than DLT-E. In general, we observe some unstable results for OPT-E, seeing as outliers in fig. 4, even for problem instances with a non-minimal number of correspondences and/or low noise. Further, even though OPT-E is technically an 'optimal' solver, we observe empirically that a solution with lower cost exists (which is found by our proposal), mainly due to numerical instabilities/inaccuracies. This behavior highlights the necessity of a refinement of the solution, as the one proposed in this manuscript, even for 'optimal' solvers.

Computational time: Table 1 collects the computational time required by each solver as mean(median) in microseconds for (a) the synthetic data (first column); and (b) real data (second column). We didn't observe any considerable variance of the times with respect to the configurations and hence, all the motions are considered to compute the above-mentioned metrics. Notice that the proposed solver is faster than OPT-E.

5.1.2. Results for the iterative certifier

Quality of initialization

We measure the quality of the initialization by the minimum eigenvalue of the matrix $\boldsymbol{S}$ and the sum of the eigenvalues which are zeroed. We move the results to the *Supplementary material* section F and include here the main observations. For zero noise, all the formulations have zero ($\sim 1e-17$) as minimum eigenvalue. Generally the eigenvalue increases with the level of noise for all the camera configurations, although the effect is more apparent with the formulations with fewer constraints. In most problem instances all the formulations attain minimum eigenvalues below $1e-04$. In terms

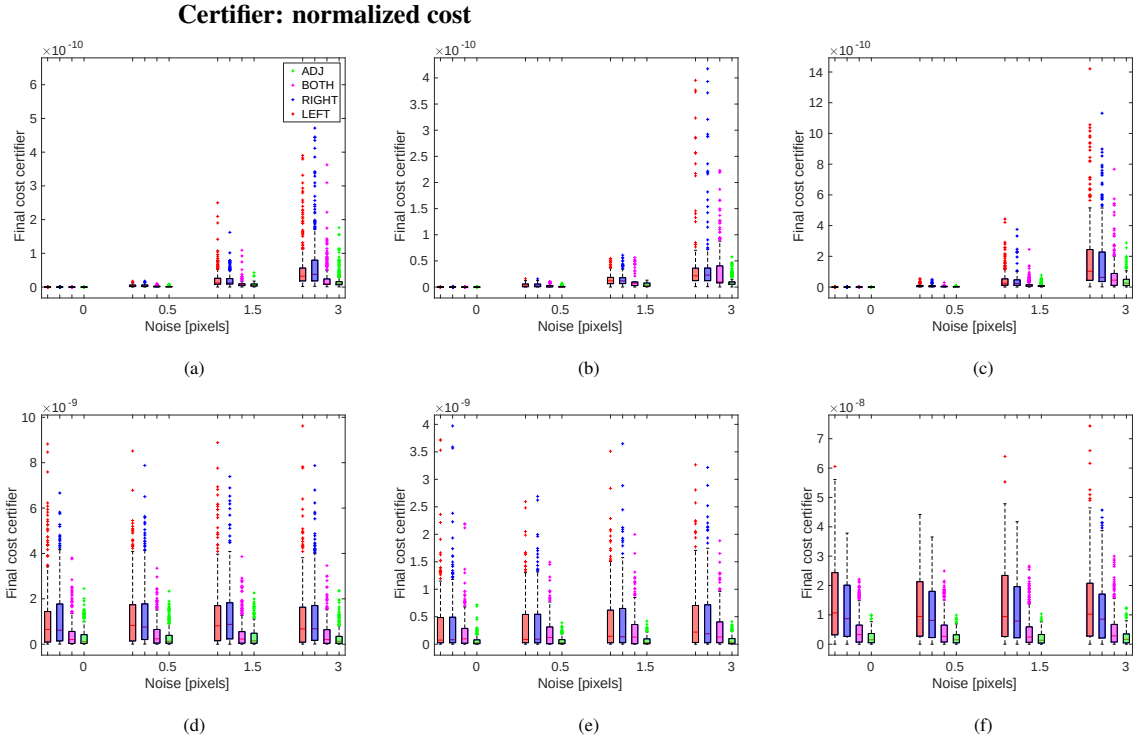


Figure 5: **Certifier: normalized cost**: Normalized cost as in eq. (11) for the different formulations (see legend), level of noise (see X-axis) and the three considered configurations (from left to right): general, lateral and forward translation. First row shows the results for pure Y-rotation and the second row those for noisy Y-rotation. We consider the solution as optimal when the cost drops below $1e-11$. Notice the difference in the Y-scales.

of configurations we observe a similar behavior for the general and lateral movements, while forward returns the worst solutions.

For noisy Y-rotation, the eigenvalue increases for all the configurations including with zero noise, being in average an order of magnitude larger than with the noiseless Y-rotation. Forward movements are still the ones with the worse performance, reaching eigenvalues circa $1e-03$.

Performance of certifier: Problem (R-Y)

Figure 5 shows the normalized costs for the different configurations for $N = 50$ correspondences. The values of the costs are included in the *Supplementary material* section F. From fig. 5 we notice that the minimal formulations LEFT and RIGHT seem to perform worse, but still are able to certify solutions even with large noise $\sigma = 3$. The redundant formulations BOTH and ADJ have similar performance regardless of the camera configuration and certify most of the solutions. For noisy Y-rotations, this difference is more notable. For these cases the noise applied to the observations does not seem to affect the certifiers, but the camera configuration plays an important role. Lateral movements seem to be less critical for the certifier and still some solutions are

certified. Forward movements on the other side increase the normalized cost up to $1e-08$ for the minimal formulations. In general the minimal formulations detect fewer optimal solutions than the redundant one, which returns optimality certificates even for large noise and low number of correspondences, highlighting the importance of this type of certifiers for these particular challenging scenarios. We include the number of certified solutions for these synthetic experiments in *Supplementary material* section F.

Comparison with SDP solver: We compare the performance of our iterative certifier with the one obtained by solving the SDP in prob. R-F with off-the-shelf tools, in this case, with **SDPT3** [27] modeled by **CVX** [45, 46]. Notice that the solution attained by the SDP isn't necessarily the same than the one obtained by our approach. Hence, we only compare the costs for both approaches for problem R-Y (f_{proj} for the cost attained by the proposal and f_{sdp} for the cost by the off-the-shelf tools), which are included in figure 2b as the cumulative distribution of the differences. We observe that the costs are practically the same for all problem instances, although they are closer for the redundant formulations. Therefore, our heuristic certifier is able to obtain the global solution for problem R-Y, and hence a valid optimality certificate for the RPp.

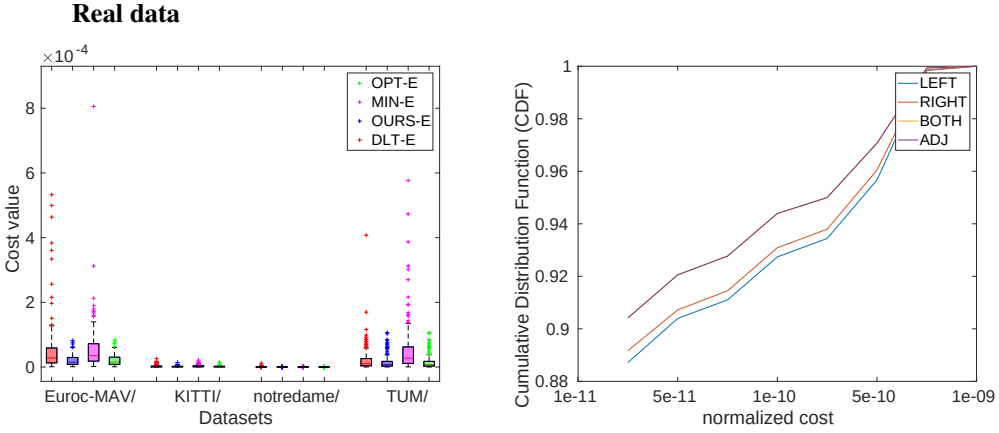
Computational time: To finalize the evaluation on synthetic data we summarize here the computational time required by each formulation. These metrics only consider cases with certified optimal solutions, since the algorithm only stops when all the possible ranks for the Hessian are checked. Thus, these metrics can be also employed by the user to stop the certification if the execution time is too large w.r.t. that expected. We didn't observe any substantial variation of the times with respect to the camera configurations. However, the level of noise and number of correspondences do impact the performance, also affecting the number of iterations required by the certifier. For the extreme cases with large noise (*e.g.* 3 pixels), and low number of correspondences (*e.g.* 10), the averaged number of iterations goes up to 2 for all the formulations. For more common cases, the mean remains below 1.2 for all the configurations. For noisy Y-rotation, the level of noise and number of correspondences does not affect the certifier and the averaged number of iterations reaches 2 for all formulations, although it is slightly better for ADJ. Nevertheless, for all the configurations, noise and number of correspondences the median number of iterations is one, even for noisy Y-rotations.

Table 3b shows the mean and median times for each formulation for problem R-Y. Recall that these values include all the noise level and number of correspondences, which, as explained above, affect the certifier. We must highlight here that for the problem instances with noiseless Y-rotation the certifier returns a solution in 1.5 milliseconds, even for the over-constrained formulations.

5.2. Evaluation on real data

To conclude this Section we evaluate our proposal on large scale real datasets which are publicly available. The employed datasets are the next:

1. KITTI dataset [47]: The sequences were recorded by a moving vehicle with a forward pointing camera. The next sequences are considered: ROAD-2011-09-26-DRIVE-0027-SYNC denoted by KITTI: ROAD-27; ROAD-2011-09-30-DRIVE-0016-SYNC denoted by KITTI: ROAD-30; ROAD-2011-09-26-DRIVE-0029-SYNC denoted



(a) (b)

Figure 6: **Real data**: Cost value for the considered solvers for all the datasets is shown in fig. 6a. Fig. 6b shows the Cumulative sum (normalized) of the normalized cost returned by the certifiers. The graphics for BOTH and ADJ overlap.

by KITTI: ROAD-29; and ROAD-2011-09-26-DRIVE-0032-SYNC denoted by KITTI: ROAD-32.

2. TUM dataset [48]: The images were taken by hand-held cameras and show different scenes with texture. We consider the sequences: FREIBURG3_TEDDY denoted by TUM-TEDDY; FREIBURG3-STRUCTURE-TEXTURE-FAR denoted by TUM-STRUCT-FAR; and FREIBURG3-STRUCTURE-TEXTURE-NEAR denoted by TUM-STRUCT-NEAR.
3. Notredame [2]: The unordered images show Notredame from different point of views and were taken by hand-held cameras. There are 715 images. Since the set is not sequential, we take all the combinations.
4. EuRoC-MAV dataset [49]: The images were taken by an autonomous aerial vehicle in a machine hall. We consider all the sequences: MH-01-EASY denoted by MH1; MH-02-EASY denoted by MH2; MH-03-MEDIUM denoted by MH3; MH-04-DIFFICULT denoted by MH4; and MH-05-DIFFICULT denoted by MH5.

We want to recall that this work focused on the certification of the solution, rather than in the robustness of the solvers. Therefore, we do not contemplate wrong correspondences (outliers) on the data and filter the matches with the provided ground truth before feeding it to the solvers. We keep all the problem instances with at least 50 correspondences. Nevertheless, the proposed solvers are suitable for being embedded into robust nonminimal paradigms. The gravity prior in these sequences is estimated and applied as follows: We obtain a 2-decimal approximation of the ground-truth gravity vector. This limitation on the accuracy of the measurement allows us to model noisy vectors, as in the previous Section. We obtain the associated rotation, which is later applied to the observations. These operations on the observations provide us

[2] http://phototour.cs.washington.edu/datasets/

with 'noisy' problem instances, which are closer to those encountered in real-world applications. Further, for the KITTI dataset we assume that the cameras are already aligned, *i.e.* $\boldsymbol{R}_Y = \boldsymbol{I}_3$ since the movement is supposed to be 'planar', although this doesn't match the ground-truth results. We believe this last case is relevant since it's a common assumption for this kind of configurations.

Figure 6a shows the cost of the returned solution for the different datasets (considering all the sequences) by all the solvers. As with the synthetic data, the minimal solver MIN-E attains large errors. The linear method DLT-E performs well in most problem instances, while the non-minimal OURS-E and OPT-E return similar solutions, although our proposal always attain the lowest cost of all the methods.

Figure 6b depicts the cumulative sum (normalized) of the normalized cost of the certifiers for all the considered datasets. The behavior of the formulations doesn't change on the real data, with the redundant BOTH and ADJ being able to certify more solutions that the minimal LEFT and RIGHT.

In terms of computational time, we didn't observe any substantial change with respect the synthetic experiments and all the formulations, including the redundant ones, return an optimality certification in less than 1.5 milliseconds.

6. Conclusion and future work

In this work we have tackled the Relative Pose problem (RPp) between two central cameras with known gravity vector. This information allows to reduce the three degrees of freedom for the rotation to only one. Our proposal relied on a efficient estimation of the essential matrix and a fast iterative optimality certifier based on the dual problem. This certifier leveraged the low-rank decomposition of the Hessian of the Lagrangian. Since the certifier depended on the formulation of the set of essential matrices, we provided four different set of constraints that were shown experimentally to perform different in terms of certification. We evaluated our proposal exhaustively on synthetic and real data. The estimation of the variable was shown to return solutions with lower cost than other nonminimal solvers while being more stable. The certifier worked with large noise and returned optimality certificates for all formulations in less than 1.5 millisecond.

As future work we contemplate the extension of this certifier to the N-view triangulation problem, and introduce our proposal in robust paradigms, such as Graduated Non-convexity (GNC) [50] (with the Black-Rangarajan duality [51]).

Acknowledgments

This work has been supported by the grant program FPU18/01526 and the research projects ARPEGGIO (PID2020-117057) and HOUNDBOT (P20-01302) funded by the Spanish and Andalusian Governments. The authors thank the anonymous reviewers whose comments and suggestions greatly improved the present manuscript.

References

[1] T. Taketomi, H. Uchiyama, S. Ikeda, Visual slam algorithms: a survey from 2010 to 2016, IPSJ Transactions on Computer Vision and Applications 9 (1) (2017) 16.

[2] C. Kerl, J. Sturm, D. Cremers, Dense visual slam for rgb-d cameras, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 2100–2106.

[3] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, D. Scaramuzza, J. Gonzalez-Jimenez, Pl-slam: A stereo slam system through the combination of points and line segments, IEEE Transactions on Robotics 35 (3) (2019) 734–746.

[4] C. Sweeney, Theia multiview geometry library: Tutorial & reference, http://theia-sfm.org.

[5] R. Mur-Artal, J. M. M. Montiel, J. D. Tardos, Orb-slam: a versatile and accurate monocular slam system, IEEE transactions on robotics 31 (5) (2015) 1147–1163.

[6] L. Kneip, S. Lynen, Direct optimization of frame-to-frame rotation, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 2352–2359.

[7] J. Briales, L. Kneip, J. Gonzalez-Jimenez, A certifiably globally optimal solution to the non-minimal relative pose problem, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 145–154.

[8] J. Zhao, An efficient solution to non-minimal case essential matrix estimation, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020). doi:10.1109/TPAMI.2020.3030161.

[9] Y. Ding, D. Barath, J. Yang, H. Kong, Z. Kukelova, Globally optimal relative pose estimation with gravity prior, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 394–403.

[10] A. S. Bandeira, A note on probably certifiably correct algorithms, Comptes Rendus Mathematique 354 (3) (2016) 329–333.

[11] L. Carlone, D. M. Rosen, G. Calafiore, J. J. Leonard, F. Dellaert, Lagrangian duality in 3d slam: Verification techniques and optimal solutions, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 125–132.

[12] A. Eriksson, C. Olsson, F. Kahl, T.-J. Chin, Rotation averaging and strong duality, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 127–135.

[13] M. Garcia-Salguero, J. Briales, J. Gonzalez-Jimenez, A tighter relaxation for the relaitve pose problem between cameras, http:/....

[14] D. Nistér, An efficient solution to the five-point relative pose problem, IEEE transactions on pattern analysis and machine intelligence 26 (6) (2004) 756–770.

[15] H. Stewenius, C. Engels, D. Nistér, Recent developments on direct relative orientation, ISPRS Journal of Photogrammetry and Remote Sensing 60 (4) (2006) 284–294.

[16] Z. Kukelova, M. Bujnak, T. Pajdla, Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems., in: BMVC, Vol. 2, 2008, p. 2008.

[17] F. Fraundorfer, P. Tanskanen, M. Pollefeys, A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles, in: European Conference on Computer Vision, Springer, 2010, pp. 269–282.

[18] Y. Ding, J. Yang, H. Kong, An efficient solution to the relative pose estimation with a common direction, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 11053–11059.

[19] C. Sweeney, J. Flynn, M. Turk, Solving for relative pose with a partially known rotation is a quadratic eigenvalue problem, in: 2014 2nd International Conference on 3D Vision, Vol. 1, IEEE, 2014, pp. 483–490.

[20] U. Helmke, K. Hüper, P. Y. Lee, J. Moore, Essential matrix estimation using gauss-newton iterations on a manifold, International Journal of Computer Vision 74 (2) (2007) 117–136.

[21] Y. Ma, J. Košecká, S. Sastry, Optimization criteria and geometric algorithms for motion and structure estimation, International Journal of Computer Vision 44 (3) (2001) 219–249.

[22] V. Lui, T. Drummond, An iterative 5-pt algorithm for fast and robust essential matrix estimation., in: BMVC, 2013.

[23] R. Tron, K. Daniilidis, The space of essential matrices as a riemannian quotient manifold, SIAM Journal on Imaging Sciences 10 (3) (2017) 1416–1445.

[24] T. Botterill, S. Mills, R. Green, Refining essential matrix estimates from ransac, in: Proceedings Image and Vision Computing New Zealand, 2011, pp. 1–6.

[25] R. I. Hartley, F. Kahl, Global optimization through searching rotation space and optimal estimation of the essential matrix, in: 2007 IEEE 11th International Conference on Computer Vision, IEEE, 2007, pp. 1–8.

[26] J. F. Sturm, Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones, Optimization methods and software 11 (1-4) (1999) 625–653.

[27] K.-C. Toh, M. J. Todd, R. H. Tütüncü, Sdpt3—a matlab software package for semidefinite programming, version 1.3, Optimization methods and software 11 (1-4) (1999) 545–581.

[28] N. Boumal, A riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints, arXiv preprint arXiv:1506.00575 (2015).

[29] D. M. Rosen, Scalable low-rank semidefinite programming for certifiably correct machine perception, in: Intl. Workshop on the Algorithmic Foundations of Robotics (WAFR), Vol. 3, 2020, p. 3.

[30] H. Yang, L. Liang, K.-C. Toh, L. Carlone, Stride along spectrahedral vertices for solving large-scale rank-one semidefinite relaxations, arXiv preprint arXiv:2105.14033 (2021).

[31] J. Briales, J. Gonzalez-Jimenez, Convex global 3d registration with lagrangian duality, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4960–4969.

[32] M. Garcia-Salguero, J. Gonzalez-Jimenez, Fast and robust certifiable estimation of the relative pose between two calibrated cameras, Journal of Mathematical Imaging and Vision (2021) 1–21.

[33] R. Hartley, A. Zisserman, Multiple view geometry in computer vision, Cambridge university press, 2003.

[34] S. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.

[35] D. Cifuentes, A. Moitra, Polynomial time guarantees for the burer-monteiro method, arXiv preprint arXiv:1912.01745 (2019).

[36] S. Burer, R. D. Monteiro, Local minima and convergence in low-rank semidefinite programming, Mathematical programming 103 (3) (2005) 427–444.

[37] F. Alizadeh, J.-P. A. Haeberly, M. L. Overton, Complementarity and nondegeneracy in semidefinite programming, Mathematical programming 77 (1) (1997) 111–128.

[38] J. Dattorro, Convex optimization & Euclidean distance geometry, Lulu. com, 2010.

[39] E. de Klerk, C. Roos, T. Terlaky, Initialization in semidefinite programming via a self-dual skew-symmetric embedding, Operations Research Letters 20 (5) (1997) 213–221.

[40] N. Boumal, B. Mishra, P.-A. Absil, R. Sepulchre, Manopt, a Matlab toolbox for optimization on manifolds, Journal of Machine Learning Research 15 (2014) 1455–1459.
URL http://www.manopt.org

[41] D. M. Rosen, L. Carlone, A. S. Bandeira, J. J. Leonard, Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group, The International Journal of Robotics Research 38 (2-3) (2019) 95–125.

[42] M. Journée, F. Bach, P.-A. Absil, R. Sepulchre, Low-rank optimization on the cone of positive semidefinite matrices, SIAM Journal on Optimization 20 (5) (2010) 2327–2351. doi:10.1137/080731359.

[43] E. Massart, P.-A. Absil, Quotient geometry with simple geodesics for the manifold of fixed-rank positive-semidefinite matrices, Tech. Rep. UCL-INMA-2018.06-v2, UCLouvain, preprint: http://sites.uclouvain.be/absil/2018.06 (2018).

[44] P.-A. Absil, R. Mahony, R. Sepulchre, Optimization algorithms on matrix manifolds, Princeton University Press, 2009.

[45] M. Grant, S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1, http://cvxr.com/cvx (Mar. 2014).

[46] M. Grant, S. Boyd, Graph implementations for nonsmooth convex programs, in: V. Blondel, S. Boyd, H. Kimura (Eds.), Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 2008, pp. 95–110.

[47] [dataset] Andreas Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The kitti dataset, International Journal of Robotics Research (IJRR) (2013).

[48] J. Sturm, N. Engelhard, F. Endres, W. Burgard, D. Cremers, A benchmark for the evaluation of rgb-d slam systems, in: Proc. of the International Conference on Intelligent Robot Systems (IROS), 2012.

[49] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, R. Siegwart, The euroc micro aerial vehicle datasets, The International Journal of Robotics Research 35 (10) (2016) 1157–1163.

[50] A. Blake, A. Zisserman, Visual reconstruction, MIT press, 1987.

[51] M. J. Black, A. Rangarajan, On the unification of line processes, outlier rejection, and robust statistics with applications in early vision, International journal of computer vision 19 (1) (1996) 57–91.

Appendices

A. Essential matrix with form (4) has a Y-rotation.

In this section we show that any matrix with the form given in (4) has a Y-rotation when decomposed as $[\boldsymbol{t}]_{\times}\boldsymbol{R}$. Matrices with this form have a specific pattern given by

$$\boldsymbol{E}_{\mathrm{Y}} = \begin{pmatrix} e_1 & e_2 & e_3 \\ e_4 & 0 & e_5 \\ -e_3 & e_6 & e_1 \end{pmatrix}, \tag{12}$$

where the Y-rotation is

$$\boldsymbol{R}_{\mathrm{Y}} = \begin{pmatrix} c & 0 & s \\ 0 & 1 & 0 \\ -s & 0 & c \end{pmatrix}, \tag{13}$$

and $c, s \in \mathbb{R}$ such that $c^2 + s^2 = 1$

$\Longrightarrow$: Consider a 3D vector $\boldsymbol{t} \in \mathcal{S}^2$ and the Y-rotation as in (2). The associated essential matrix has the form

$$\boldsymbol{E} = [\boldsymbol{t}]_{\times}\boldsymbol{R} = \begin{pmatrix} -t_2 s & -t_3 & t_2 c \\ t_3 c + t_1 s & 0 & t_3 s - t_1 c \\ -t_2 c & t_1 & -t_2 s \end{pmatrix}, \tag{14}$$

where the matrix $[\boldsymbol{t}]_{\times}$ is defined as

$$[\boldsymbol{t}]_{\times} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{15}$$

The form for $\boldsymbol{E}_{\mathrm{Y}}$ follows from the above relation.

$\Longleftarrow$: Consider a 3×3 essential matrix $\boldsymbol{E} = [e_{i,j}]_{i,j=1}^{3}$ whose elements fulfill the constraints in eq. (4). Since $\boldsymbol{E}$ is an essential matrix, we can decompose it as the product $[\boldsymbol{t}]_{\times}\boldsymbol{R}$ for $\boldsymbol{t} \in \mathcal{S}^2$ and a 3D rotation matrix $\boldsymbol{R} = [r_{i,j}]_{i,j=1}^{3} \in \mathrm{SO}(3)$. For $\boldsymbol{E}$ to have a Y-rotation as $\boldsymbol{R}$, three additional relations must be fulfilled. These requirements can be written in terms of the elements of $\boldsymbol{t}$ and $\boldsymbol{R}$ as

$$e_{1,1} - e_{3,3} = 0 \qquad \Leftrightarrow -t_3 r_{2,1} + t_2 r_{3,1} + t_2 r_{1,3} - t_1 r_{2,3} = 0 \tag{16}$$

$$e_{1,3} + e_{3,1} = 0 \qquad \Leftrightarrow -t_3 r_{2,3} + t_2 r_{3,3} - t_2 r_{1,1} + t_1 r_{2,1} = 0 \tag{17}$$

$$e_{2,2} = 0 \qquad \Leftrightarrow t_3 r_{1,2} - t_1 r_{3,2} = 0 \tag{18}$$

We can re-write these expressions as

$$\underbrace{\begin{pmatrix} -r_{3,2} & 0 & r_{1,2} \\ r_{2,1} & r_{3,3} - r_{1,1} & -r_{2,3} \\ -r_{2,3} & r_{3,1} + r_{1,3} & -r_{2,1} \end{pmatrix}}_{\boldsymbol{K_R}} \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} = \boldsymbol{0}_{3\times 1} \tag{19}$$

Since $\boldsymbol{t} \in \mathcal{S}^2$ it cannot be the null vector due to the norm constraint. It follows that the 3×3 matrix $\boldsymbol{K_R}$ has at most rank 2. Next we analyze each possible rank.

Rank zero: Assume that $\boldsymbol{K_R}$ has zero rank, and so $\boldsymbol{K_R} = \mathbf{0}_{3\times3}$. This implies that $r_{3,2} = r_{1,2} = r_{2,1} = r_{2,3} = 0$, and $r_{3,3} = r_{1,1}$, $r_{3,1} = -r_{1,3}$. Since $\boldsymbol{R}$ is a rotation matrix, due to the orthogonal constraint we have that $r_{2,2} = \pm 1$. Filling the entries of the matrix with these values and defining $c \doteq r_{1,1}$ and $s \doteq r_{1,3}$, we obtain a Y-rotation when $r_{2,2} = +1$ and a Y-rotation with the same angle followed by a reflection around the Y-axis if $r_{2,2} = -1$. The determinant of the latter is not positive, hence being an orthogonal transformation and not a rotation as assumed. Then, $\boldsymbol{K_R}$ with rank zero implies that $\boldsymbol{R}$ is a Y-rotation.

Rank one: Assume that $\boldsymbol{K_R}$ as rank 1. Thus, we can decompose it as the outer product of two real 3D vectors $\boldsymbol{u}, \boldsymbol{v}$ as $\boldsymbol{K_R} = \boldsymbol{u}\boldsymbol{v}^T$. Observe that $\boldsymbol{K_R}_{1,2} = 0$; then, either $\boldsymbol{v}_2 = 0$ or $\boldsymbol{u}_1 = 0$ (or both at the same time).

Consider $\boldsymbol{u}_1 = 0$, which leads to the first row of $\boldsymbol{K_R}$ to be zero. Then $r_{3,2} = r_{1,2} = 0$. Since $\boldsymbol{R}$ is still a rotation matrix, we have that $r_{2,2} = \pm 1$ and in turn $r_{2,3} = r_{2,1} = 0$. Since $\boldsymbol{R}$ must be a rotation, we have that $r_{2,2} = 1$, $r_{1,1} = r_{3,3}$ ans $r_{1,3} = -r_{3,1}$. Then, $\boldsymbol{K_R}$ has rank zero and not one, and we fall onto the previous case where $\boldsymbol{R}$ was a Y-rotation by construction.

Let us now consider $\boldsymbol{v}_2 = 0$. This leads to the second column of $\boldsymbol{K_R}$ being zero. Then, $r_{3,3} = r_{1,1}$ and $r_{3,1} = -r_{1,3}$. Since $\boldsymbol{R}$ is still a rotation matrix, any column and row has (Euclidean) norm one, and so $r_{1,2} = r_{3,2} = r_{2,1} = r_{2,3}$. We also have that

$$r_{1,1}r_{3,1} + r_{1,2}r_{3,2} + r_{1,3}r_{3,3} = 0 \implies r_{1,2}^2 = 0, \tag{20}$$

and so $r_{1,2} = 0$, leading to the two previous scenarios with $\boldsymbol{R}$ being a Y-rotation. Then, the rank one condition cannot be achieved with $\boldsymbol{v}_2 = 0$ either. We conclude that $\boldsymbol{K_R}$ does not have rank one under any condition.

Rank two: Last, we contemplate the case in which $\boldsymbol{K_R}$ has rank 2. As we have shown previously, a Y-rotation leads to a matrix $\boldsymbol{K_R} = \mathbf{0}_{3\times3}$. Then, we will assume that $\boldsymbol{R}$ is not a Y-rotation. Consider then a $\boldsymbol{K_R}$ which is not the zero matrix but has rank-two. Its right-nullspace is then one-dimensional and it is given by $\boldsymbol{t}$.

Recall that the set of constraints imposed by eq. (4) are functions of the essential matrix, not the rotation matrix. From classic computer vision theory we know that two rotations are associated with the same essential matrix, $\boldsymbol{R}$ and $\boldsymbol{PR}$, being $\boldsymbol{P}$ the reflection w.r.t. the translation vector $\boldsymbol{t}$ (the right nullspace of $\boldsymbol{K_R}$) defined as $\boldsymbol{P} = 2\boldsymbol{t}\boldsymbol{t}^T - \boldsymbol{I}_3$. Therefore, if $\boldsymbol{R}$ fulfills the constraints, so does $\boldsymbol{R_P} \doteq \boldsymbol{PR}$ for the appropriated $\boldsymbol{t}$. From the above development, we know that the matrix $\boldsymbol{K_R}$ evaluated at $\boldsymbol{R_P}$ must have either rank 0 or rank 2.

Consider the (2,2) entry of this matrix, denoted by $(\boldsymbol{R_P})_{2,2}$. This entry has the form (in terms of $\boldsymbol{R}$ and $\boldsymbol{t}$):

$$(\boldsymbol{R_P})_{2,2} = r_{2,2}(\boldsymbol{t}_2^2 - \boldsymbol{t}_1^2 - \boldsymbol{t}_3^2) + 2r_{1,2}\boldsymbol{t}_1\boldsymbol{t}_2 + 2r_{3,2}\boldsymbol{t}_2\boldsymbol{t}_3. \tag{21}$$

We can re-write this expression in matricial form by defining the matrix $\boldsymbol{B}$ so

$$(\boldsymbol{R_P})_{2,2} = \boldsymbol{t}^t \underbrace{\begin{pmatrix} -r_{2,2} & r_{1,2} & 0 \\ r_{1,2} & r_{2,2} & r_{3,2} \\ 0 & r_{3,2} & -r_{2,2} \end{pmatrix}}_{\boldsymbol{B}} \boldsymbol{t}. \tag{22}$$

Matrix $\boldsymbol{B}$ has as eigenvalues $\{-1; +1; -r_{2,2}\}$. Since we assume that $\boldsymbol{R}$ is *not* a Y-rotation, then $r_{2,2} \neq 1$. See that if $r_{2,2} = 1$, then eq. (19) holds since the eigenvector is $[0, 0, r_{2,2}]^T$. In this case, the matrix $\boldsymbol{K_R}$ is the zero matrix by construction. We are interested in the other cases. Of interest for us is the eigenvector associated with the eigenvalue +1. Up to scale, this eigenvector has the form

$$\boldsymbol{u} = \begin{pmatrix} r_{1,2} \\ r_{2,2} + 1 \\ r_{3,2} \end{pmatrix}. \tag{23}$$

This vector is, in fact, the right nullspace of $\boldsymbol{K_R}$ for the considered $\boldsymbol{R}$. To prove this, consider each of the three entries of the expression $\boldsymbol{K_R u} = [(\boldsymbol{K_R u})_1, (\boldsymbol{K_R u})_2, (\boldsymbol{K_R u})_3]^T$. The first entry has the form:

$$(\boldsymbol{K_R u})_1 = -r_{3,2} r_{1,2} + r_{1,2} r_{3,2} = 0 \tag{24}$$

For the second and third entries, it is necessary to remind that any rotation matrix its adjugate agrees with its transpose, $\text{Adj}\,(\boldsymbol{R}) = \boldsymbol{R}^T$. Out of the nine identities that follow from this equivalence, we will leverage the next four:

$$r_{1,1} = r_{2,2} r_{3,3} - r_{2,3} r_{3,2} \tag{25}$$

$$r_{3,3} = r_{1,1} r_{2,2} - r_{1,2} r_{2,1} \tag{26}$$

$$r_{1,3} = r_{2,1} r_{3,2} - r_{2,2} r_{3,1} \tag{27}$$

$$r_{3,1} = r_{1,2} r_{2,3} - r_{1,3} r_{2,2} \tag{28}$$

With these expressions at hand, we can now derived the last two entries of $\boldsymbol{K_R u}$.

$$(\boldsymbol{K_R u})_2 = \overbrace{r_{3,3} + r_{2,1} r_{1,2} - r_{1,1} r_{2,2}}^{\text{Eq. (26) } 0} + \overbrace{r_{2,2} r_{3,3} - r_{2,3} r_{3,2} - r_{1,1}}^{\text{Eq. (25) } 0} = 0 \tag{29}$$

where we substituted the expressions in (25) and (26). For the last entry, we have that:

$$(\boldsymbol{K_R u})_3 = \overbrace{r_{3,1} - r_{2,3} r_{1,2} + r_{2,2} r_{1,3}}^{\text{Eq. (28) } 0} + \overbrace{r_{1,3} - r_{2,1} r_{3,2} + r_{2,2} r_{3,1}}^{\text{Eq. (27) } 0} = 0 \tag{30}$$

where we substituted the expressions in (27) and (28).

Recall that we are treating the case where the 3×3 matrix $\boldsymbol{K_R}$ has rank 2, and hence its nullspace is one-dimensional, which we associated with the translation vector $\boldsymbol{t}$. Form the above development we see that the vector $\boldsymbol{u}$ with form in (23) defines this right nullspace, that is, $\boldsymbol{t} \sim \boldsymbol{u}$. Hence, the pair $(\boldsymbol{R}, \boldsymbol{u})$ fulfills the constraints for $\boldsymbol{E}$ to have the pattern in (4). Notice, however, that the vector $\boldsymbol{u}$ was the eigenvector associated with the eigenvalue +1 of $\boldsymbol{B}$. Re-scaling the vector $\boldsymbol{u}$ to have norm one, we have that $\boldsymbol{u}^T \boldsymbol{B u} = +1$. Since this expressions is the matricial form of the entry $(\boldsymbol{R_P})_{2,2}$ of the matrix $\boldsymbol{R_P}$, we have that $(\boldsymbol{R_P})_{2,2} = 1$. Note that $\boldsymbol{R_P}$ is a rotation matrix by construction; in order to fulfill the orthogonal constraints with $(\boldsymbol{R_P})_{2,2} = 1$, the next entries must be equal to zero: $(\boldsymbol{R_P})_{1,2} = (\boldsymbol{R_P})_{2,1} = (\boldsymbol{R_P})_{2,3} = (\boldsymbol{R_P})_{3,2} = 0$, that is, $\boldsymbol{R_P}$ has the general form

$$\boldsymbol{R_P} = \begin{pmatrix} a & 0 & b \\ 0 & 1 & 0 \\ c & 0 & d \end{pmatrix} \tag{31}$$

for the four scalars a, b, c, d.

Since $\boldsymbol{t}$ has norm one, we have that $\boldsymbol{P}^T = \boldsymbol{P}$ and so $\boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{P}\boldsymbol{P}^T = \boldsymbol{I}_3$ with $\det(\boldsymbol{P}) = +1$. Therefore, for any rotation $\boldsymbol{R}$ we have that $\boldsymbol{R}_{\boldsymbol{P}}^T\boldsymbol{R}_{\boldsymbol{P}} = \boldsymbol{R}_{\boldsymbol{P}}\boldsymbol{R}_{\boldsymbol{P}}^T = \boldsymbol{I}_3$ with $\det(\boldsymbol{R}_{\boldsymbol{P}}) = +1$, hence being a rotation matrix, *i.e.* $\boldsymbol{R}_{\boldsymbol{P}}$ is a Y-rotation.

For any rotation $\boldsymbol{R}$, the matrix $\boldsymbol{K}_{\boldsymbol{R}}$ evaluated at that rotation is rank deficient (with rank generally 2). Its right nullspace is given by $\boldsymbol{t}$. We can form the rotation $\boldsymbol{R}_{\boldsymbol{P}} = \boldsymbol{P}\boldsymbol{R}$, with $\boldsymbol{P} = 2\boldsymbol{t}\boldsymbol{t}^T - \boldsymbol{I}_3$ (the reflection w.r.t. $\boldsymbol{t}$). The rotation matrix $\boldsymbol{R}_{\boldsymbol{P}}$ is a Y-rotation by construction. Since $\boldsymbol{R}_{\boldsymbol{P}}$ and $\boldsymbol{R}$ are the two rotations derived from the same essential matrix $\boldsymbol{E}$ for a given $\boldsymbol{t}$, it follows that for any essential matrix $\boldsymbol{E}$ that fulfills the constraints we can always derive a factorization with a Y-rotation.

To wrap-up this section, we have proven that any essential matrix that fulfills the set of constraints given by the pattern in (4) has a factorization in which one of the rotation matrices is a Y-rotation. Further, for a given Y-rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$, the "reflected" rotation $\boldsymbol{R}_{\boldsymbol{P}} = \boldsymbol{P}\boldsymbol{R}$ with $\boldsymbol{P} = 2\boldsymbol{t}\boldsymbol{t}^T - \boldsymbol{I}_3$ also fulfills the constraints, as expected. The 3×3 matrix $\boldsymbol{K}_{\boldsymbol{R}}$ evaluated at $\boldsymbol{R}$ is the zero matrix, while evaluated at $\boldsymbol{R}_{\boldsymbol{P}}$ the matrix $\boldsymbol{K}_{\boldsymbol{R}}$ has rank 2, and its right nullspace is given by $\boldsymbol{t}$.

B. Problem (R-F) is convex

In this section we show that problem (R-F) is a convex problem, *i.e.* it minimizes a convex cost function over a convex set. First, since the feasible set is the Cartesian product of two convex sets (the Euclidean space and the cone of positive semidefinite matrices), then it is also convex [34, Sec. 2]. Next we prove that the cost function is also convex. Recall that the objective takes the form:

$$f_{\mathrm{S}}^{\star} = \min_{\boldsymbol{S}\in\mathbb{R}^{n\times n}} \underbrace{\|\boldsymbol{S}\boldsymbol{x}\|_{\mathrm{F}}^2}_{f_1} + \underbrace{\left\|\boldsymbol{S} - \boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i\right\|_{\mathrm{F}}^2}_{f_2}. \tag{32}$$

Let $\boldsymbol{s} \doteq \mathrm{vec}(\boldsymbol{S}) \in \mathbb{R}^{n^2}$ and the vector with all the unknowns $\boldsymbol{y} \doteq [\boldsymbol{s}, \lambda_1, \ldots, \lambda_m]^T \in \mathbb{R}^{n^n+m}$ and so

$$f_1 = \|\boldsymbol{S}\boldsymbol{x}\|_{\mathrm{F}}^2 = \left\|(\boldsymbol{x}^T \otimes \boldsymbol{I}_n)\boldsymbol{s}\right\|_2^2 = \left\|\tilde{\boldsymbol{X}}\boldsymbol{y}\right\|_2^2 \tag{33}$$

with the matrix $\tilde{\boldsymbol{X}} \in \mathbb{R}^{n^2+m}$ only has non-zero values in its top-left $n \times n^2$ entries with the block $(\boldsymbol{x}^T \otimes \boldsymbol{I}_n) \in \mathbb{R}^{n\times n^2}$.

The second term in the objective is written in a similar way as:

$$f_2 = \left\|\boldsymbol{S} - \boldsymbol{C} + \sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i\right\|_{\mathrm{F}}^2 = \left\|\boldsymbol{s} - \boldsymbol{c} + \sum_{i=1}^{m} \lambda_i \boldsymbol{a}_i\right\|_2^2 = \left\|\tilde{\boldsymbol{A}}\boldsymbol{y} - \boldsymbol{c}\right\|_2^2 \tag{34}$$

with $\tilde{\boldsymbol{A}} \in \mathbb{R}^{n^2\times(n^2+m)}$

$$\tilde{\boldsymbol{A}} = \begin{pmatrix} \boldsymbol{I}_{n^2} & \boldsymbol{a}_1 & \ldots & \boldsymbol{a}_m \end{pmatrix}. \tag{35}$$

Functions f_1 and f_2 are convex since their Hessians $\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}}$, respectively, are always positive semidefinite independently of the matrices $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{A}}$. The full cost is the sum of them and thus it is also convex [34, Sec. 3.2], proving the claim.

C. Line search algorithm for extended point with lower cost

This section explains the line search algorithm employed to obtain a solution $\boldsymbol{Y}_{j+1}$ with rank $k+1$ from a solution $\boldsymbol{Y}_j$ with rank k for the iterative certifier. The algorithm follows the same idea than the Riemmanian staircase [36]: use the Hessian of the Lagrangian to find a search direction for the next solution. In our proposal it is the rank of the matrix $\boldsymbol{Y}$ (associated with the Hessian $\boldsymbol{S}$ of the original problem) the one which is extended. The extension does not affect the lagrange multipliers and thus we are interested only on the contribution of the matrix $\boldsymbol{Y}$. As in [36], the naive extension with a zero column $\boldsymbol{Y}_+ \doteq [\boldsymbol{Y}, \boldsymbol{0}_{n\times 1}]$ leads to rank-deficient solution, hence triggering the stop condition, since the extended $\boldsymbol{Y}$ is a stationary point of rank k. Nevertheless, the Hessian of the problem evaluated at this point $\boldsymbol{Y}_+$ gives us a search direction for the next $k+1$ rank solution.

Let us define the direction $\dot{\boldsymbol{Y}} = [\boldsymbol{0}_{n\times k} \,|\boldsymbol{v}] \in \mathbb{R}^{n\times(k+1)}$, where $\boldsymbol{v}$ is the search direction that we want to computed from the Hessian. The Hessian for this problem $\mathcal{H}_{\mathrm{F}} \in \mathbb{R}^{(nk+m)\times(nk+m)}$ evaluated at a generic point $\boldsymbol{Y}\times\boldsymbol{\lambda} \in \mathbb{R}^{n\times k}\times\mathbb{R}^m$ can be written in terms of three blocks $\mathcal{H}_{\mathrm{YY}} \in \mathbb{R}^{nk\times nk}, \mathcal{H}_{\lambda\lambda} \in \mathbb{R}^{m\times m}, \mathcal{H}_{\mathrm{Y}\lambda} \in \mathbb{R}^{nk\times m}$ (these blocks depend on the point but we omit this dependency for clarity). The full Hessian is expressed by

$$\mathcal{H}_{\mathrm{F}} \doteq \begin{pmatrix} \mathcal{H}_{\mathrm{YY}} & \mathcal{H}_{\mathrm{Y}\lambda} \\ \mathcal{H}_{\mathrm{Y}\lambda}^T & \mathcal{H}_{\lambda\lambda} \end{pmatrix}. \tag{36}$$

Thus, the Hessian vector product from section D on the direction $[\boldsymbol{V}_{\boldsymbol{Y}}, \boldsymbol{V}_{\lambda}]$ is $\nabla^2 f_{\mathrm{F}}[\boldsymbol{V}_{\boldsymbol{Y}}, \boldsymbol{V}_{\lambda}] = \mathcal{H}_{\mathrm{F}}[\boldsymbol{V}_{\boldsymbol{Y}}, \boldsymbol{V}_{\lambda}] \in \mathbb{R}^{nk+m}$. For the extended version we have that $\boldsymbol{V}_{\lambda} = \boldsymbol{0}_{m\times 1}$ and $\boldsymbol{V}_{\boldsymbol{Y}} = \mathrm{vec}(\dot{\boldsymbol{Y}})$. Since we need to compute a vector that makes it negative, we study the expression $[\boldsymbol{V}_{\boldsymbol{Y}}, \boldsymbol{V}_{\lambda}]^T \nabla^2 f_{\mathrm{F}}[\boldsymbol{V}_{\boldsymbol{Y}}, \boldsymbol{V}_{\lambda}] = \boldsymbol{V}_{\boldsymbol{Y}}^T \mathcal{H}_{\mathrm{YY}} \boldsymbol{V}_{\boldsymbol{Y}}$. Therefore, we are only interested on the top-left block, $\mathcal{H}_{\mathrm{YY}}$. Further, since $\dot{\boldsymbol{Y}}$ is all zeros except its last column, its vectorization has the form $\boldsymbol{V}_{\boldsymbol{Y}} = \mathrm{vec}(\dot{\boldsymbol{Y}}) = [\underbrace{0,\ldots,0}_{nk}, \boldsymbol{v}]$ and thus, from $\mathcal{H}_{\mathrm{YY}}$ we will need to compute only the bottom-right $n\times n$ block. Through a symbolic computation we obtain that this block is given by $2\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\boldsymbol{x}^T + 2\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{Y}^T + 4(\boldsymbol{Y}\boldsymbol{Y}^T - \boldsymbol{C} + \sum_{i=1}^m \lambda_i \boldsymbol{A}_i)$. The eigenvector associated with the smallest eigenvalue of this block is considered as the search direction $\boldsymbol{v}$.

D. On-manifold iterative certifier

In this section we include the information regarding the iterative certifier of section 4: the operators associated with the manifold and the quadratic model.

D.1. Domain

This problem optimizes over the product $\mathbb{R}^m \times \mathbb{S}^n_+$. Since we are considering the Cartesian product, we can tackle each component separately in terms of operators. There are four operators that have to be defined: retraction, projection from tangent space, gradient and Hessian. These functions are independent of the cost. Roughly these operators allow to pass from the Euclidean space where all the more consuming operations are computed to the Riemmanian space. Since the Lagrange multipliers

are only real, the operators associated with them are the identity functions. For the PSD matrix, these operators depend on the geometric definition of the space. In this work, we rely on the low-rank factorization s $\boldsymbol{S} = \boldsymbol{Y}\boldsymbol{Y}^T$, with $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$ full rank. The optimization can be performed in the $n \times k$-Euclidean space. However, under this decomposition of $\boldsymbol{S}$ we can derive two different, full rank matrices $\boldsymbol{Y}, \boldsymbol{W}$ that originates the same PSD matrix $\boldsymbol{S}$. These two matrices are related by a $k \times k$ matrix $\boldsymbol{O}$ which is orthogonal, $\boldsymbol{Y} \sim \boldsymbol{W}\boldsymbol{O}$. Other matrices $\boldsymbol{O}$ give different yet equivalent points $\boldsymbol{O}\boldsymbol{Y}$ from the point on view of the cost and the relation $\boldsymbol{S} = \boldsymbol{Y}\boldsymbol{Y}^T$. Hence, this equivalence has to be taken into account when optimizing any function, since equivalent points are considered as different by the naive version on only the Euclidean space. The operators reflect this characteristics.

We employ a projection-based retraction defined as

$$\mathrm{Retr}_{\boldsymbol{Y}}(\boldsymbol{U}) = \boldsymbol{Y} + \boldsymbol{U}, \tag{37}$$

where $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$ is a point on the manifold and $\boldsymbol{U}$ a point on the (horizontal) tangent space.

The horizontal tangent space projector is

$$\mathrm{P}_{\boldsymbol{Y}}(\boldsymbol{H}) = \boldsymbol{H} - \boldsymbol{Y}\Omega, \tag{38}$$

where $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$ is a point on the manifold, $\boldsymbol{H}$ a point on the ambient space, and Ω is a skew-symmetric matrix. Ω is the unique solution to the Sylvester equation $\Omega(\boldsymbol{Y}^T\boldsymbol{Y}) + (\boldsymbol{Y}^T\boldsymbol{Y})\Omega = \boldsymbol{Y}^T\boldsymbol{H} - \boldsymbol{H}^T\boldsymbol{Y}$.

The Riemannian gradient at a point $\boldsymbol{Y}$ is the Euclidean gradient of the function evaluated at the point. The Riemmanian Hessian is also the Euclidean Hessian and no further operation is required.

D.2. Euclidean quadratic model

In this last part we include the Euclidean quadratic model for the certifier.

Recall that the optimization problem has the form

$$f_{\mathrm{F}}^{\star} = \min_{\boldsymbol{Y}\in\mathbb{R}^{n\times k}, \boldsymbol{\lambda}\in\mathbb{R}^m} \left\|\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\right\|_{\mathrm{F}}^2 + \left\|\boldsymbol{Y}\boldsymbol{Y}^T - \boldsymbol{C} + \sum_{i=1}^{m}\lambda_i\boldsymbol{A}_i\right\|_{\mathrm{F}}^2 \tag{39}$$

The cost is quartic in $\boldsymbol{Y} \in \mathbb{R}^{n\times k}$ and quadratic in $\boldsymbol{\lambda}$ as

$$f_{\mathrm{F}} = \boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x} + \mathrm{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{Y}\boldsymbol{Y}^T) + \mathrm{tr}\left((\boldsymbol{C} - \sum_{i=1}^{m}\lambda_i\boldsymbol{A}_i)^T(\boldsymbol{C} - \sum_{i=1}^{m}\lambda_i\boldsymbol{A}_i)\right) - 2\,\mathrm{tr}\left(\boldsymbol{Y}\boldsymbol{Y}^T(\boldsymbol{C} - \sum_{i=1}^{m}\lambda_i\boldsymbol{A}_i)\right). \tag{40}$$

For the gradient and Hessian it is useful to re-write the cost f_{F} highlighting one

lagrange multiplier. Hence, we have that:

$$f_{\mathrm{F}} = \operatorname{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{Y}^T) + \operatorname{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{Y}\boldsymbol{Y}^T) + \tag{41}$$

$$+ \lambda_i^2 \operatorname{tr}(\boldsymbol{A}_i^T\boldsymbol{A}_i) - 2\lambda_i \operatorname{tr}\left(\boldsymbol{A}_i(\boldsymbol{C} - \sum_{j=1, j\neq i}^{m} \lambda_j \boldsymbol{A}_j)\right) +$$

$$+ \operatorname{tr}\left((\boldsymbol{C} - \sum_{j=1, j\neq i}^{m} \lambda_j \boldsymbol{A}_j)^T(\boldsymbol{C} - \sum_{j=1, j\neq i}^{m} \lambda_j \boldsymbol{A}_j)\right) +$$

$$+ 2\lambda_i \operatorname{tr}\left(\boldsymbol{A}_i^T\boldsymbol{Y}\boldsymbol{Y}^T\right)$$

$$- 2\operatorname{tr}\left((\boldsymbol{C} - \sum_{j=1, j\neq i}^{m} \lambda_j \boldsymbol{A}_j)\boldsymbol{Y}\boldsymbol{Y}^T\right). \tag{42}$$

The Euclidean gradient $\nabla f_{\mathrm{F}} = (\nabla f_{\mathrm{F}\boldsymbol{\lambda}}, \nabla f_{\mathrm{F}\boldsymbol{Y}})$ has the elements

$$\nabla f_{\mathrm{F}\boldsymbol{\lambda}} = [\nabla f_{\mathrm{F}\lambda 1}, \ldots, \nabla f_{\mathrm{F}\lambda m}]^T \in \mathbb{R}^{m\times 1}, \tag{43}$$

with

$$\nabla f_{\mathrm{F}\lambda i} = 2\lambda_i \operatorname{tr}(\boldsymbol{A}_i^T\boldsymbol{A}_i) + 2\operatorname{tr}\left(\boldsymbol{A}_i^T(\boldsymbol{Y}\boldsymbol{Y}^T - \boldsymbol{C} + \sum_{j=1, j\neq i}^{m} \lambda_j \boldsymbol{A}_j)\right) \tag{44}$$

For the $\boldsymbol{Y}$ part we will need some basic notions of multivariate calculus. Let us define the generic expression $g(\boldsymbol{Y}) = \operatorname{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}\boldsymbol{Y}^T)$. We can obtain a second-order approximation by evaluating $\boldsymbol{Y} + \boldsymbol{H}$ and discarding higher order terms as

$$g(\boldsymbol{Y} + \boldsymbol{H}) = \operatorname{tr}\left((\boldsymbol{Y} + \boldsymbol{H})(\boldsymbol{Y} + \boldsymbol{H})^T\boldsymbol{A}(\boldsymbol{Y} + \boldsymbol{H})(\boldsymbol{Y} + \boldsymbol{H})^T\right) = \tag{45}$$

$$= \operatorname{tr}\left((\boldsymbol{Y}\boldsymbol{Y}^T + 2\boldsymbol{Y}\boldsymbol{H}^T + \boldsymbol{H}\boldsymbol{H}^T)\boldsymbol{A}(\boldsymbol{Y}\boldsymbol{Y}^T + 2\boldsymbol{Y}\boldsymbol{H}^T + \boldsymbol{H}\boldsymbol{H}^T)\right) =$$

$$= \underbrace{\operatorname{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}\boldsymbol{Y}^T)}_{g(\boldsymbol{Y}+\boldsymbol{H})} +$$

$$+ \underbrace{2\operatorname{tr}(\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}\boldsymbol{H}^T) + 2\operatorname{tr}(\boldsymbol{A}\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{Y}\boldsymbol{H}^T)}_{\langle \Delta g(\boldsymbol{Y}), \boldsymbol{H}\rangle}$$

$$+ \underbrace{2\operatorname{tr}(\boldsymbol{Y}\boldsymbol{H}^T\boldsymbol{A}\boldsymbol{Y}\boldsymbol{H}^T) + 2\operatorname{tr}(\boldsymbol{A}\boldsymbol{Y}\boldsymbol{H}^T\boldsymbol{Y}\boldsymbol{H}^T) + 4\operatorname{tr}(\boldsymbol{H}\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}\boldsymbol{H}^T)}_{\langle \Delta^2 g(\boldsymbol{Y})\boldsymbol{H}, \boldsymbol{H}\rangle}. \tag{46}$$

Hence, taking into account that $\boldsymbol{A} = (\boldsymbol{x}\boldsymbol{x}^T + \boldsymbol{I}_{n\times n})$ we have that the gradient is

$$\nabla f_{\mathrm{F}\boldsymbol{Y}} = 2\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y} + 2\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{Y} + 4(\boldsymbol{Y}\boldsymbol{Y}^T)^T\boldsymbol{Y} + 4(\sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i - \boldsymbol{C})^T\boldsymbol{Y} \in \mathbb{R}^{n\times k} \tag{47}$$

The vector-product Hessian is given in a similar form as $\nabla^2 f_{\mathrm{F}}[\boldsymbol{V}_\lambda, \boldsymbol{V}_{\boldsymbol{Y}}] = (\nabla^2 f_{\mathrm{F}\boldsymbol{\lambda}}[\boldsymbol{V}_\lambda, \boldsymbol{V}_{\boldsymbol{Y}}], \nabla^2 f_{\mathrm{F}\boldsymbol{Y}}[\boldsymbol{V}_\lambda, \boldsymbol{V}_{\boldsymbol{Y}}])$ with

$$\nabla^2 f_{\mathrm{F}\boldsymbol{\lambda}}[\boldsymbol{V}_\lambda, \boldsymbol{V}_{\boldsymbol{Y}}] = [\nabla^2 f_{\mathrm{F}\lambda 1}, \ldots, \nabla^2 f_{\mathrm{F}\lambda m}] \in \mathbb{R}^{m\times 1}, \tag{48}$$

where

$$\nabla^2 f_{\mathrm{F}\lambda i} = 4\operatorname{tr}(\boldsymbol{A}_i\boldsymbol{Y}\boldsymbol{V}_{\boldsymbol{Y}}^T) + 2\sum_{j=1}^{m} \boldsymbol{V}_{\lambda j} \operatorname{tr}(\boldsymbol{A}_j^T\boldsymbol{A}_i). \tag{49}$$

For the element corresponding with the $\boldsymbol{Y}$ we have that

$$\nabla^2 f_{\mathrm{F}\boldsymbol{Y}}[\boldsymbol{V}_\lambda, \boldsymbol{V}_Y] = 4\boldsymbol{Y}\boldsymbol{V}_Y^T\boldsymbol{Y} + 4\boldsymbol{V}_Y\boldsymbol{Y}^T\boldsymbol{Y} + 4\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{V}_Y + \tag{50}$$

$$+ 4(\sum_{i=1}^{m} \lambda_i \boldsymbol{A}_i - \boldsymbol{C})^T \boldsymbol{V}_Y +$$

$$+ 2\boldsymbol{V}_Y\boldsymbol{Y}^T\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y} + 2\boldsymbol{Y}\boldsymbol{V}_Y{}^T\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y} + 2\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{V}_Y +$$

$$+ 2\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{V}_Y\boldsymbol{Y}^T\boldsymbol{Y} + 2\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{V}_Y^T\boldsymbol{Y} + 2\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{V}_Y +$$

$$+ 4\sum_{i=1}^{m} \boldsymbol{V}_{\lambda_i}\boldsymbol{A}_i\boldsymbol{Y} \tag{51}$$

E. On-manifold estimation of essential matrix

In this section we include the necessary information for the on-manifold estimation of the essential matrix: the required operators associated with the geometry of the manifold and the quadratic models.

E.1. Domain

As mentioned in the main manuscript, for the optimization we define the set as the product of manifolds $\mathcal{S}^2 \times \mathrm{SO}(2)$. Since we are considering the product, we can tackle each component separately in terms of operators. There are four operators that have to be defined are independent of the problem: retraction, projection from tangent space, gradient and Hessian. Roughly these operators allow to pass from the Euclidean space, where all the more consuming operations are computed, to the Riemmanian space.

Sphere We employ a projection-based retraction defined as:

$$\mathrm{Retr}_{\boldsymbol{t}}(\boldsymbol{u}) = \frac{\boldsymbol{t} + \boldsymbol{u}}{\|\boldsymbol{t} + \boldsymbol{u}\|} \tag{52}$$

where $\boldsymbol{t}$ is a point on the manifold and $\boldsymbol{u}$ is a point of the tangent space.

The tangent space projector is

$$\mathrm{P}_{\boldsymbol{t}}(\boldsymbol{u}) = \boldsymbol{u} - \boldsymbol{t}^T\boldsymbol{u}\boldsymbol{t} \tag{53}$$

with $\boldsymbol{t}$ is a point on the manifold and $\boldsymbol{u}$ a vector in the ambient space.

The Riemmanian gradient at a point $\boldsymbol{t}$ is obtained by projecting the Euclidean gradient $\nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t})$

$$\mathrm{grad}\ \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t}) = \mathrm{P}_{\boldsymbol{t}}(\nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t})). \tag{54}$$

Last, the Riemmanian Hessian depends on the point $\boldsymbol{t}$, the Euclidean gradient $\nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t})$ and the Euclidean vector-product Hessian $\nabla^2 f_{\boldsymbol{t}}(\boldsymbol{t})[\boldsymbol{u}]$. It is computed as

$$\mathrm{Hess}\ \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t})[\boldsymbol{u}] = \mathrm{P}_{\boldsymbol{t}}(\nabla^2 f_{\boldsymbol{t}}(\boldsymbol{t})[\boldsymbol{u}]) - \boldsymbol{t}^T \nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t})\boldsymbol{u} \tag{55}$$

Rotation The projection-like retraction is given by:

$$\mathrm{Retr}_{\boldsymbol{R}}(\boldsymbol{Y}) = \boldsymbol{U}\boldsymbol{V}^T, \quad \boldsymbol{R} + \boldsymbol{Y} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T \tag{56}$$

where $\boldsymbol{R}$ is a point on the manifold and $\boldsymbol{Y}$ is a point of the tangent space. To guarantee that the solution returned is a rotation matrix, we check the determinant of $\boldsymbol{U}\boldsymbol{V}^T$. If it is negative, we multiply the last column of $\boldsymbol{U}$ by -1.

The tangent space projector is

$$\mathrm{P}_{\boldsymbol{R}}(\boldsymbol{Y}) = \boldsymbol{Y} - \boldsymbol{R}\mathrm{symm}(\boldsymbol{R}^T\boldsymbol{Y}) \tag{57}$$

where symm($\bullet$) takes the symmetric part of the argument, $\boldsymbol{R}$ is a rotation matrix and $\boldsymbol{Y}$ is a matrix on the ambient space.

The Riemmanian gradient at a point $\boldsymbol{R}$ is obtained by projecting the Euclidean gradient $\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R})$

$$\mathrm{grad}\ \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R}) = \mathrm{P}_{\boldsymbol{R}}(\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R})). \tag{58}$$

Last, the Riemmanian Hessian depends on the point $\boldsymbol{R}$, the Euclidean gradient $\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R})$ and the Euclidean vector-product Hessian $\nabla^2 f_{\boldsymbol{R}}(\boldsymbol{R})[\boldsymbol{Y}]$. It is computed as

$$\mathrm{Hess}\ \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R})[\boldsymbol{Y}] = \mathrm{P}_{\boldsymbol{R}}\Big(\nabla^2 f_{\boldsymbol{R}}(\boldsymbol{R})[\boldsymbol{Y}] - \boldsymbol{Y}\mathrm{symm}(\boldsymbol{R}^T\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R}))\Big) \tag{59}$$

E.2. Euclidean quadratic model

Last, we include here the Euclidean model. Recall that we consider the matrix $\boldsymbol{R}$ as the 3D extension of the 2D rotation which is the to-be-estimated variable. The vectors $\boldsymbol{t}$ is a real 3D-vector.

The cost can be written as [32]

$$\frac{1}{2}\boldsymbol{e}^T\boldsymbol{C}\boldsymbol{e} = \frac{1}{2}\boldsymbol{t}^T\boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{t} = \frac{1}{2}\boldsymbol{r}^T\boldsymbol{M}_{\boldsymbol{R}}\boldsymbol{r} \tag{60}$$

with $\boldsymbol{M}_{\boldsymbol{t}} = \sum_{i=1}^{N}(\boldsymbol{f}_i \times \boldsymbol{R}\boldsymbol{f}_i')(\boldsymbol{f}_i \times \boldsymbol{R}\boldsymbol{f}_i')^T = \tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{R}} \in \mathbb{S}_+^3$ and $\boldsymbol{M}_{\boldsymbol{R}} = \sum_{i=1}^{N}(\boldsymbol{f}_i' \otimes [\boldsymbol{f}_i]_{\mathrm{x}}\boldsymbol{t})(\boldsymbol{f}_i' \otimes [\boldsymbol{f}_i]_{\mathrm{x}}\boldsymbol{t})^T = \tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{T}} \in \mathbb{S}_+^9$, where $\tilde{\boldsymbol{R}} = (\boldsymbol{R}^T \otimes \boldsymbol{I}_3)\boldsymbol{B}$, $\boldsymbol{B} \in \mathbb{R}^{9\times 3}$ is the matrix that holds for $\mathrm{vec}([\boldsymbol{t}]_{\mathrm{x}}) = \boldsymbol{B}\boldsymbol{t}$ and $\tilde{\boldsymbol{T}} = \boldsymbol{I}_3 \otimes [\boldsymbol{t}]_{\mathrm{x}}$.

The Euclidean gradient $\nabla \mathrm{f}_{\boldsymbol{R},\boldsymbol{t}}(\boldsymbol{R},\boldsymbol{t}) = (\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R}), \nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t}))$ reads:

$$\nabla \mathrm{f}_{\boldsymbol{R}}(\boldsymbol{R}) = \boldsymbol{M}_{\boldsymbol{R}}\boldsymbol{r}, \quad \nabla \mathrm{f}_{\boldsymbol{t}}(\boldsymbol{t}) = \boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{t}. \tag{61}$$

The action of the Euclidean Hessian on the vector $(\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}})$ is calculated by applying multivariate calculus as

$$\nabla^2 f[\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}}] = (\nabla^2 f_{\boldsymbol{R}}(\boldsymbol{R})[\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}}], \nabla^2 f_{\boldsymbol{t}}(\boldsymbol{t})[\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}}]), \tag{62}$$

with

$$\nabla^2 f_{\boldsymbol{R}}(\boldsymbol{R})[\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}}] = \boldsymbol{M}_{\boldsymbol{R}}\boldsymbol{V}_{\boldsymbol{r}} + \boldsymbol{M}_{\boldsymbol{R},\boldsymbol{t}}^T\boldsymbol{V}_{\boldsymbol{t}}, \quad \nabla^2 f_{\boldsymbol{t}}(\boldsymbol{t})[\boldsymbol{V}_{\boldsymbol{R}}, \boldsymbol{V}_{\boldsymbol{t}}] = \boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{V}_{\boldsymbol{t}} + \boldsymbol{M}_{\boldsymbol{R},\boldsymbol{t}}\boldsymbol{V}_{\boldsymbol{R}} \tag{63}$$

where $\boldsymbol{M}_{\mathrm{t;r}}(\boldsymbol{R},\boldsymbol{t}) \doteq \boldsymbol{M}_{\mathrm{t;r}}(\boldsymbol{R},\boldsymbol{t}) \in \mathbb{R}^{9\times 3}$ is defined as follows.

Recall that the cost function in eq. (60) can be written as

$$\frac{1}{2}\boldsymbol{e}^T\boldsymbol{C}\boldsymbol{e} = \frac{1}{2}\boldsymbol{t}^T\tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{R}}\boldsymbol{t} = \frac{1}{2}\boldsymbol{r}^T\tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{T}}\boldsymbol{r} \tag{64}$$

and that

$$\boldsymbol{M}_{\boldsymbol{R},\boldsymbol{t}} = \frac{\partial(\partial f(\boldsymbol{R},\boldsymbol{t}))}{\partial \boldsymbol{r}\partial \boldsymbol{t}} = \frac{\partial(\boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{t})}{\partial \boldsymbol{r}} \in \mathbb{R}^{3\times 9}, \tag{65}$$

with

$$\boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{t} = \tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{R}}\boldsymbol{t} = \tilde{\boldsymbol{R}}^T\boldsymbol{C}\tilde{\boldsymbol{T}}\boldsymbol{r} \in \mathbb{R}^3. \tag{66}$$

Recall that $\tilde{\boldsymbol{R}}^T \in \mathbb{R}^{3\times 9}$ and let $\tilde{\boldsymbol{r}}_i \in \mathbb{R}^{1\times 9}$ be the i-th row of the matrix, *i.e.*,

$$\tilde{\boldsymbol{R}}^T = \begin{pmatrix} \tilde{\boldsymbol{r}}_1 \\ \tilde{\boldsymbol{r}}_2 \\ \tilde{\boldsymbol{r}}_3. \end{pmatrix} \tag{67}$$

See that we can express each row as $\tilde{\boldsymbol{r}}_i = \boldsymbol{r}^T\boldsymbol{B}_i$, where $\boldsymbol{B}_i \in \mathbb{R}^{9\times 9}$ are sparse matrices (shown in equation (69)). Then, the i-th row of $\boldsymbol{M}_{\boldsymbol{t}}\boldsymbol{t}$ is given by

$$\tilde{\boldsymbol{r}}_i\boldsymbol{C}\tilde{\boldsymbol{T}}\boldsymbol{r} = \boldsymbol{r}^T\boldsymbol{B}_i\boldsymbol{C}\tilde{\boldsymbol{T}}\boldsymbol{r}, \tag{68}$$

and its derivative w.r.t. $\boldsymbol{r}$ is $2\boldsymbol{B}_i\boldsymbol{C}\tilde{\boldsymbol{T}}\boldsymbol{r} \in \mathbb{R}^{9\times 1}$. The gradient $\boldsymbol{M}_{\boldsymbol{R},\boldsymbol{t}}$ is obtained by stacking the derivatives of each row.

$$\boldsymbol{B}_1 = \begin{pmatrix} 0&0&0&0&0&0&0&0&0\\ 0&0&-1&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&-1&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&-1\\ 0&0&0&0&0&0&0&1&0 \end{pmatrix}, \boldsymbol{B}_2 = \begin{pmatrix} 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ -1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&-1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&-1&0&0 \end{pmatrix}, \boldsymbol{B}_3 = \begin{pmatrix} 0&-1&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&-1&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&-1&0\\ 0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0 \end{pmatrix} \tag{69}$$

F. Further experiments on synthetic data for the certifier

Figure (7) shows the minimum eigenvalue for the solutions to problem (R-Y). We follow the same format than in the main paper, *i.e.*, first row has the problems with pure Y-rotation and the second row those with noisy Y-rotation. From left to right, we have problems with general, lateral and forward motion. We only plot the results for $N = 50$ correspondences, while the X-axis indicates the level of noise. Recall that the focal length is fixed to 512 pixels. Figure (8) shows the absolute costs for the same set of experiments than in fig. (5) with the same format. Last, we include the number of certified optimal solutions (cost of the certifier below the given threshold), normalized by the number of problem instances.

For the noiseless motions, BOTH and ADJ certify all solutions. LEFT and RIGHT only failed to certify solutions for problems with $N = 10, 15$ and large noise. Even for these cases, the per-unit values are above 0.9. The worst performance is seen with the

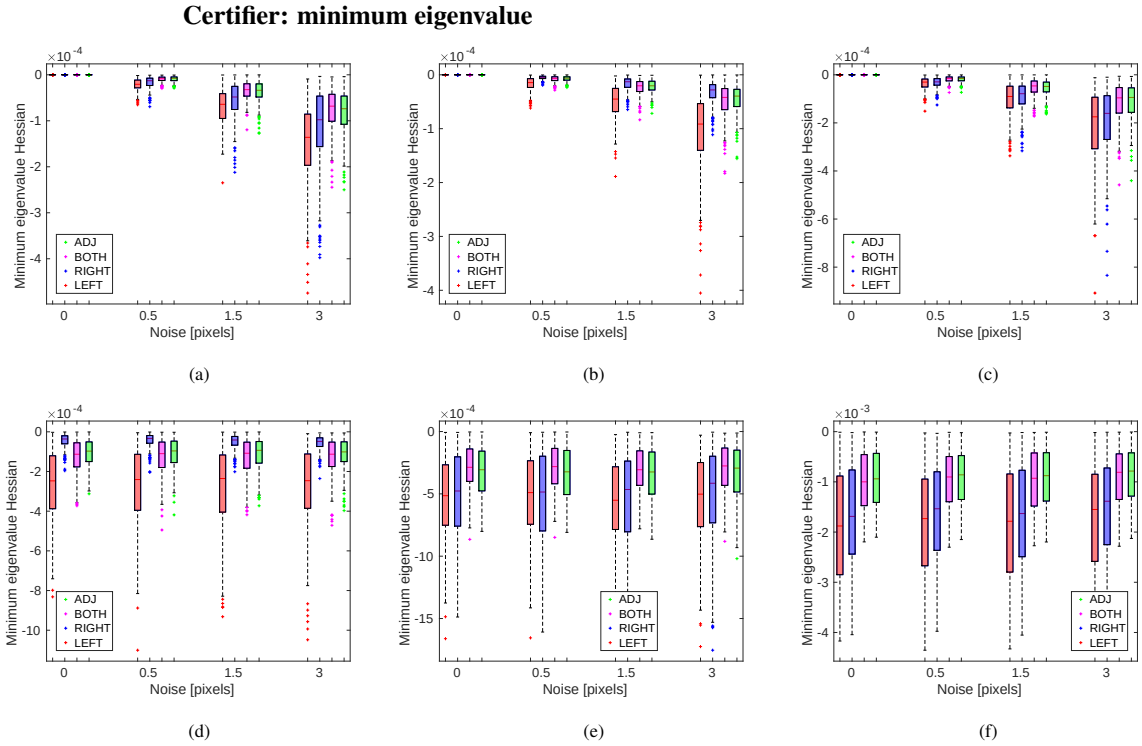


Figure 7: Minimum eigenvalue of the solution from the initialization problem (R-Y). Top row shows the experiments with pure Y-rotation and bottom row the experiments where a small perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis). The number of correspondences is fixed to $N = 50$.

forward motion. For noisy motions the tendency is similar, but we obtain more non-certified solutions. Recall that the certification depends on the provided solution and the data. Figure 9 shows the per-unit certified solutions for (from left to right): general, lateral and forward motions, for problem instances with $N = 50$ points and increasing level of noise (X-axis).

G. Further experiments on synthetic data for the on-manifold estimation

In this section we include the comparison between the different solvers employed to estimate the essential matrix.

First, we include in figure 10 the cost for the different solvers (including the minimal MIN-E) normalized by the one by OURS-E (Y-axis) for problem instances with $N = 15$ correspondences and noise level $\sigma = 0.5, 1.5, 3.0$ pixels (X-axis). We follow the same format that in the main document. The first row shows the errors for those cases with pure Y-rotation and the second those with noise Y-rotation. From left to right, we plot the configurations with general, forward and lateral translation. Notice that even for these small problem instances, MIN-E attains larger costs than the

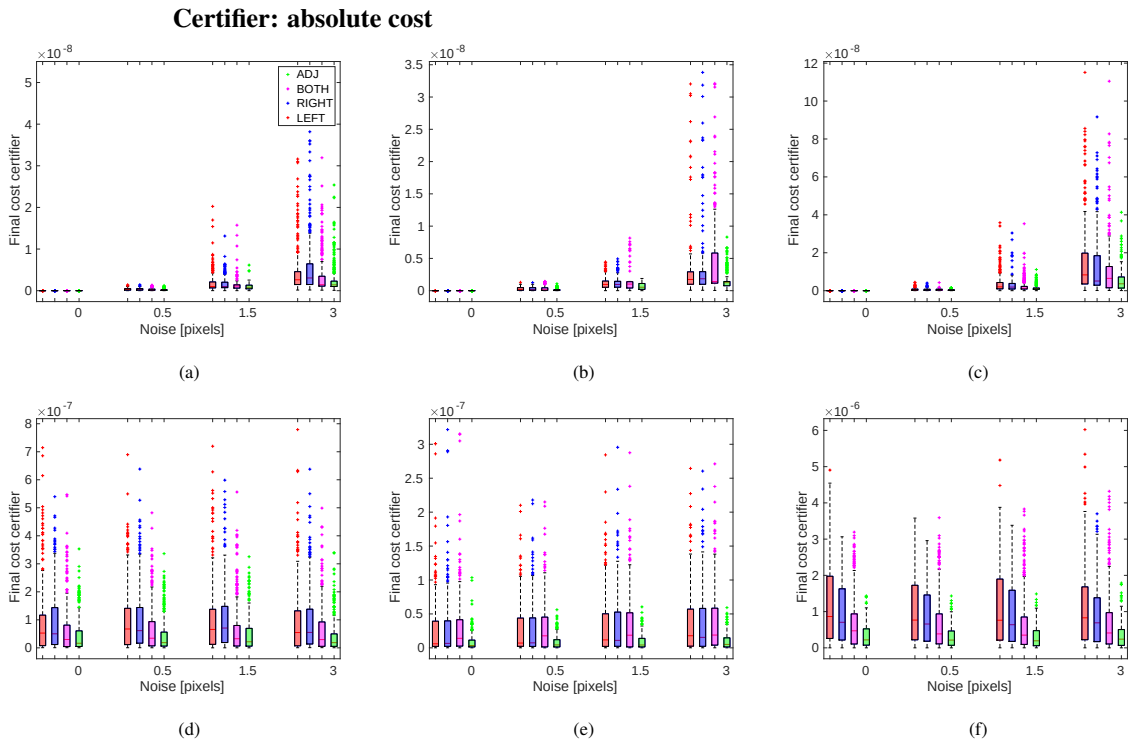


Figure 8: **Certifier for Essential matrix: absolute cost**: Absolute cost as in problem (R-Y) for the different formulations (see legend), level of noise (see X-axis) and the three considered configurations (from left to right): general, lateral and forward translation. First row shows the results for pure Y-rotation and the second row those for noisy Y-rotation. Notice the difference in the Y-scales.

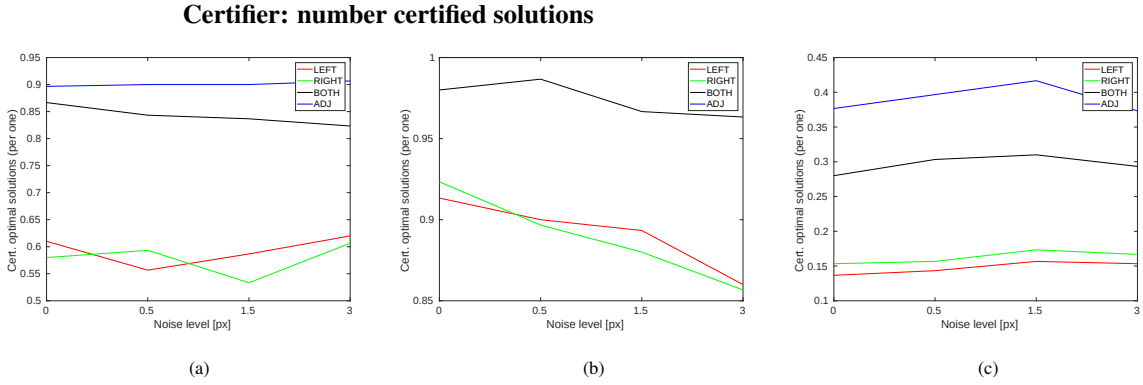


Figure 9: Number of certified solutions per-unit for noisy camera motions (left to right): general, lateral and forward translation, for $N = 50$ points and increasing level of noise.

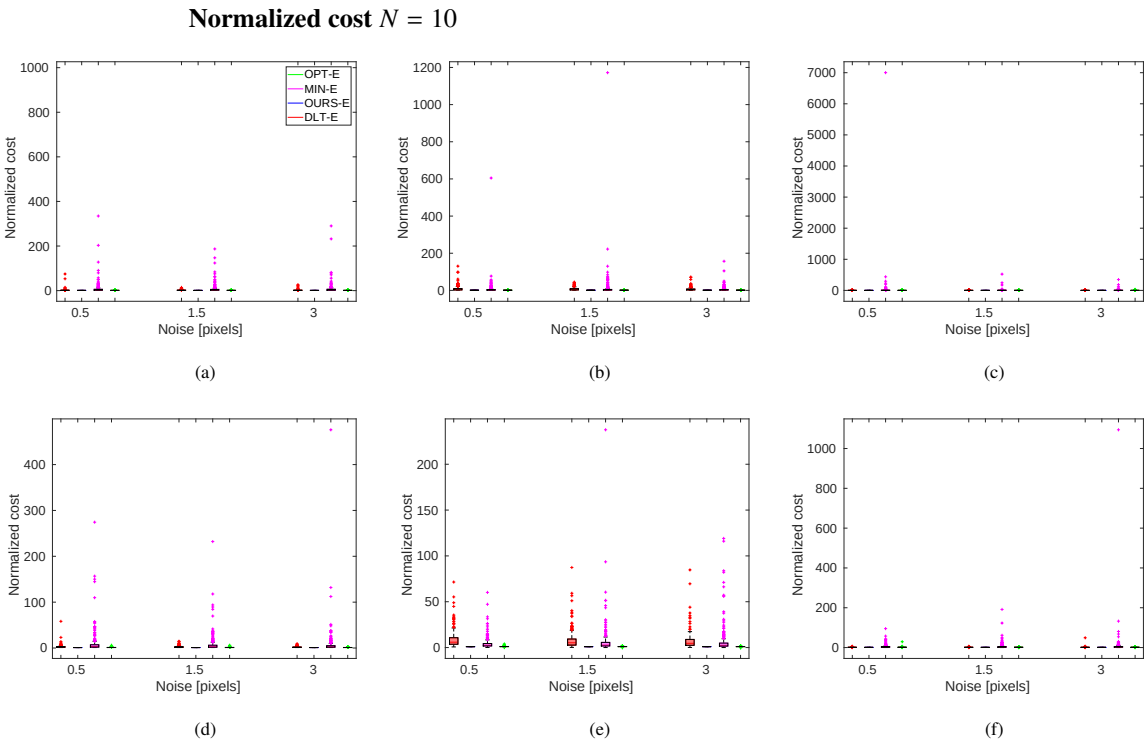


Figure 10: **Normalized cost** $N = 10$: Cost by the different solvers, including MIN-E, normalized by the cost attained by OURS-E for problem instances with $N = 10$. Values above one indicate costs larger than those of OURS-E. Top row shows the experiments with pure Y-rotation and bottom row the experiments where a small perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis).

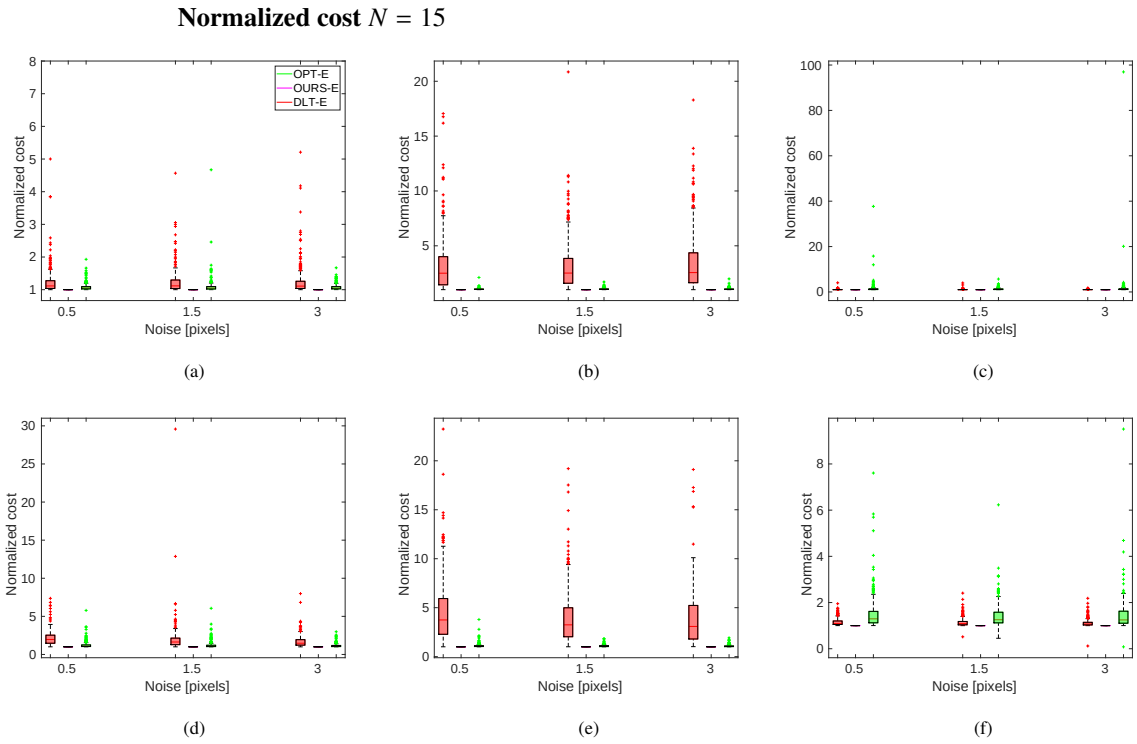


Figure 11: **Normalized cost** $N = 15$: Cost by the different solvers, including MIN-E, normalized by the cost attained by OURS-E for problem instances with $N = 15$. Values above one indicate costs larger than those of OURS-E. Top row shows the experiments with pure Y-rotation and bottom row the experiments where a small perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis).

rest of the solvers. This difference is more accentuated for problems with more correspondences, and thus we don't include the results for this solver for making the plots difficult to analyze. Notice, on the other hand, that DLT-E and OPT-E also obtain larger errors than OURS-E.

Figures 11 and 12 present the errors for problem instances with $N = 15, 100$ correspondences, respectively, for all the solvers except MIN-E. In general, as we indicate in the main manuscript, OPT-E returns solutions with larger errors despite being technically optimal, hence the necessity of the refinement proposed in this work. In some cases, the large cost by OPT-E shows numerical instabilities associated with the underlying tools. We must say, though, that while OURS-E obtains solutions with lower costs, for some problem instances (3 for $N = 10$ and 3 for $N = 15$ for forward movements with noisy rotations), our solver performs worse than OPT-E.

Further experiments on synthetic data We will now focus on problems with low number of correspondences. Our goal is to highlight the different performance of the state-of-the-art solvers and the one proposed in this work. For this set of experiments, we consider problem instances with number of correspondences from 8 to 12, noise

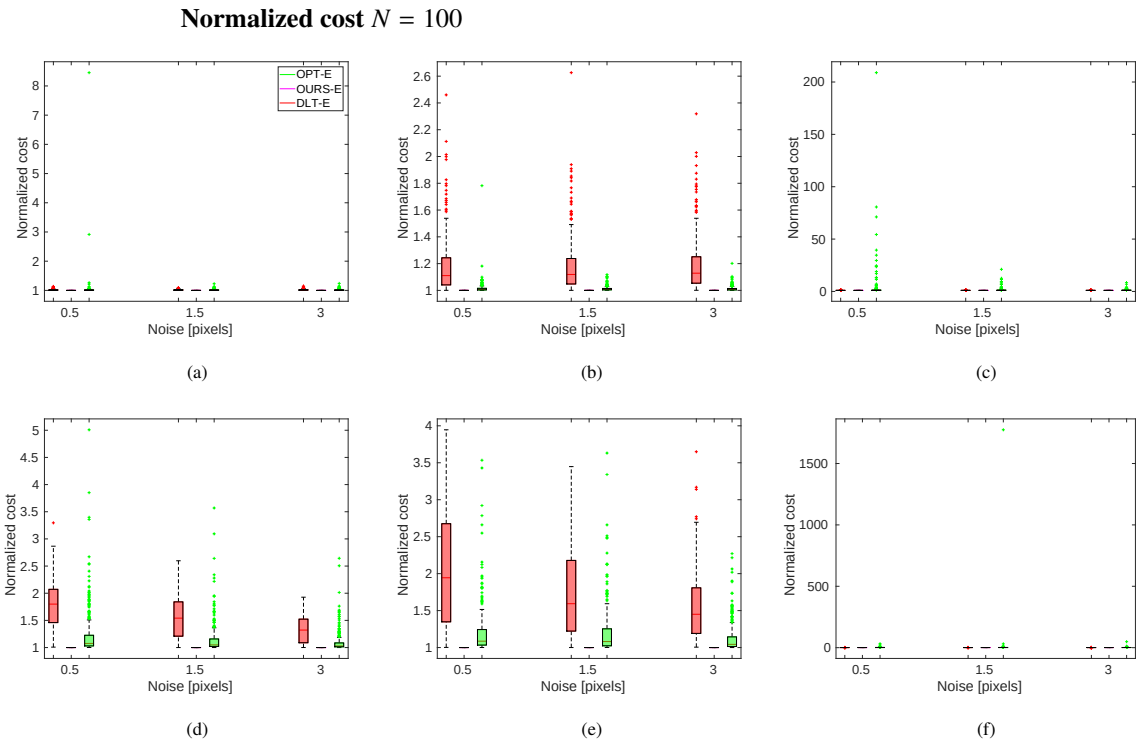


Figure 12: **Normalized cost** $N = 100$: Cost by the different solvers, including MIN-E, normalized by the cost attained by OURS-E for problem instances with $N = 100$. Values above one indicate costs larger than those of OURS-E. Top row shows the experiments with pure Y-rotation and bottom row the experiments where a small perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis).

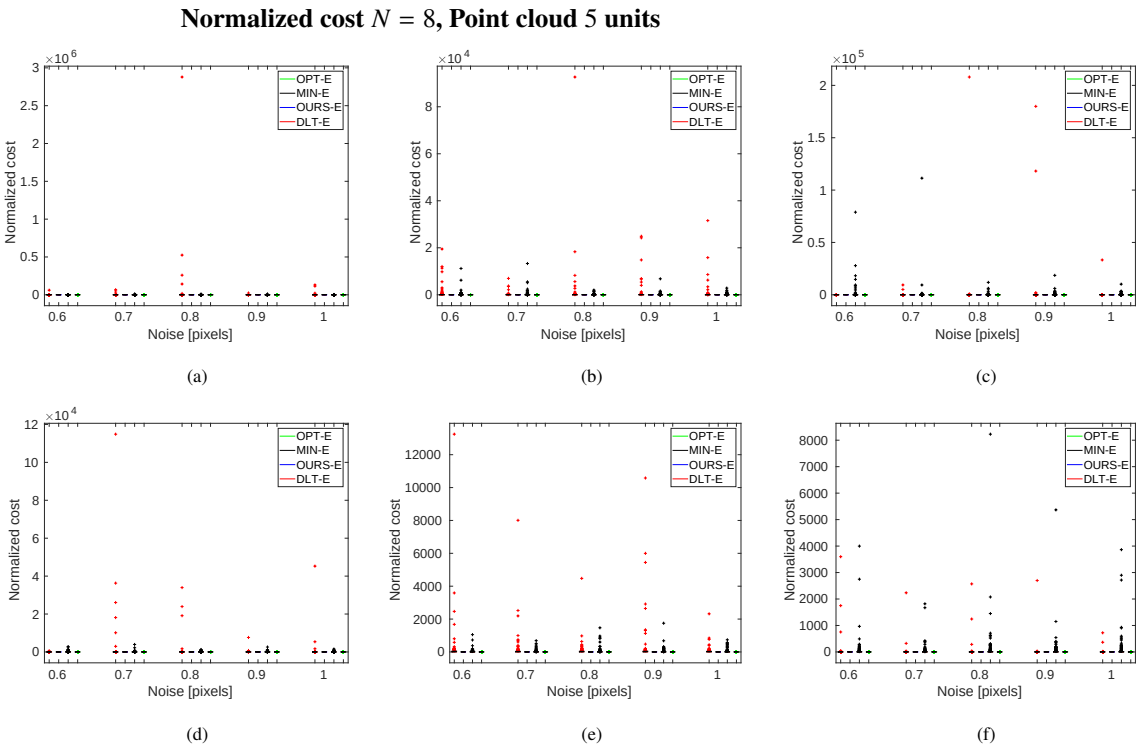


Figure 13: **Normalized cost $N = 8$, Point cloud 5 units**: Cost by the different solvers, including MIN-E, normalized by the cost attained by OURS-E for problem instances with $N = 8$ for problem instances with point clouds defined between 5 and 8 units from the world frame.

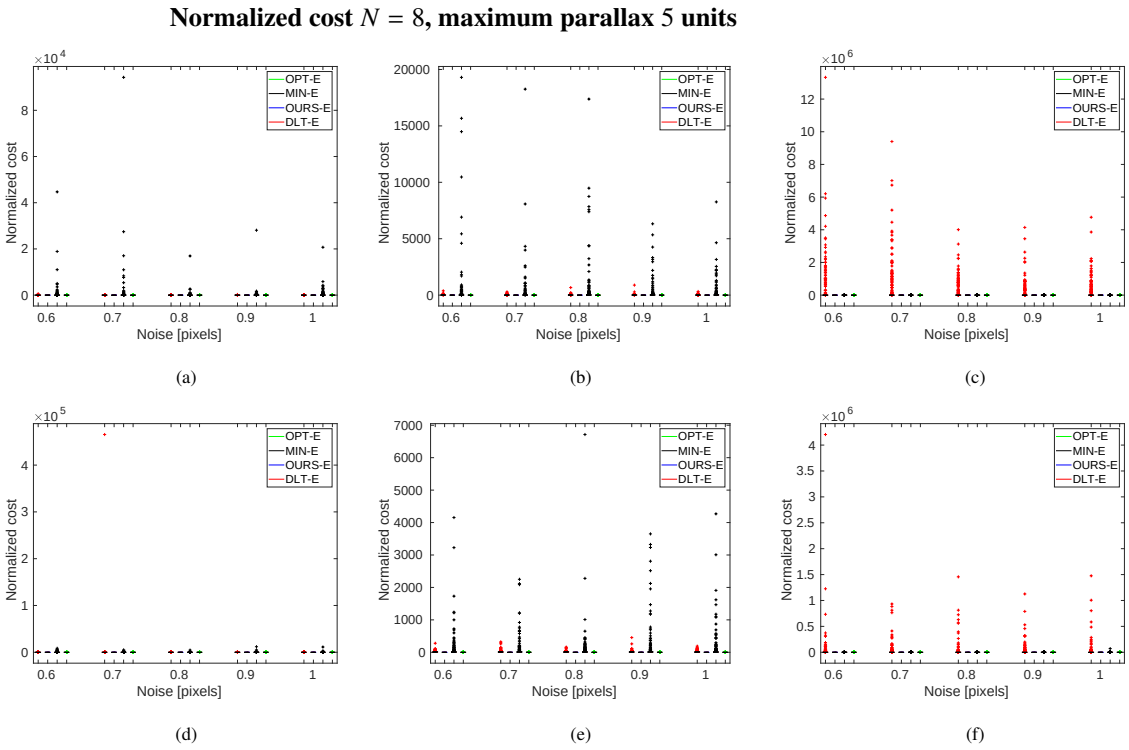


Figure 14: **Normalized cost $N = 8$, Maximum parallax 5 units**: Cost by the different solvers, including MIN-E, normalized by the cost attained by OURS-E for problem instances with $N = 8$ for problem instances with maximum parallax 5 units.

levels from 0.6 to 1.0 `pix` . In addition, we tested two different scene configurations: (a) point cloud with minimum distance of 5 units from world frame; and (b) maximum parallax of 5 units. Figures 13 and 14 show the normalized error for all the solvers (including MIN-E) for $N = 8$ correspondences and varying noise level (X-axis) for (a) minimum distance 5 units; and (b) maximum parallax 5 units, respectively. We follow the same format than in the other figures. Notice that, despite the problem instances being close to the minimal case, the non-minimal solvers attain better solutions (with lower costs) than the minimal solver. We observe a large number of outliers for both MIN-E and DLT-E for all the considered configurations. Whereas this could be related to the small optimal cost, we notice that these gross outliers don't appear for the optimal solver OPT-E. We provide in figures 15 and 16 the same results but only for OPT-E and OURS-E. Notice that for most cases, but not all of them, our solver is able to obtain better solutions. For reference, we include in figures 17 and 18 the results under the same setups for problems with $N = 12$ correspondences. For the other solvers the results were similar, and thus we only show the comparison with OPT-E. We observe that we obtain better results with more correspondences, as expected, as we show for problem instances with $N = 15, 50, 100$ in previous figures. Interesting, our solver performs better for problem instances with maximum parallax 5 units, and the number of cases in which OPT-E attains better results is reduced. As with the non-minimal problem instances, forward motions hinder the performance of OPT-E (see the different scale for the Y-axis), even for $N = 8$ correspondences.

Rotation error Last, we include the error in rotation in figure 19 for the the different configurations of cameras and $N = 50$ correspondences. We follow the same format where the X-axis indicates the level of noise applied to the correspondences. As expected, the minimal solver MIN-E has large error when the noise of the observations is large. Notice that for forward motions the solver OPT-E becomes unstable, attaining large errors. This is also observed when the rotation is not a pure Y-rotation (fig. (19f)). When the number of correspondences increases up to $100 - 200$, we also obtain large errors for OPT-E. These errors do not appear in our proposal, not even with forward motions with and without noisy rotation. We want to point out as well the low error attained by the initial guess in DLT-E in almost all the problem instances and camera configurations. For noisy Y-rotations the errors are larger and independent of the noise. We obtain large values even for noiseless correspondences for all the solvers.

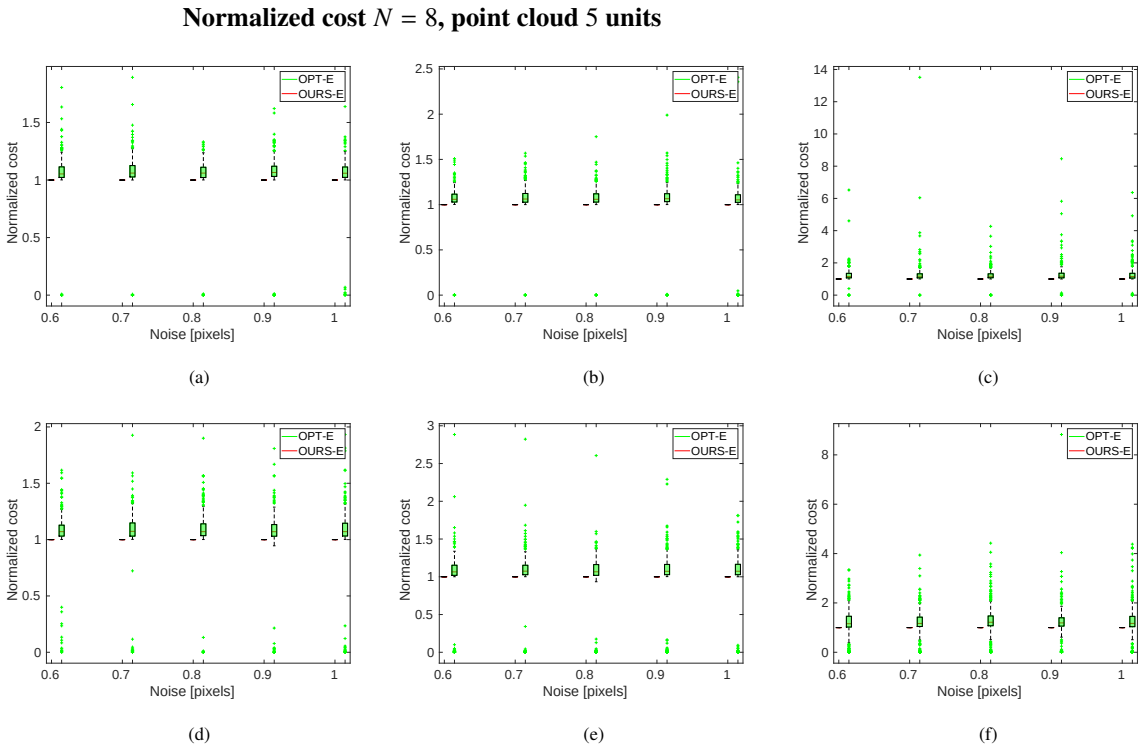


Figure 15: **Normalized cost** $N = 8$**, point cloud** 5 **units**: Cost by the optimal solvers OPT-E and OURS-E normalized by the cost attained by OURS-E for problem instances with $N = 8$ for problem instances with point clouds defined from 5 units.

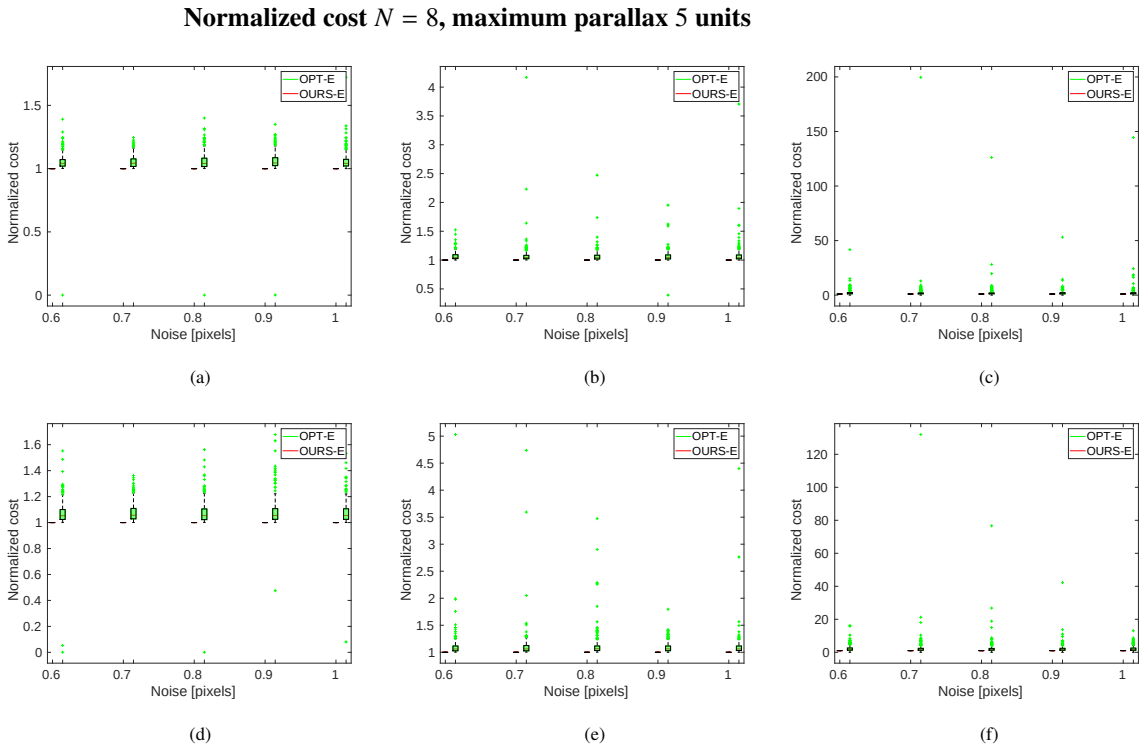


Figure 16: **Normalized cost $N = 8$, Maximum parallax 5 units**: Cost by the optimal solvers OPT-E and OURS-E normalized by the cost attained by OURS-E for problem instances with $N = 8$ for problem instances with maximum parallax 5 units.

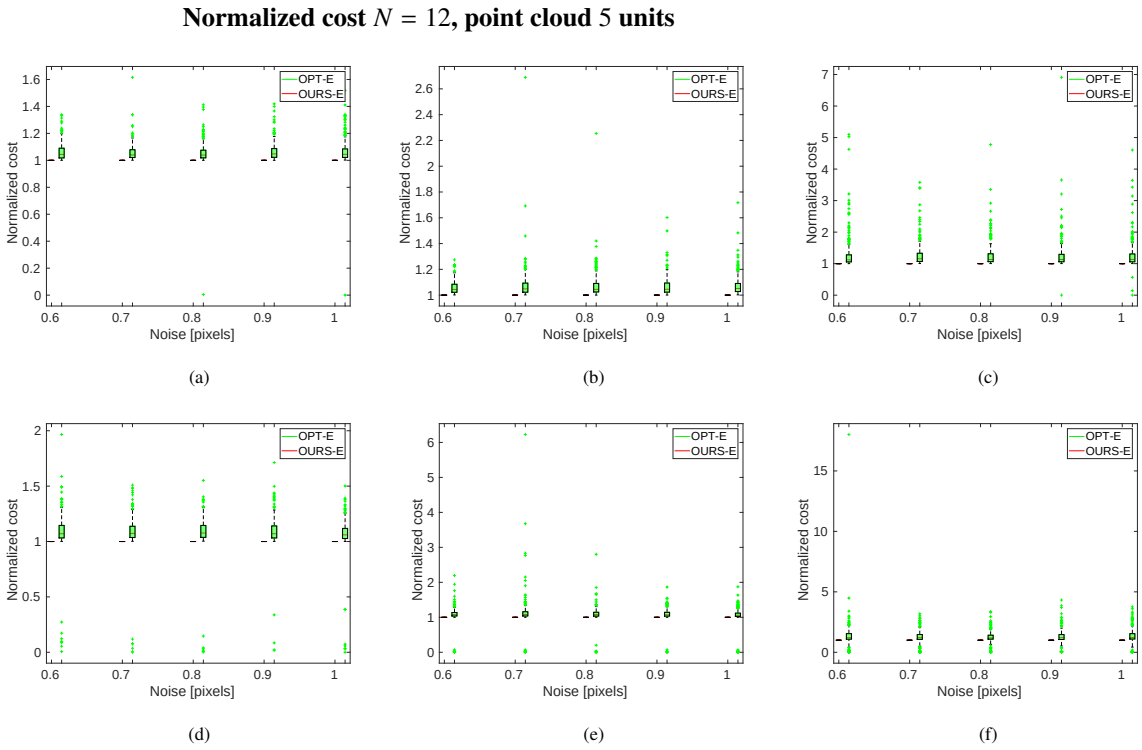


Figure 17: **Normalized cost** $N = 12$**, point cloud** 5 **units**: Cost by the optimal solvers OPT-E and OURS-E normalized by the cost attained by OURS-E for problem instances with $N = 12$ for problem instances with point clouds defined from 5 units.

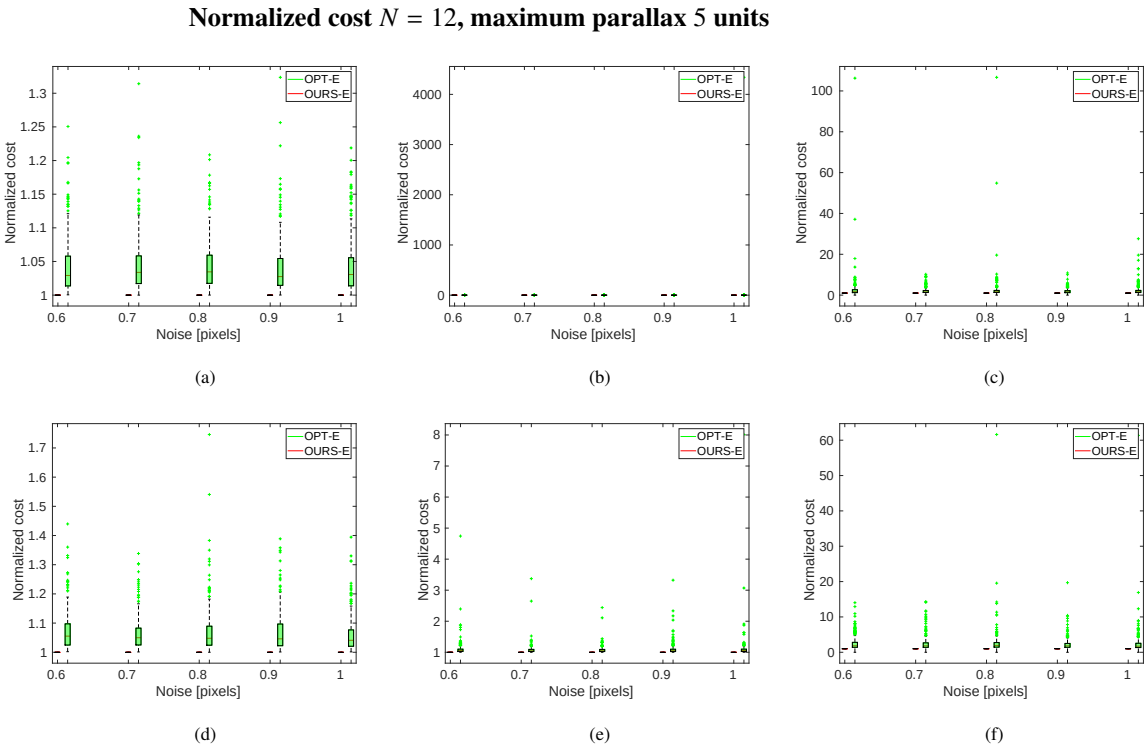


Figure 18: **Normalized cost $N = 12$, Maximum parallax 5 units**: Cost by the optimal solvers OPT-E and OURS-E normalized by the cost attained by OURS-E for problem instances with $N = 12$ for problem instances with maximum parallax 5 units.

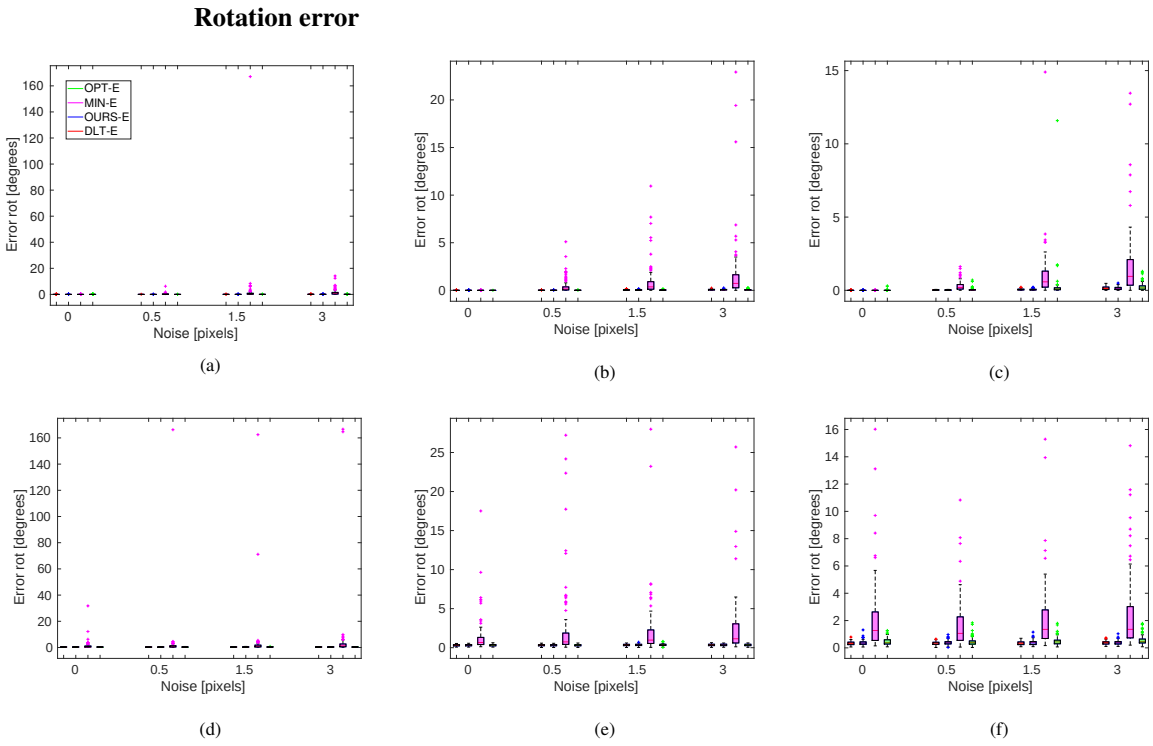


Figure 19: **Rotation error**: Rotation error for the solutions returned by each solver. Top row shows the experiments with pure Y-rotation and bottom row the experiments where a small perturbation was added to the rotation. Left column: translation is a random vector; middle column: lateral translation on the X-direction; and right column: forward motion (Z-axis). The number of correspondences is fixed to $N = 50$.