A Novel Closed-Form Approach for Enhancing Efficiency in Pose Estimation from 3D Correspondences

Ezio Malis

A Novel Closed-Form Approach for Enhancing Efficiency in Pose Estimation from 3D Correspondences

Ezio Malis

***Abstract*—Determining the pose using 3D data captured from two distinct frames holds significant significance in various robotic applications such as odometry, simultaneous localization and mapping and place recognition. Typically, this pose is calculated through the resolution of a least-squares problem, which involves establishing correspondences between points, points and planes, or points and lines. This non-linear least-squares problem can be addressed either through iterative optimization or, with greater efficiency, through closed-form solutions achieved via polynomial system solvers. In this paper, the focus is on enhancing the computational efficiency of polynomial solvers utilizing resultants. The newly proposed methodology outperforms previous techniques in terms of speed, all while maintaining identical levels of accuracy and robustness. Through simulations and real-world experiments, we validate the superior precision and robustness of the proposed algorithm when compared with prior methods.**

***Index Terms*—Localization, Optimization and Optimal Control, SLAM, Autonomous Vehicle Navigation**

I. Introduction

The widespread adoption of 3D sensors such as lidars [1], stereo cameras [2], and RGB-D cameras [3] has become a standard practice in the field of autonomous robotics. These sensors are highly valued for their precision in localizing robots and capturing detailed 3D information about their surroundings. They generate a collection of 3D points during each data acquisition process. Consequently, the alignment of two sets of 3D points is of paramount importance for various tasks, including odometry [4], Simultaneous Localization and Mapping (SLAM) [5], and place recognition [6]. Typically, this challenge is approached as the solution to a non-linear optimization problem, often involving least-squares optimization, following the association of points with points [7], points with planes [8], or points with lines [9].

The initial proposal for a unified solution to address the points-to-points, points-to-line, and points-to-plane registration problem emerged in [10]. In their approach, the authors introduced an iterative branch and bound optimization solver that is capable to find the global optimum for non-convex problems. However, it's worth noting that iterative solvers tends to exhibit relatively slow performance and typically converges to a suboptimal solution, meaning they approach the global minimum with a specified level of precision. To overcome this problem and speed up the computation time [11] and [12] use a Lagrangian dual relaxation solver. These methods can find the global minimum but are iterative, therefore time consuming. Even if it is possible to certify the optimality of the recovered estimate a posteriori [13] we cannot guarantee that the optimal solution will be found. In this paper, the focus will be given on algorithms for which the global minimum is guaranteed to be found in a reasonable time.

A significantly accelerated algorithm can be obtained by employing solvers which operate using a "closed-form" approach and for which the global minimum is guaranteed to be found, as discussed in [14], [15], [16], [17]. These solvers rely exclusively on basic linear algebra operations to solve the problem efficiently. The translation is removed from the equations since it appears quadratically in the least-squares problem, resulting in a simplified optimization problem dependent solely on rotation (with 3 degrees of freedom). The non-linearity arises from the rotation matrix, as it must remain orthonormal. Therefore, the choice of rotation matrix parametrization plays a pivotal role in converting the least-squares optimization into a polynomial problem.

A common method for representing rotations are unit quaternions, as in [7], [15], [17]. This representation is always valid but not minimal, requiring the use of 4 variables to describe all possible rotations. Another alternative is the Cayley-Gibbs-Rodrigues (C-G-R) representation, as in [14], [16]. These parameters are minimal, necessitating only 3 variables, but they cannot effectively represent rotations with an angle of π around an arbitrary axis. Despite its minimal nature, the C-G-R parametrization results in a rational representation of the rotation. In [16], the authors introduced intermediate variables to transform the problem from a rational to a polynomial form. However, one of these intermediate variables is dependent on the other three, but it was treated as a free variable, disregarding the constraint. Both methods in [16] and in [15] built a solver using Gröebner basis. A "u-resultant" MacAulay method has been proposed in [17] using a quaternion parametrisation. This approach outperforms previous methods in term of accuracy and robustness but was not consistently the more efficient in term of computation time. In general, resultant-based usually lead to larger and less efficient solvers than Gröebner basis solvers.

In this paper, we aim to address the limitations and unanswered questions that emerged from earlier study and offer novel perspectives and methodologies to advance the field. Notably, one of the main contributing factors to the limited efficiency observed in the "u-resultant" approach in [17] was the failure to leverage the two-fold symmetry inherent in the quaternion parametrization, a missed opportunity that could have led to a more efficient solver, as previously proposed in [18], [19], [20]. Moreover, it has been shown in [21] that resultant-based approaches can lead to efficient solvers for minimal problems. For these reasons, the approach proposed in this paper shifts from utilizing the "u-resultant" method to adopting the "h-resultant" approach [22]. The "h-resultant" approach consists in *hiding* one variable considering it as a

Ezio Malis is with the ACENTAURI team at Université Côte d'Azur, Inria, France. ezio.malis@inria.fr

parameter. Such a hidden variable approach has been used in the past to solve different problems in computer vision [23]. Therefore, the key contribution of this paper is the design of a novel efficient "h-resultant" solver that exploits the two-fold symmetry of the quaternion parametrisation, that was not considered in previous works.

II. Theoretical background

Let $\mathbf{m}_c$ be a 3D point in a current frame $\mathcal{F}_c$. The point belongs to a line in the same frame with origin $\mathbf{m}$ (any point on the line) and unitary direction vector $\mathbf{d}$ if and only if:

$$[\mathbf{d}]_\times^2 (\mathbf{m}_c - \mathbf{m}) = 0 \tag{1}$$

And it belongs to a plane in the same frame with origin $\mathbf{m}$ (any point on the plane) and with unitary normal vector $\mathbf{n}$ if and only if:

$$\mathbf{n}^\top (\mathbf{m}_c - \mathbf{m}) = 0 \tag{2}$$

We consider now the same point but in a different reference frame $\mathcal{F}_r$. Let ${}^c\mathbf{T}_r \in SE(3)$ the homogeneous transformation matrix between the two frames:

$${}^c\mathbf{T}_r = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$$

where $\mathbf{R} \in SO(3)$ is the 3×3 rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is the 3×1 translation vector. The current point $\mathbf{m}_c$ is obtained transforming the reference point $\mathbf{m}_r$ as follows:

$$\mathbf{m}_c = \mathbf{R}\, \mathbf{m}_r + \mathbf{t} \tag{3}$$

Plugging equation (3) into equation (1) we get:

$$[\mathbf{d}]_\times^2 (\mathbf{R}\, \mathbf{m}_r + \mathbf{t} - \mathbf{m}) = 0 \tag{4}$$

And plugging equation (3) into equation (2) we get:

$$\mathbf{n}^\top (\mathbf{R}\, \mathbf{m}_r + \mathbf{t} - \mathbf{m}) = 0 \tag{5}$$

Stacking the entries r_{ij} of the rotation matrix $\mathbf{R}$, rows by rows, into a 9×1 vector:

$$\mathbf{r} = [\mathbf{R}]_\vee = [r_{11};\, r_{12};\, r_{13};\, r_{21};\, r_{22};\, r_{23};\, r_{31};\, r_{32};\, r_{33}] \tag{6}$$

we can write:

$$\mathbf{R}\, \mathbf{m}_r = \mathbf{M}\, \mathbf{r} \tag{7}$$

where $\mathbf{M}$ is the following 3×9 matrix:

$$\mathbf{M} = \begin{bmatrix} \mathbf{m}_r^\top & 0 & 0 \\ \mathbf{0} & \mathbf{m}_r^\top & 0 \\ 0 & 0 & \mathbf{m}_r^\top \end{bmatrix}$$

In order to get a similar notation for all cases (points, lines and planes), plugging equation (7) into equations (3), (4) and (5) we obatain:

$$\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m} = 0 \tag{8}$$

For point-to-point correspondences the point $\mathbf{m}$ is set to the current points $\mathbf{m} = \mathbf{m}_c$ in equation (8).
For point-to-line correspondences $\mathbf{m}$ is an arbitrary point on the current line, and from equation (4) we get:

$$[\mathbf{d}]_\times^2 (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m}) = 0 \tag{9}$$

For point-to-plane correspondences $\mathbf{m}$ is an arbitrary point on the current plane, and from equation (5) we get:

$$\mathbf{n}^\top (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m}) = 0 \tag{10}$$

Our objective is to estimate the pose (rotation vector $\mathbf{r}$ and translation vector $\mathbf{t}$) given several measured points and corresponding planes. Note that even if the equations are linear in the vector $\mathbf{r}$, the problem is non-linear since the entries of $\mathbf{r} = [\mathbf{R}]_\vee$ must satisfy the constraints $\mathbf{R}^\top \mathbf{R} = \mathbf{I}$.

A. Rotation parametrisation

The rotation matrix $\mathbf{R} \in SO(3)$ is a 3×3 matrix but it has 3 degrees of freedom only since it is orthonormal, $\mathbf{R}^\top \mathbf{R} = \mathbf{I}$, that means 6 constraints on the entries of the rotation. Instead to represent the rotation by 9 entries subject to 6 constraints we can use more compact representations.

One can use the Cayley-Gibbs vector parametrisation. Unfortunately, this parametrisation leads to a rational form for $\mathbf{R}$ and is not complete since it cannot represent rotations when $\theta = \pi$. A more complete rotation parametrisation is the quaternion $\mathbf{q} = [q_r, \mathbf{q}_i] = [w; x; y; z]$ for which we have:

$$\mathbf{R}(\mathbf{q}) = \mathbf{I} + 2q_r [\mathbf{q}_i]_\times + 2 [\mathbf{q}_i]_\times^2 \tag{11}$$

$$\mathbf{q}^\top \mathbf{q} = 1 \tag{12}$$

All the rows of the rotation matrix can be stacked in a (9×1) vector $\mathbf{r}(\mathbf{q}) = [\mathbf{R}]_\vee$ that is quadratic in the 4 unknowns x, y, z, w. This non minimal parametrisation has a 2-fold symmetry since $\mathbf{r}(\mathbf{q}) = \mathbf{r}(-\mathbf{q})$:

$$\mathbf{R}(\mathbf{q}) = \mathbf{R}(-\mathbf{q}) \tag{13}$$

and therefore:

$$\mathbf{r}(\mathbf{q}) = \mathbf{r}(-\mathbf{q}) \tag{14}$$

B. The weighted least squares problem

Considering n_m point to point correspondences, n_l point to line correspondences and n_p point to plane correspondences the weighted least squares optimisation problem can be written as follows:

$$\min_{\mathbf{r},\mathbf{t}} \frac{1}{2} \sum_{i=1}^{n_m} w_{mi}^2 d_{mi}^2 + \frac{1}{2} \sum_{j=1}^{n_l} w_{lj}^2 d_{lj}^2 + \frac{1}{2} \sum_{k=1}^{n_p} w_{kp}^2 d_{pk}^2 \tag{15}$$

where w_m, w_l and w_p are weights. The squared distance point to point is given by:

$$d_m^2 = (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m})^\top (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m}) \tag{16}$$

the squared distance point to line is given by:

$$d_l^2 = (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m})^\top [\mathbf{d}]_\times^4 (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m}) \tag{17}$$

and the squared distance point to plane is given by:

$$d_p^2 = (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m})^\top (\mathbf{n}\mathbf{n}^\top) (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m}) \tag{18}$$

Introducing a (3×3) symmetric matrix $\mathbf{W}$, all possible squared distances can be defined with an unique equation:

$$d^2 = (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m})^\top \mathbf{W}\, (\mathbf{M}\, \mathbf{r} + \mathbf{t} - \mathbf{m})$$

where the choice of the entries of the semi-definite positive weighting matrix $\mathbf{W}$ determines a specific distance :

- for a distance point to point select $\mathbf{W} = w_m^2\ \mathbf{I}$
- for a distance point to line select $\mathbf{W} = -w_l^2\ [\mathbf{d}]_\times^2$
- for a distance point to plane select $\mathbf{W} = w_p^2\ \mathbf{n}\mathbf{n}^\top$

Therefore, the optimisation problem mixing points, lines and plane can be generally written as follows:

$$\min_{\mathbf{r},\mathbf{t}} \sum_{k=1}^{n} \left(\mathbf{M}_k\ \mathbf{r} + \mathbf{t} - \mathbf{m}_k\right)^\top \mathbf{W}_k\ \left(\mathbf{M}_k\ \mathbf{r} + \mathbf{t} - \mathbf{m}_k\right) \quad (19)$$

This optimisation problem can be solved iteratively starting from an initial guess of the global optimum, or in a closed form as proposed in the next section.

III. Least square closed-form solution

Similar to prior studies [10], [15], [16], [17], we remove the translation vector that is squared in the cost function (19), leading to the resolution of a novel non-linear problem depending solely on the rotation.

A. Separating rotation and translation

The least squares problem in equation (19) can be written as a quadratic function of $\mathbf{t}$:

$$c(\mathbf{r},\mathbf{t}) = \mathbf{t}^\top \sum_{k=1}^{n} \mathbf{W}_k \mathbf{t} + 2\sum_{k=1}^{n} \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right)^\top \mathbf{W}_k\ \mathbf{t} + \sum_{k=1}^{n} \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right)^\top \mathbf{W}_k\ \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right) \quad (20)$$

The optimal solution for the translation is obtained from:

$$\frac{\partial c(\mathbf{r},\mathbf{t})}{\partial \mathbf{t}} = \sum_{k=1}^{n} \mathbf{W}_k\ \mathbf{t} + \sum_{k=1}^{n} \mathbf{W}_k^\top\ \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right) = 0$$

That is:

$$\mathbf{t} = -\left(\sum_{k=1}^{n} \mathbf{W}_k\right)^{-1} \sum_{k=1}^{n} \mathbf{W}_k^\top\ \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right)$$

where $\mathbf{t}$ is a linear function in $\mathbf{r}$. The translation can be obtained as a linear function of the rotation vector $\mathbf{r}$:

$$\mathbf{t} = \mathbf{A}_t\ \mathbf{r} + \mathbf{b}_t \quad (21)$$

where the matrix $\mathbf{A}_t$ and vector $\mathbf{b}_t$ are:

$$\mathbf{A}_t = -\left(\sum_{k=1}^{n} \mathbf{W}_k\right)^{-1} \sum_{k=1}^{n} \mathbf{W}_k^\top \mathbf{M}_k \quad (22)$$

$$\mathbf{b}_t = +\left(\sum_{k=1}^{n} \mathbf{W}_k\right)^{-1} \sum_{k=1}^{n} \mathbf{W}_k^\top \mathbf{m}_k \quad (23)$$

Plugging back the optimal translation into the cost function we have now to solve the following optimisation problem:

$$\min_{\mathbf{r}} \sum_{k=1}^{n} \left(\overline{\mathbf{M}}_k\mathbf{r} - \overline{\mathbf{m}}_k\right)^\top \mathbf{W}_k\ \left(\overline{\mathbf{M}}_k\mathbf{r} - \overline{\mathbf{m}}_k\right)$$

where:

$$\overline{\mathbf{M}}_k = \mathbf{M}_k + \mathbf{A}_t \quad (24)$$

$$\overline{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{b}_t \quad (25)$$

we can set the problem in the following canonical form:

$$\min_{\mathbf{r}} c(\mathbf{r}) = \min_{\mathbf{r}} \left(\mathbf{r}^\top \mathbf{A}_r \mathbf{r} + 2\mathbf{b}_r^\top \mathbf{r} + c_r\right) \quad (26)$$

where $\mathbf{A}_r$ is the following 9×9 matrix:

$$\mathbf{A}_r = +\sum_{k=1}^{n} \overline{\mathbf{M}}_k^\top \mathbf{W}_k \overline{\mathbf{M}}_k$$

$\mathbf{b}_r$ is the following 9×1 vector:

$$\mathbf{b}_r = -\sum_{k=1}^{n} \overline{\mathbf{M}}_k^\top \mathbf{W}_k \overline{\mathbf{m}}_k$$

and c_r is the following scalar:

$$c_r = +\sum_{k=1}^{n} \overline{\mathbf{m}}_k^\top \mathbf{W}_k \overline{\mathbf{m}}_k$$

B. Closed-form solution for the rotation

Using the quaternion parametrisation, the vector $\mathbf{r}$ can be written as a quadratic function of 4 unknowns $\mathbf{q} = [w; x; y; z]$. We can impose the constraint $\mathbf{q}^\top\mathbf{q} = 1$ using the Lagrange multiplier method [24]:

$$\min_{\lambda,\mathbf{q}} \mathcal{L}(\lambda,\mathbf{q}) = c(\mathbf{r}(\mathbf{q})) + \lambda(1 - \mathbf{q}^\top\mathbf{q}) \quad (27)$$

The stationary points of the Lagrangian are found by solving the following polynomial equations:

$$\frac{\partial \mathcal{L}(\lambda,\mathbf{q})}{\partial \lambda} = 1 - \mathbf{q}^\top\mathbf{q} = 0 \quad (28)$$

$$\frac{\partial \mathcal{L}(\lambda,\mathbf{q})}{\partial \mathbf{q}} = \mathbf{g}^\top(\mathbf{q}) - \lambda\mathbf{q}^\top = 0 \quad (29)$$

where $\mathbf{g}^\top(\mathbf{q}) = \begin{bmatrix} g_w(\mathbf{q}) & g_x(\mathbf{q}) & g_y(\mathbf{q}) & g_z(\mathbf{q}) \end{bmatrix}$ is the following (1×4) vector:

$$\mathbf{g}^\top(\mathbf{q}) = \frac{\partial c(\mathbf{r})}{\partial \mathbf{r}} \frac{\partial \mathbf{r}(\mathbf{q})}{\partial \mathbf{q}}$$

that is the multiplication of the following (1×9) vector:

$$\frac{\partial c(\mathbf{r})}{\partial \mathbf{r}} = \left(\mathbf{A}_r\mathbf{r} + \mathbf{b}_r\right)^\top$$

and the following (9×4) Jacobian matrix depending linearly on $\mathbf{q}$:

$$\frac{\partial \mathbf{r}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{J}(\mathbf{q})$$

Starting from the optimality condition in equation (29):

$$\mathbf{g}(\mathbf{q}) = \lambda\mathbf{q} \quad (30)$$

eliminate λ with the wedge product (or exterior product):

$$\mathbf{g}(\mathbf{q}) \wedge \mathbf{q} = \lambda\mathbf{q} \wedge \mathbf{q} = 0 \quad (31)$$

This polynomial system of equations has been solved eliminating λ and using a "u-resultant" approach in [17]. The solution of the system can be extracted by computing the

eigenvectors of a 128×128 matrix. In this paper, the two-fold symmetry of the quaternion parametrisation (14) is exploited to obtain a faster solver as suggested in [18], [19], [20].

To begin with, it's important to recognize that $\mathbf{g}(\mathbf{q})$ can be expressed as the combination of a third-order polynomial, denoted as $\mathbf{g}_3(\mathbf{q})$, and a first-order polynomial, denoted as $\mathbf{g}_1(\mathbf{q})$:

$$\mathbf{g}(\mathbf{q}) = \mathbf{g}_3(\mathbf{q}) + \mathbf{g}_1(\mathbf{q}) = \lambda\mathbf{q} \tag{32}$$

Substituting the quaternion constraint $\mathbf{q}^\top\mathbf{q} = 1$ into equation (32) we can write the following third order *homogeneous* polynomial equations of degree 3 in $\mathbf{q}$:

$$\mathbf{g}_3(\mathbf{q}) + (\mathbf{q}^\top\mathbf{q})\mathbf{g}_1(\mathbf{q}) = \lambda(\mathbf{q}^\top\mathbf{q})\mathbf{q} \tag{33}$$

Secondly, the approach proposed in this paper shifts from utilizing the "u-resultant" method to adopting the "h-resultant" approach [22]. The "h-resultant" approach consists in *hiding* one variable considering it as a parameter. In our case, λ is chosen to be the hidden variable in order to ensure that the two-fold symmetry is not broken. We obtain the following 4 polynomials $\mathbf{e}(\mathbf{q}, \lambda)$ such that:

$$\mathbf{e}(\mathbf{q}, \lambda) = \mathbf{g}_3(\mathbf{q}) + (\mathbf{q}^\top\mathbf{q})\mathbf{g}_1(\mathbf{q}) - \lambda(\mathbf{q}^\top\mathbf{q})\mathbf{q} = 0 \tag{34}$$

Let us define the homogeneous monomial set as follows:

$$\mathbf{m}_d = \{w^{d_w} x^{d_x} y^{d_y} z^{d_z}\} \; \forall \; d_w, d_x, d_y, d_z : d = d_w + d_x + d_y + d_z$$

For n unknowns, the total number of possible monomials will be $(d+n-1)!/(d!\ (n-1)!$ In our case $n = 4$ and $d = 3$, therefore using the graded lexicographic order with $w > x > y > z$, the polynomial equation 34 contains 20 monomials in $\mathbf{q}$, and can be written as:

$$\mathbf{e}(\mathbf{q}, \lambda) = \mathbf{C}_3(\lambda, \mathbf{c})\ \mathbf{m}_3(\mathbf{q}) = 0 \tag{35}$$

where $\mathbf{C}_3$ is a 4×20 matrix depending linearly on the coefficients $\mathbf{c}$ and the hidden variable λ.

Consider all the possible sixth order monomials $\mathbf{m}_6(\mathbf{q})$ of size 84×1. We can create $84 \times 4 = 336$ new equations:

$$\mathbf{e}(\mathbf{q}, \lambda, \mathbf{c}) \otimes \mathbf{m}_6(\mathbf{q}) = \mathbf{E}(\lambda, \mathbf{c})\ \mathbf{m}_9(\mathbf{q}) = 0 \tag{36}$$

where $\otimes$ is the Kronecker product, $\text{size}(\mathbf{E}(\lambda), \mathbf{c}) = (336 \times 220)$ and $\text{rank}(\mathbf{E}(\lambda), \mathbf{c}) = 220$ for a generic value of λ. Note that we can write:

$$\mathbf{E}(\lambda, \mathbf{c}) = \mathbf{E}_0(\mathbf{c}) + \lambda\ \mathbf{E}_1$$

where $\mathbf{E}_0$ depends on the coefficients $\mathbf{c}$ while $\mathbf{E}_1$ is a constant matrix.

Similarly to [17], 6 additional homogeneous equations of degree 4 $\mathbf{f}(\mathbf{q})$ can be created eliminating λ using the wedge product:

$$\begin{aligned}\mathbf{f}(\mathbf{q}, \mathbf{c}) &= (\mathbf{g}_3(\mathbf{q}) + \mathbf{g}_1(\mathbf{q})\mathbf{q}^\top\mathbf{q}) \wedge \mathbf{q} - \lambda(\mathbf{q}^\top\mathbf{q})\mathbf{q} \wedge \mathbf{q} = \\ &= (\mathbf{g}_3(\mathbf{q}) + \mathbf{g}_1(\mathbf{q})\mathbf{q}^\top\mathbf{q}) \wedge \mathbf{q} = 0\end{aligned} \tag{37}$$

Therefore, consider all the possible fifth order monomials $\mathbf{m}_5(\mathbf{q})$ of size 56×1. We can create $56 \times 6 = 336$ equations:

$$\mathbf{f}(\mathbf{q}, \mathbf{c}) \otimes \mathbf{m}_5(\mathbf{q}) = \mathbf{F}(\mathbf{c})\ \mathbf{m}_9(\mathbf{q}) = 0 \tag{38}$$

From the set of equations (36) and (38), the objective now is to extract a 220×220 action matrix $\mathbf{M}$ with the following structure:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_0 & \mathbf{D}_0 \end{bmatrix} + \lambda \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ 0 & 0 \end{bmatrix} \tag{39}$$

where $\text{size}(\mathbf{A}) = (40 \times 40)$, $\text{size}(\mathbf{B}) = (40 \times 180)$, $\text{size}(\mathbf{C}) = (180 \times 40)$, $\text{size}(\mathbf{D}) = (180 \times 180)$.

Note that the we need to choose $\mathbf{M}$ such that: $\text{rank}(\mathbf{M}) = 220$, $\text{rank}(\mathbf{A}) = 40$ and $\text{rank}(\mathbf{D}) = 180$ for a general value of λ. The non trivial solution $\mathbf{m}_9 \neq 0$ of our problem is to find λ such that:

$$\mathbf{M}(\lambda, \mathbf{c})\ \mathbf{m}_9(\mathbf{q}) = 0$$

This is possible if and only if:

$$\det(\mathbf{M}) = \det(\mathbf{M}_0 + \lambda\mathbf{M}_1) = 0$$

Noting that, since $\det(\mathbf{D}) = \det(\mathbf{D}_0) \neq 0$:

$$\det(\mathbf{M}) = \det\left(\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}\right) = \det\big(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}\big)\det(\mathbf{D})$$

Therefore:

$$\det\big((\mathbf{A}_0 - \mathbf{B}_0\mathbf{D}_0^{-1}\mathbf{C}_0) + \lambda(\mathbf{A}_1 - \mathbf{B}_1\mathbf{D}_0^{-1}\mathbf{C}_0)\big) = 0$$

Setting the two following 40×40 matrices:

$$\mathbf{Q}_0 = \mathbf{A}_0 - \mathbf{B}_0\mathbf{D}_0^{-1}\mathbf{C}_0$$
$$\mathbf{Q}_1 = \mathbf{A}_1 - \mathbf{B}_1\mathbf{D}_0^{-1}\mathbf{C}_0$$

The solutions are the real generalised eigenvalues λ:

$$\det(\mathbf{Q}_0 + \lambda\mathbf{Q}_1) = 0$$

The quaternion corresponding to the solutions can be extracted from the real generalised eigenvectors. Compared to [17] we need to compute the eigenvalue decomposition of a 40×40 matrix instead of a 128×128 matrix. This, considerably reduce the computation time. To extract safely the solution from the 40×1 eigenvectors we choose a particular ordering of the monomials $\mathbf{m}_9$ and we select the action matrix $\mathbf{M}$ as described in the following subsections.

1) Selecting the monomials ordering:
The monomial ordering in the vector $\mathbf{m}_9$ is important when extracting the solutions. We choose the grdlex ordering except for the first 16 monomials:

$$\mathbf{m}_9 = [\mathbf{m}_w, \mathbf{m}_x, \mathbf{m}_y, \mathbf{m}_z, \cdots]$$

where:

$$\mathbf{m}_w = [w^9, w^8x, w^8y, w^8z]$$
$$\mathbf{m}_x = [wx^8, x^9, x^8y, x^8z]$$
$$\mathbf{m}_y = [wy^8, xy^8, y^9, y^8z]$$
$$\mathbf{m}_z = [wz^8, xz^8, yz^8, z^9]$$

With such choice, we can always extract the solution from the first 16 monomials of $\mathbf{m}_9$ even when one of the variables is 0. Indeed, we have $||\mathbf{m}_w|| = w^4$, $||\mathbf{m}_x|| = x^4$, $||\mathbf{m}_y|| = y^4$, $||\mathbf{m}_z|| = z^4$. The solution is extracted depending on the maximum measured values extracted from $\mathbf{m}_9$:

If $\max(w^4, x^4, y^4, z^4) = w^4$, then:

$$\mathbf{q} = [w, x, y, z] = \frac{\mathbf{m}_w}{||\mathbf{m}_w||}$$

If $\max(w^4, x^4, y^4, z^4) = x^4$, then:

$$\mathbf{q} = [w, x, y, z] = \frac{\mathbf{m}_x}{||\mathbf{m}_x||}$$

If $\max(w^4, x^4, y^4, z^4) = y^4$, then:

$$\mathbf{q} = [w, x, y, z] = \frac{\mathbf{m}_y}{||\mathbf{m}_y||}$$

If $\max(w^4, x^4, y^4, z^4) = z^4$, then:

$$\mathbf{q} = [w, x, y, z] = \frac{\mathbf{m}_z}{||\mathbf{m}_z||}$$

2) Selecting the Action Matrix:

The Action Matrix $\mathbf{M}$ in equation (39) can be chosen in several ways. The easiest method we found is to select the $\mathbf{A}$ and $\mathbf{B}$ matrices from the set of equations (36) while the $\mathbf{C}$ and $\mathbf{D}$ matrices are selected from the set of equations (38). Note that $\operatorname{rank} \mathbf{A} = 40$ and $\operatorname{rank} \mathbf{D} = 180$. One can perform the selection on-line with a QR decomposition but this would lead to a higher computation time. Instead, the choice can be done once and for all in an off-line "learning" step where several simulation are performed to select the best matrices as proposed in [17].

IV. Iteratively re-weighted least squares

The optimisation problem in equation (19) is an ordinary weighted Least Squares (LS) problem. Indeed, the 3×3 positive symmetric matrix $\mathbf{W}_k$ can be written using the Cholesky decomposition under the form:

$$\mathbf{W}_k = w_k \mathbf{L}_k^\top \mathbf{L}_k$$

Therefore, one coan define the residuals ε_k as follows:

$$\varepsilon_k(\mathbf{x}) = \|\mathbf{L}_k \, (\mathbf{M}_k \, \mathbf{r}(\mathbf{q}) + \mathbf{t} - \mathbf{m}_k)\|$$

where $\mathbf{x} = [\mathbf{t}; \mathbf{q}]$ is a vector containing the unknowns. Finally, the weighted problem in equation (19) can be written as:

$$\min_{\mathbf{x}} \sum_{k=1}^{n} w_k \, \varepsilon_k^2(\mathbf{x})$$

This conventional Least Squares (LS) approach for estimating $\mathbf{x}$ lacks of robustness, as the cost function may exhibit unbounded growth with the residuals ε_k. Consequently, outliers with extremely large residuals (e.g. due to wrong correspondences) can exert an infinitely substantial impact on the resultant estimation. Indeed, a solitary outlier has the capacity to entirely compromise the integrity of the Least Squares estimate, leading to its breakdown.

Numerous robust estimator methodologies have been suggested in the literature [25]. Among these approaches, M-estimators [26], are frequently favoured for parameter estimation. This preference stems from their robustness against outliers and their capacity to furnish dependable estimates even in scenarios involving skewed or heavy-tailed distributions. The primary objective of M-estimators is to alleviate the influence of outliers by replacing the squared residuals with a function $\rho(\varepsilon)$ that exhibits a less rapid increase as a function of the residual:

$$\min_{\mathbf{x}} \sum_{k=1}^{n} \rho(\varepsilon_k(\mathbf{x})) \tag{40}$$

For example, choosing $\rho(\varepsilon_k(\mathbf{x})) = |\varepsilon_k(\mathbf{x})|$ we minimize the least absolute errors rather than the least square errors. Note that Least Squares can be considered an M-estimator, even though it is not a *robust* M-estimator. One popular approach to implement the M-estimators is to use the iteratively re-weighted least squares (IRLS) method [27], [26]. It can be obtained by observing that the optimal value of the robust optimisation is obtained solving the following system of equations:

$$\sum_{k=1}^{n} \psi(\varepsilon_k(\mathbf{x})) = 0$$

where

$$\psi(\varepsilon_k(\mathbf{x})) = \frac{\partial \rho(\varepsilon_k(\mathbf{x}))}{\partial \mathbf{x}}$$

Defining the weights w_k as:

$$w(\varepsilon_k(\mathbf{x})) = \frac{\psi(\varepsilon_k(\mathbf{x}))}{\varepsilon_k(\mathbf{x})} \tag{41}$$

the M-estimator can be obtained by an iterative method in which each step involves solving a weighted least squares problem of the form:

$$\widehat{x}_{i+1} = \min_{\mathbf{x}} \sum_{k=1}^{n} w_k(\widehat{x}_i) \; \varepsilon_k^2(\mathbf{x}) \tag{42}$$

At each iteration i the weights w_k are computed with the current estimate $\widehat{x}$ using equation (41) and then *fixed* to find the next best estimate of $\mathbf{x}$ solving problem (42). The ρ-functions for some familiar M-estimators are listed in Table I. Some M-estimators rely on a fixed tuning constant c that depends on an accurate estimate of the scale of the residuals distribution. Most M-estimators use a multiple of the Median of Absolute Deviations (MAD) as a scale estimate which implicitly assumes that the noise contamination rate is 50%. The most common scale estimate used is $\widehat{\sigma} = 1.4826 MAD$ where the 1.4826 factor adjusts the scale for maximum efficiency when the data samples come from a Gaussian distribution. For Tukey we have $c = 4.6851 \; \widehat{\sigma}$ while for Huber we have $c = 1.2107 \; \widehat{\sigma}$.

Type	$\rho(\varepsilon)$	$w(\varepsilon)$	range
L_2	$\frac{1}{2}\varepsilon^2$	1	$\varepsilon \in \mathbb{R}$
L_1	$\lvert\varepsilon\rvert$	$\frac{\operatorname{sign}\varepsilon}{\varepsilon}$	$\varepsilon \in \mathbb{R}$
Huber	$\begin{cases} \frac{1}{2}\varepsilon^2 \\ c\left(\lvert\varepsilon\rvert - \frac{c^2}{2}\right) \end{cases}$	$\begin{cases} 1 \\ c\frac{\operatorname{sign}(\varepsilon)}{\varepsilon} \end{cases}$	$\begin{cases} \lvert\varepsilon\rvert \le c \\ \lvert\varepsilon\rvert > c \end{cases}$
Tukey	$\begin{cases} 1 - \left(1 - \frac{\varepsilon^2}{c^2}\right)^3 \\ 0 \end{cases}$	$\begin{cases} \left(1 - \left(\frac{\varepsilon}{c}\right)^2\right)^2 \\ 0 \end{cases}$	$\begin{cases} \lvert\varepsilon\rvert \le c \\ \lvert\varepsilon\rvert > c \end{cases}$

TABLE I: Different examples of standard M-Estimators.

V. Experiments

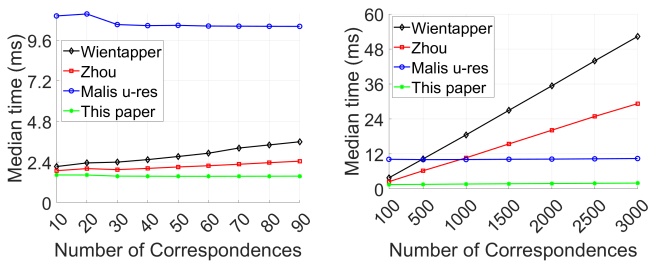
In this section, we compare the proposed algorithm with the state-of-the-art closed-form algorithms: Wientapper's algorithm [15], Zhou's algorithm [16] and Malis's algorithm [17]. Moreover we discuss the robustness of the proposed algorithm when integrated in an iteratively re-weighted least square approach for implementing M-estimators to optimise robust kernels.

A. Experiments with simulated data

The simulated data are generated similarly to [16] and [17]. Specifically, 3D points are determined by randomly sampling a sphere of radius 10m. Unit directions for lines and normal for planes are generated randomly. Random rotations are generated by uniformly sampling the Euler angles φ, θ and ψ (φ, $\psi \in [0°; 360°]$ and $\theta \in [0°; 180°]$). The translation displacements are uniformly distributed within [-10m; 10m]. For n_p point-to-plane, n_l point-to-line, and n_m point-to-point correspondences, the number of correspondences is calculated as $N = n_p + 2n_l + 3n_m$. Given an N, a combination of point, line and point is randomly generated whose effective number is N. The estimated rotation $\widehat{\mathbf{R}}$ and translation $\widehat{\mathbf{t}}$ are compared to the ground truth rotation $\mathbf{R}$ and translation $\mathbf{t}$. The rotation error is the absolute value of the rotation angle computed as $\delta r = ||\text{logm}(\widehat{\mathbf{R}}\ \mathbf{R}^\top)||_F$, where logm is the logarithm of a rotation matrix and $||\bullet||_F$ is the Frobenius norm that gives the magnitude of the rotation angle. The translation error is $\delta t = ||\widehat{\mathbf{t}} - \mathbf{t}||$. The median computation time, the average rotation error $\mu(\delta r)$ and the average translation error $\mu(\delta t)$ are computed for 1000 trials.

1) Performance comparison:

In this simulation, the computational efficiency of closed-form algorithms is evaluated across a spectrum of correspondence values ranging from 10 to to 3000. Figure 1 presents the median runtime results derived from 1000 trials. For the sake of visibility, the results are separated into two figures. In figure (1a) for N range from 10 to 90 while in figure (1b) N range from 100 to 3000. It's worth highlighting that in our current Matlab implementation, the proposed approach consistently surpasses other closed-form techniques, achieving for example a computation time of roughly 1.5 milliseconds for 100 correspondences and approximately 2.0 milliseconds for 3000 correspondences.



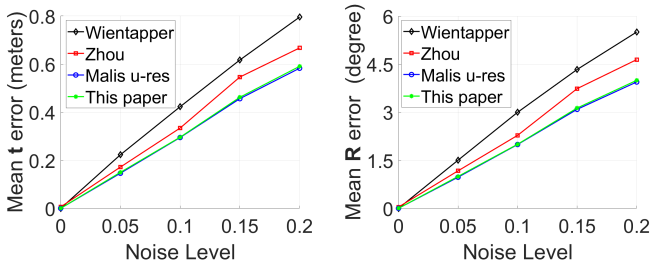
(a) N from 10 to 90 (b) N from 100 to 3000

Fig. 1: Computational time of closed-form algorithms (ms)

2) Accuracy comparison:

In this simulation, the accuracy of closed-form algorithms is assessed, firstly considering a fixed number of correspondences and varying the noise standard deviation on the generated 3D data and then considering a fixed noise standard deviation and varying the number of correspondences.

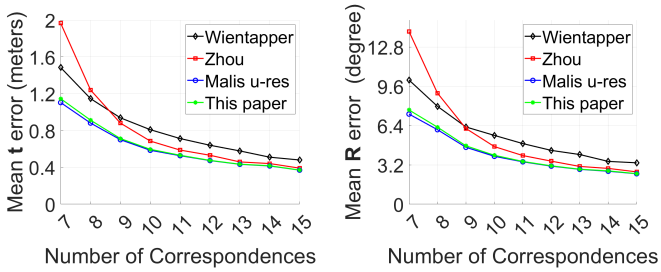
a) Fixed number of correspondences, increasing noise: Figure (2) illustrates the results of a simulation where the number of correspondences is fixed to N = 10. The rotation and the translation are random. The noise standard deviation increases from 0 to 0.2 m. Resultant-based approaches consistently outperform the accuracy of Gröebner-basis approaches. The novel h-resultant approach provide comparable results to the u-resultant approach while being more efficient. The novel "h-resultant" approach and the "u-resultant" approach provide comparable results. However, the "h-resultant" approach (1.5 ms) is 7 times faster than the "u-resultant" approach (10.0 ms).



(a) Translation error (m). (b) Rotation error (degrees).

Fig. 2: Simulation with noise standard deviation increasing from 0 to 0.2 m. The number of correspondences is set to 10.

b) Fixed noise, increasing number of correspondences: Figure (3) illustrates the results of a simulation where the noise standard deviation is set to 0.2 m. The rotation and the translation are random. The number of correspondences increases from 7 to 15. The remarks on the results are very similar to the previous experiment. Again, resultant-based approaches consistently outperform the accuracy of Gröebner-basis approaches. The novel h-resultant approach provide comparable results to the u-resultant approach while being always more efficient.

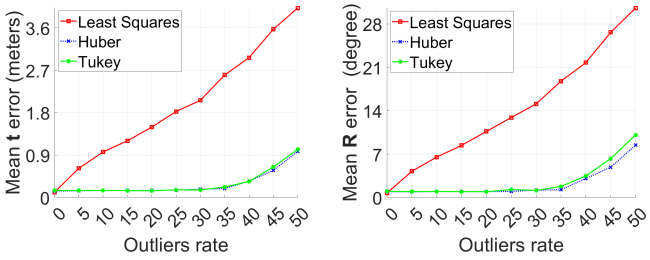


(a) Comparison with closed-form methods, translation error (m). (b) Comparison with closed-form methods, rotation error (degrees).

Fig. 3: Simulation with number of correspondences increasing from 7 to 15. The noise standard deviation is set to 0.2 m.

3) Robustenss to outliers:
The robustness of the closed-form algorithm is assessed using the iteratively reweighted least squares appoach described in Section (IV) setting the maximum number of iterations to 10. The number of correspondences is fixed to 100 and the noise standard deviation to $\sigma = 0.2m$. Some good matches are replaced by outliers with a rate increasing from 0% to 50%.

In the first simulation the outliers induce residuals with low amplitude. Tukey and Huber robust functions works fine reducing the effect that the outliers have on the least squares.

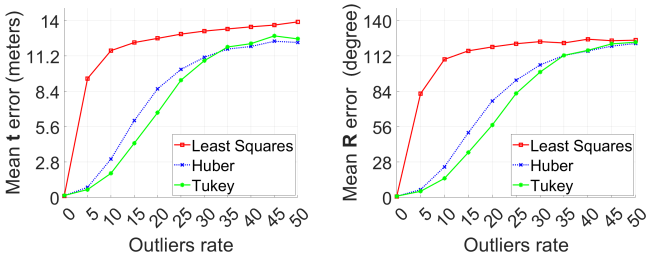


(a) Comparison with robust methods, translation error (m).

(b) Comparison robust methods, rotation error (degrees).

Fig. 4: The outliers induce residuals with low amplitude.

In the second simulation the outliers induce residuals with high amplitude. The Tukey robust function works better since some ouliers with high amplitude are removed.



(a) Comparison with robust methods, translation error (m).

(b) Comparison with robust methods, rotation error (degrees).

Fig. 5: The outliers induce residuals with high amplitude.

B. Experiments with real data

Similarly to previous works [16], [17], the KITTI dataset [28] is used for experimental results. The current 3D points set (lidar scan k+1) is segmented to extract lines and planes and matched with the closest 3D points of the reference set (lidar scan k). There may be more than 50,000 correspondences in each frame, with the majority being the point-to-plane correspondences. For simplicity, only points-to-planes correspondences are considered since they represent more than 99% of the found correspondences. The planes are extracted from the point cloud by performing least squares fitting on k-neighbours (k = 8) around each point. If the least square error is below a threshold the points are considered to be on the same plane. To find the point-to-plane correspondences, a KD-tree search is performed to find the current point closest (minimal 3D distance) to a given reference point. If the current point belong to a plane, the reference point and the the current plane are set as corresponding.

After finding the closest points-to-planes correspondences, the optimal pose between scans k+1 and k is estimated and used to register the reference 3D points into the current frame. A robust Tukey M-estimator [25] is used to compute the weights of the weighted least-square problem in equation (15). This ICP process is repeated at most for 10 iterations or stopped before if the computed incremental translation displacement is less than 1 mm and the computed incremental rotation displacement is less than 0.1 deg. Finally, the optimal pose is compared to the ground truth and a translation error δt (in meters) and a rotation error δR (in degrees) are computed for each frame. Table II show the mean and standard deviation of translation and rotation errors and computation time obtained on sequences 03, 04, and 07 of the KITTI dataset like in [16]. Finally, the proposed approach is not only more accurate and robust than state-of-the-art closed-form approaches but also much faster.

method	Sequence 03 (800 frames)		
	Translation (m)	Rotation (°)	Time (s)
	$\mu(\delta t) \pm \sigma(\delta t)$	$\mu(\delta r) \pm \sigma(\delta r)$	$\mu(t) \pm \sigma(t)$
Wien. [15]	0.020 ± 0.031	0.077 ± 0.126	0.830 ± 0.0373
Zhou [16]	0.018 ± 0.015	0.068 ± 0.090	0.471 ± 0.0091
Malis [17]	0.017 ± 0.012	0.050 ± 0.036	0.029 ± 0.0062
This paper	0.016 ± 0.001	0.053 ± 0.035	0.015 ± 0.0004
method	Sequence 04 (270 frames)		
	Translation (m)	Rotation (°)	Time (s)
	$\mu(\delta t) \pm \sigma(\delta t)$	$\mu(\delta r) \pm \sigma(\delta r)$	$\mu(t) \pm \sigma(t)$
Wien. [15]	0.091 ± 0.255	0.060 ± 0.080	0.871 ± 0.0123
Zhou [16]	0.030 ± 0.021	0.044 ± 0.035	0.458 ± 0.0074
Malis [17]	0.026 ± 0.018	0.036 ± 0.022	0.023 ± 0.0012
This paper	0.020 ± 0.010	0.033 ± 0.017	0.013 ± 0.0004
method	Sequence 07 (1100 frames)		
	Translation (m)	Rotation (°)	Time (s)
	$\mu(\delta t) \pm \sigma(\delta t)$	$\mu(\delta r) \pm \sigma(\delta r)$	$\mu(t) \pm \sigma(t)$
Wien. [15]	0.012 ± 0.007	0.046 ± 0.034	0.860 ± 0.0055
Zhou [16]	0.012 ± 0.007	0.044 ± 0.038	0.477 ± 0.0113
Malis [17]	0.011 ± 0.007	0.041 ± 0.029	0.031 ± 0.0031
This paper	0.011 ± 0.012	0.037 ± 0.027	0.017 ± 0.0022

TABLE II: Comparison with state-of-the-art closed-form algorithm on KITTI Dataset.

VI. Conclusions

This paper presents an efficient approach for pose estimation from points-to-points, points-to-line and points-to plane correspondences. Simulated and experimental results show that the proposed algorithm consistently outperforms state-of-the-art algorithms in term of computational complexity without loosing in precision. While this study provided valuable insights, certain aspects remained unexplored, leaving gaps in our understanding and leaving room for further investigation. Future research directions would be the theoretical study if the optimal computational complexity has been reached or if further significant optimisations are still possible.

Acknowledgements

The author thanks Dr. Lipu Zhou and Dr. Folker Wientapper for providing their Matlab source code for the experimental comparison with their previous works.

REFERENCES

[1] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in *Robotics Science and Systems*, 2014.

[2] A. Comport, E. Malis, and P. Rives, "Real-time quadrifocal visual odometry," *International Journal of Robotic Research*, vol. 29, no. 2-3, 2010.

[3] B. Canovas, M. Rombaut, A. Ngre, D. Pellerin, and S. Olympieff, "Speed and memory efficient dense rgb-d slam in dynamic scenes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[4] P. F. Proenca and Y. Gao, "Probabilistic rgb-d odometry based on points, lines and planes under depth uncertainty," *Robotics and Autonomous Systems*, vol. 104, 2018.

[5] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "Point-plane slam for hand-held 3d sensors," in *IEEE International Conference on Robotics and Automation*, 2013.

[6] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard, "Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[7] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, 1987.

[8] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation*, vol. 3, 1991.

[9] A. Censi, "An icp variant using a point-to-line metric," in *International Conference on Robotics and Automation*, 2008.

[10] C. Olsson, F. Kahl, and M. Oskarsson, "The registration problem revisited: Optimal solutions from points, lines and planes," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006.

[11] C. Olsson and A. Eriksson, "Solving quadratically constrained geometrical problems using lagrangian duality," in *2008 19th International Conference on Pattern Recognition*, 2008.

[12] J. Briales and J. Gonzalez-Jimenez, "Convex global 3d registration with lagrangian duality," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[13] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, "Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group," *The International Journal of Robotics Research*, vol. 38, no. 2-3, 2019.

[14] J. A. Hesch and S. I. Roumeliotis, "A direct least-squares (dls) method for pnp," in *Int Conference on Computer Vision*, 2011.

[15] F. Wientapper, M. Schmitt, M. Fraissinet-Tachet, and A. Kuijper, "A universal, closed-form approach for absolute pose problems," *Computer Vision and Image Understanding*, vol. 173, 2018.

[16] L. Zhou, S. Wang, and M. Kaess, "A fast and accurate solution for pose estimation from 3d correspondences," in *IEEE International Conference on Robotics and Automation*, 2020.

[17] E. Malis, "Complete closed-form and accurate solution to pose estimation from 3d correspondences," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, 2023.

[18] E. Ask, Y. Kuang, and K. strm, "Exploiting p-fold symmetries for faster polynomial equation solving," in *International Conference on Pattern Recognition*, 2012.

[19] Y. Kuang, Y. Zheng, and K. strm, "Partial symmetry in polynomial systems and its applications in computer vision," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[20] V. Larsson, K. Astrom, and M. Oskarsson, "Efficient solvers for minimal problems by syzygy-based reduction," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017.

[21] S. Bhayani, Z. Kukelova, and J. Heikkil, "Computing stable resultant-based minimal solvers by hiding a variable," in *International Conference on Pattern Recognition*, 2021.

[22] D. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*. Springer, 2005.

[23] R. Hartley and H. Li, "An efficient hidden variable approach to minimal-case camera motion estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, 2012.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[25] E. Malis and E. Marchand, "Experiments with robust techniques in real-time robotic vision," in *IEEE/RSJ International Conference on Intelligent Robots Systems*, 2006.

[26] J. H. Peter, *Robust Statistics*. John Wiley & Sons, 1981.

[27] P. W. Holland and R. E. Welsch, "Robust regression using iteratively reweighted least-squares," *Communications in Statistics - Theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.

[28] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, 2013.