Complete closed-form and accurate solution to pose estimation from 3D correspondences

Ezio Malis

Complete closed-form and accurate solution to pose estimation from 3D correspondences

Ezio Malis

***Abstract*— Computing the pose from 3D data acquired in two different frames is of high importance for several robotic tasks like odometry, SLAM and place recognition. The pose is generally obtained by solving a least-squares problem given points-to-points, points-to-planes or points to lines correspondences. The non-linear least-squares problem can be solved by iterative optimization or, more efficiently, in closed-form by using solvers of polynomial systems. In this paper, a complete and accurate closed-form solution for a weighted least-squares problem is proposed. Adding weights for each correspondence allow to increase robustness to outliers. Contrary to existing methods, the proposed approach is complete since it is able to solve the problem in any non-degenerate case and it is accurate since it is guaranteed to find the global optimal estimate of the weighted least-squares problem. Simulations and experiments on real data demonstrate the superior accuracy and robustness of the proposed algorithm compared to previous approaches.**

I. Introduction

The use of 3D sensors (lidars [1], [2], stereo cameras [3], [4], RGB-D cameras [5], [6], ...) has become mainstream in autonomous robotics applications for their ability to accurately localize the robot and measure the 3D structure of the robot's environment. These sensors are able to produce a set of 3D points at each acquisition. Therefore, the registration of two sets of 3D points is of high importance for several tasks like odometry [7], SLAM [8], place recognition [9]. The problem is generally set as the solution of a non-linear optimization problem (generally a least-squares) after matching points to points [10], [11], [12], points to planes [13], [14] or points to lines [15], [16].

A unified solution for the points-to-points, points-to-line and points-to-plane registration problem was firstly proposed in [17]. The authors proposed to use an iterative branch and bound optimization solver [18] that is able to find global optimum of non-convex problems. Such an algorithm is quite slow and usually reach a suboptimal solution i.e. a solution that is close to the global minimum within a given precision. To overcome this problem and speed up the computation time [19] and [20] use a Lagrangian dual relaxation solver. These methods can find the global minimum but are iterative [21] and therefore time consuming. Much faster algorithm can be obtained by using solvers like in [22], [23], [24] that solve the problem in "closed-form" since they make use of simple linear algebra operations only. Most solvers eliminate the translation from the equations since it appears linearly (quadratically in the least-squares problem), obtaining a simpler optimization problem depending from the rotation only (3 d.o.f.). The non-linearities comes from the rotation matrix since it must be an orthonormal matrix. Therefore, the parametrisation of the rotation matrix is crucial to transform the least-square optimisation into a polynomial problem.

The most common rotation parametrisation is obtained by using unit quaternions [10], [23]. This representation is complete but not minimal: all possible rotations can be represented using 4 variables. Another possible representation is the Cayley-Gibbs-Rodrigues (C-G-R) parameters [22], [24]. This representation is minimal but not complete: only 3 variables are necessary but it is impossible to represent rotations with an angle of π around an arbitrary axis. Even if minimal, the C-G-R parametrisation leads to a rational representation of the rotation. Therefore, authors in [24] were finally forced to introduce 4 intermediate variables to transform the rational problem into a polynomial problem. One of the intermediate variables depends on the three others but it was considered as a free variable (therefore ignoring the constraint). Experimental results in [24] shows better accuracy than in [23] but using an additional Newton-Raphson refinement of the solution given by the solver.

In order to understand which is the best combination between solvers and rotation parametrisations in order to have the best accuracy a different approach is proposed in this paper. A quaternion parametrisation is used like in [10], [23] but, contrarily to [23] the unit quaternion constraint is properly imposed through Lagrangian optimization. Moreover, instead of the Gröebner-basis solver, a solver based on the u-resultant MacAulay method [25] is proposed. The method is improved in this paper in order to avoid degenerate configurations that may occur with the u-resultant. The u-resultant MacAulay method was successfully used by [22] to solve the perspective-point pose determination problem (Pnp) but jointly with a C-G-R parametrisation of the rotation. Therefore, the key contribution of this paper is the combination of a robust u-resultant solver with the quaternion parametrisation of the rotation, that was not considered in previous works as illustrated in Table I.

Solver \ Rotation	C-G-R	Quaternion
Gröebner	Zhou [24]	Wientapper [23]
Macaulay	Hesch [22]	This paper

TABLE I: Classification with state-of-the-art closed-form methods depending on the chosen solver and on the chosen rotation parametrisation.

This work was partially supported by the Robocortex S.A.S. company.

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

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:

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

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:

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

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:

$$\mathbf{m}_c = \mathbf{R}\,\mathbf{m}_r + \mathbf{t} \quad (3)$$

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

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

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

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

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

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

we can write:

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

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), equations (3), (4) and (5) can be rewritten as follows:

$$\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m} = 0 \quad (7)$$

For point-to-point correspondences the point $\mathbf{m}$ is set to the current points $\mathbf{m} = \mathbf{m}_c$ in equation (7).

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

$$[\mathbf{d}]_\times^2 (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m}) = 0 \quad (8)$$

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

$$\mathbf{n}^\top (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m}) = 0 \quad (9)$$

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.

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.

1) Cayley-Gibbs-Rodrigues parametrisation: We can use the the Cayley-Gibbs vector parametrisation for $\mathbf{R}$:

$$\mathbf{R}(\mathbf{g}) = \frac{1}{1+\mathbf{g}^\top\mathbf{g}}(\mathbf{I} + 2\,[\mathbf{g}]_\times + 2\,[\mathbf{g}]_\times^2 + (\mathbf{g}^\top\mathbf{g})\mathbf{I}) \quad (10)$$

where $\mathbf{g} = \mathbf{u}\tan\frac{\theta}{2}$. This parametrisation leads to a rational form and is not complete since it cannot represent rotations when $\theta = \pi$ (indeed $\tan\frac{\theta}{2} \to \infty$ when $|\theta| \to \pi$).

2) Quaternion parametrisation: Another rotation parametrisation is the quaternion $\mathbf{q} = [q_r, \mathbf{q}_i] = [w; x; y; z]$ for which we have:

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

$$\mathbf{q}^\top\mathbf{q} = 1 \quad (12)$$

We can stack all the rows of the rotation matrix 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 $\mathbf{r}(\mathbf{q}) = \mathbf{r}(-\mathbf{q})$.

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:

$$\min_{\mathbf{r},\mathbf{t}} \frac{1}{2}\sum_{i=1}^{n_m} w_{mi}^2 d_{mi}^2 + \frac{1}{2}\sum_{j=1}^{n_l} w_{lj}^2 d_{lj}^2 + \frac{1}{2}\sum_{k=1}^{n_p} w_{kp}^2 d_{pk}^2 \quad (13)$$

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

$$d_m^2 = (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m})^\top (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m}) \quad (14)$$

the squared distance point to line is given by:

$$d_l^2 = (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m})^\top\, [\mathbf{d}]_\times^4\, (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m}) \quad (15)$$

and the squared distance point to plane is given by:

$$d_p^2 = (\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m})^\top (\mathbf{n}\mathbf{n}^\top)(\mathbf{M}\,\mathbf{r} + \mathbf{t} - \mathbf{m}) \quad (16)$$

Introducing a (3×3) symmetric weighting matrix $\mathbf{W}$ we can define all possible squared distances 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:

$$\min_{\mathbf{r},\mathbf{t}} \sum_{k=1}^{n} (\mathbf{M}_k\,\mathbf{r} + \mathbf{t} - \mathbf{m}_k)^\top \mathbf{W}_k\, (\mathbf{M}_k\,\mathbf{r} + \mathbf{t} - \mathbf{m}_k) \quad (17)$$

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. The weights can be used to implement robust M-estimators techniques [26].

III. Least square closed-form solution

The first step to solve the problem (17) in closed-form is to eliminate the translation vector that appears quadratically in the cost function and then solve a new equivalent non-linear problem that depends on the rotation only [17], [23], [24]. Again, even if the new equivalent problem is non-linear it can be solved by finding the real roots of polynomial equations.

A. Separating rotation and translation

The least squares problem in equation (17) can be written as a quadratic function of $\mathbf{t}$:

$$c(\mathbf{r},\mathbf{t}) = \mathbf{t}^\top \sum_{k=1}^{n} \mathbf{W}_k \mathbf{t} + 2\sum_{k=1}^{n} \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right)^\top \mathbf{W}_k\ \mathbf{t} + \sum_{k=1}^{n} \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right)^\top \mathbf{W}_k\ \left(\mathbf{M}_k\ \mathbf{r} - \mathbf{m}_k\right) \quad (18)$$

Therefore, the optimal solution for the translation can be easily computed:

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

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 $\mathbf{r}$:

$$\mathbf{t} = \mathbf{A}_t\, \mathbf{r} + \mathbf{b}_t \quad (19)$$

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

$$\mathbf{A}_t = -\left(\sum_{k=1}^{n} \mathbf{W}_k\right)^{-1} \sum_{k=1}^{n} \mathbf{W}_k^\top \mathbf{M}_k \quad (20)$$

$$\mathbf{b}_t = +\left(\sum_{k=1}^{n} \mathbf{W}_k\right)^{-1} \sum_{k=1}^{n} \mathbf{W}_k^\top \mathbf{m}_k \quad (21)$$

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:

$$\overline{\mathbf{M}}_k = \mathbf{M}_k + \mathbf{A}_t \quad (22)$$

$$\overline{\mathbf{m}}_k = \mathbf{m}_k - \mathbf{b}_t \quad (23)$$

we can set the problem in the following canonical form:

$$\min_{\mathbf{r}} c(\mathbf{r}) = \min_{\mathbf{r}} \left(\mathbf{r}^\top \mathbf{A}_r \mathbf{r} + 2\mathbf{b}_r^\top \mathbf{r} + c_r\right) \quad (24)$$

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. Special case of point-to-point correspondences

As noticed by [23] when point-to-point correspondences are used exclusively their in-homogeneous solver GAPS had difficulties. Indeed, for a point-to-point correspondence the matrix $\mathbf{A}_r$ has a very special block-diagonal structure:

$$\mathbf{A}_r = \begin{bmatrix} \mathbf{S} & 0 & 0 \\ 0 & \mathbf{S} & 0 \\ 0 & 0 & \mathbf{S} \end{bmatrix}$$

where $\mathbf{S}$ is a 3×3 symmetric matrix. In this case we have $\forall\ \mathbf{r} = [\mathbf{R}]_\vee$:

$$\mathbf{r}^\top \mathbf{A}_r \mathbf{r} = \frac{1}{3}\,\mathrm{tr}(\mathbf{A}_r)$$

Therefore the total degree of the system of equations point-to-point correspondence drops from 4 to 2.

$$\min_{\mathbf{r}} c(\mathbf{r}) = \min_{\mathbf{r}} \left(2\mathbf{b}_r^\top \mathbf{r} + c_r + \frac{1}{3}\,\mathrm{tr}(\mathbf{A}_r)\right) \quad (25)$$

This explains the difficulties of using the same solver when point-to-point correspondences are used exclusively. In this case, the problem can be rewritten as follows:

$$\min_{\mathbf{q}} c(\mathbf{q}) = \min_{\mathbf{q}} \left(\mathbf{q}^\top \mathbf{B} \mathbf{q} + c_q\right) \quad (26)$$

The cost function is quadratic in the quaternion $\mathbf{q}$, $\mathbf{B}$ being a symmetric (4×4) matrix. Therefore the solution of the problem is straightforward and it is given to the unitary eigenvalue of the symmetric matrix $\mathbf{B}$ corresponding to the smallest eigenvalue [27]. This optimal result can also be obtained by the method proposed in the following section.

C. Closed-form solutions 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 [21]:

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

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:

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

and the following (9×4) Jacobian matrix:

$$\frac{\partial \mathbf{r}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{J}(\mathbf{q}) = 2 \begin{bmatrix} 0 & 0 & -2y & -2z \\ -z & y & x & -w \\ y & z & w & x \\ z & y & x & w \\ 0 & -2x & 0 & -2z \\ -x & -w & z & y \\ -y & z & -w & x \\ x & w & z & y \\ 0 & -2x & -2y & 0 \end{bmatrix}$$

Note that one could compute λ as follows:

$$(\mathbf{A}_r \mathbf{r} + \mathbf{b}_r)^\top \mathbf{J}(\mathbf{q}) \mathbf{q} - \lambda \mathbf{q}^\top \mathbf{q} = 0$$

therefore we could back-substitute λ in the optimization problem:

$$\lambda = (\mathbf{A}_r \mathbf{r} + \mathbf{b}_r)^\top \mathbf{J}(\mathbf{q}) \mathbf{q}$$

This would lead to a system of 5th degree in 4 unknowns. Another way to solve the problem is to consider equation (29):

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

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

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

which is a 4×4 skew symmetric matrix:

$$\mathbf{g}(\mathbf{q}) \wedge \mathbf{q} = \mathbf{q}\, \mathbf{g}^\top(\mathbf{q}) - \mathbf{g}(\mathbf{q})\, \mathbf{q}^\top = 0$$

therefore obtaining the following 6 equations (of degree 4 in the general case and of degree 2 when considering only point-to-point correspondences), jointly with the quaternion constraint of degree 2:

$$f_1(\mathbf{q}) = x^2 + y^2 + z^2 + w^2 - 1 = 0 \quad (30)$$
$$f_2(\mathbf{q}) = x\, g_w - w\, g_x = 0 \quad (31)$$
$$f_3(\mathbf{q}) = y\, g_w - w\, g_y = 0 \quad (32)$$
$$f_4(\mathbf{q}) = z\, g_w - w\, g_z = 0 \quad (33)$$
$$f_5(\mathbf{q}) = x\, g_y - y\, g_x = 0 \quad (34)$$
$$f_6(\mathbf{q}) = x\, g_z - z\, g_x = 0 \quad (35)$$
$$f_7(\mathbf{q}) = y\, g_z - z\, g_y = 0 \quad (36)$$

The Bezout theorem states that the system has at most 128 solutions in the general case and at most 16 solutions when considering only point-to-point correspondences. We use the u-resultant method proposed by [25]. This method has already been used to solve a different problem (the perspective-n-point camera pose estimation) by [22].

The u-resultant approach consist in adding an auxiliary linear equation in the unknown $\mathbf{q}$:

$$f_0(\mathbf{q}) = \mathbf{u}^\top \mathbf{q} = u_0 + u_1\, w + u_2\, x + u_3\, y + u_4\, z \quad (37)$$

where the coefficients $\mathbf{u} = [u_0, u_1, u_2, u_3, u_4]$ can be any. Note that, in general, $f_0(\mathbf{q})$ will not be zero at the roots of the system of polynomial equations (30-36). The second step is to select $n = 4$ independent equations of degree d_i from (30-36) and to define the total degree of the system of equations, including the auxiliary polynomial as $d = 1 + \sum_{i=1}^{n} d_i - n$ (in our case $d = 11$). We multiply the equations f_j will all possible monomial $m^\gamma = w^{\gamma_1} x^{\gamma_2} y^{\gamma_3} z^{\gamma_4}$ with degree $\gamma = \sum_{i=1}^{4} \gamma_i$ such that the obtained polynomials have degree equal to the total degree. Grouping all the monomial in a vector $\mathbf{v} = [1, w, x, y, z, w^2, wx, wy, wz, x^2, ...]$ we can rewrite the obtained polynomials as $\mathbf{M}\ \mathbf{v}$, where $\mathbf{M}$ is the *Macaulay matrix* depending on the coefficients $\mathbf{c}$ of the original polynomials and on the coefficients $\mathbf{u}$ of the auxiliary equation. In our case, we get a matrix $\mathbf{M}$ of size 1365×1365 that can be partitioned as follows [25]:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}(\mathbf{u}) & \mathbf{B}(\mathbf{u}) \\ \mathbf{C}(\mathbf{c}) & \mathbf{D}(\mathbf{c}) \end{bmatrix}$$

where size$(\mathbf{A}) = (128 \times 128)$ and size$(\mathbf{D}) = (1237 \times 1237)$. The solutions of the problem are the eigenvectors of the following 128×128 *multiplication matrix*:

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

In order to reduce the computation time and increase the accuracy three main improvements over the original algorithm [25] are proposed in this paper. First of all, to solve the possible problem of rank deficiency of the $\mathbf{D}$ matrix [28] when building the Macaulay matrix all possible equations (30-36) are considered when selecting polynomials to form the resultant. Secondly, notice that it is possible to reduce the number of monomials by using the equation with the lowest degree. In our case, we substitute $w^2 = 1 - x^2 - y^2 - z^2$ in the equations. Therefore, only monomial linear in w will be present in the equations. The size of matrix $\mathbf{M}$ is then reduced to 650×650, and size$(\mathbf{D}) = (522 \times 522)$. The third simplification is obtained by noticing that $\mathbf{M}$ matrix is generally very sparse (in our case 95% of the matrix entries are equal to 0) and that the matrix may be reordered in such a way that it can be written as a block diagonal matrix:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}(\mathbf{u}) & \mathbf{B}_1(\mathbf{u}) & \mathbf{B}_2(\mathbf{u}) \\ \mathbf{C}_1(\mathbf{c}) & \mathbf{D}_1(\mathbf{c}) & 0 \\ \mathbf{C}_2(\mathbf{c}) & 0 & \mathbf{D}_2(\mathbf{c}) \end{bmatrix}$$

where in our case size$(\mathbf{D}_1) = (222 \times 222)$ and size$(\mathbf{D}_2) = (300 \times 300)$. Finally, the solutions are obtained computing the eigenvectors of the following 128×128 *multiplication matrix* that is much faster to compute:

$$\mathbf{Q} = \mathbf{A} - \mathbf{B}_1\mathbf{D}_1^{-1}\mathbf{C}_1 - \mathbf{B}_2\mathbf{D}_2^{-1}\mathbf{C}_2$$

Obviously we select only the real solutions and discard complex-conjugate solutions. Note also that the possible solutions can be reduced to 64 due to the two-fold symmetry of quaternions [29], [30], [31].

IV. Experiments

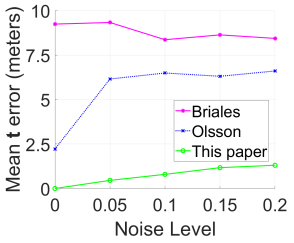
In this section, we compare our algorithm with the state-of-the-art iterative algorithms (Olsson's algorithm using Branch and Bound [18] and Briales's algorithm [20] using Lagrangian duality) and with the state-of-the-art closed-form algorithms (Wientapper's algorithm [23] and Zhou's algorithm [24]). In the experiments with closed-form solvers, only the closed-form solution of the least square problem is considered. Therefore, in order to have a fair comparison between the algorithms, the Newton-Raphson iterative refinement, starting from the best pose found by the closed-form optimization, is not performed as in [24]. Another solution would have been to perform this iterative refinement step after any of the algorithm but this may have hidden the true performance of the solvers. We present experiments both on simulated and real data.

A. Experiments with simulated data

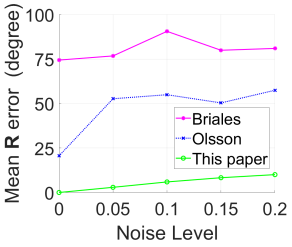
The simulated data are generated similarly to [20] and [24]. Specifically, 3D points are determined by randomly sampling a sphere of radius 10m. Unit directions for lines and normal for planes are generated randomly. Random rotations are generated by uniformly sampling the Euler angles φ, θ and ψ (φ, $\psi \in [0°; 360°]$ 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 [32] that gives the skew-symmetric matrix $[\mathbf{u}\theta]_\times$ such that $\widehat{\mathbf{R}}\ \mathbf{R}^\top = \text{expm}([\mathbf{u}\theta]_\times)$ and $||\bullet||_F$ is the Frobenius norm that gives the magnitude of the rotation angle. The translation error is $\delta t = ||\widehat{\mathbf{t}} - \mathbf{t}||$. For all experiments the average rotation error $\mu(\delta r)$ and the average translation error $\mu(\delta t)$ are computed for 1000 trials.

1) Fixed number of correspondences, increasing noise: In the first set of simulations the number of correspondences is fixed while the noise standard deviation on the generated 3D data increases from 0 to 0.2 m. Figure (1) shows the results of a simulation where the rotation angle is fixed to $\theta = 180°$ and the rotation axis and the translation are random. The number of correspondences is fixed to the minimum $N = 6$. For more clarity, we separate the comparison of the proposed method with closed-form algorithm from the comparison with iterative algorithms that have much more computational complexity. The iterative algorithms show poor results when considering the minimum number of correspondences where multiple global minima may exist. As expected, the method proposed in [24] is highly unstable since the C-G-R parametrisation of the rotation cannot handle the case when the rotation axis is fixed to 180°. The proposed method provide a better accuracy than [23] when the noise level increases.

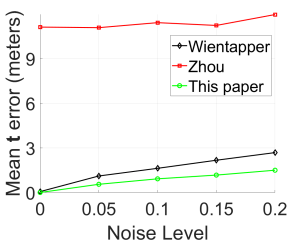
Figure (2) shows the results of a simulation where the rotation and the translation are random. The number of correspondences is again fixed to N=6. For iterative algorithms we obtain similar results to the previous simulation. Note that when the number of correspondences is minimal there may be several possible global minima. The closed-form methods provide all the possible solutions while iterative methods may fail to find them all. Since the rotation angle is random the method proposed by Zhou performs better than in the previous simulation but it clearly has problems to handle the minimal case. Wientapper's algorithm handles correctly this minimal case, the proposed method even better.



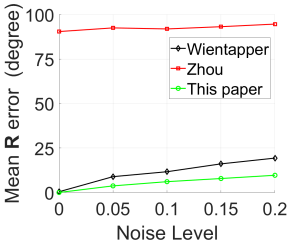
(a) Comparison with iterative methods, translation error (m).



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

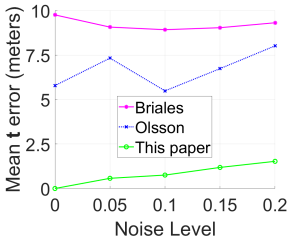


(c) Comparison with closed-form methods, translation error (m).

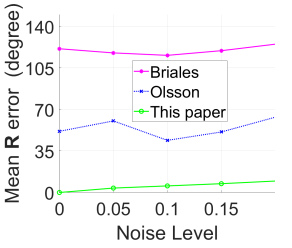


(d) Comparison with closed-form methods, rotation error (degrees).

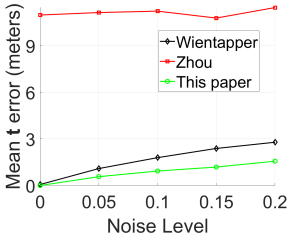
Fig. 1: Simulation with a rotation angle fixed to $\theta = 180°$ around a random rotation axis. The number of correspondences is fixed to the minimum $N = 6$.



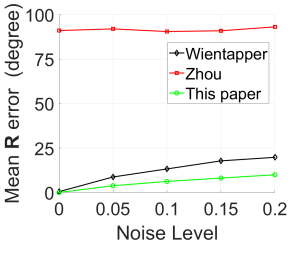
(a) Comparison with iterative methods, translation error (m).



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



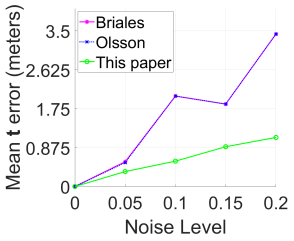
(c) Comparison with closed-form methods, translation error (m).



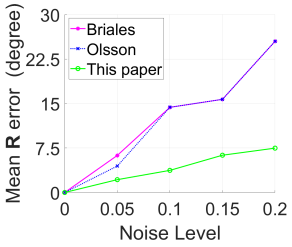
(d) Comparison with closed-form methods, rotation error (degrees).

Fig. 2: Simulation with a random rotation. The number of correspondences is fixed to the minimum $N = 6$.

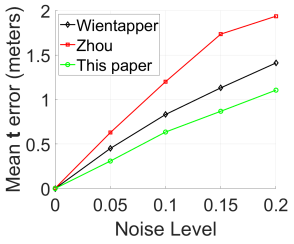
Figure (3) shows the results of a simulation where the rotation angle is fixed to $\theta = 180°$ and the rotation axis and the translation are random. The number of correspondences is fixed to N=7. Contrarily to the simulation with the minimal number of correspondences, iterative methods work well and are almost equivalent. The proposed method provides better results while having a much lower computational complexity. Surprisingly, the method proposed by Zhou gives better results even if the C-G-R parametrisation of the rotation may not handle this case. Noise and numerical errors make that the estimated error is never exactly $180°$ that would cause a division by zero in the algorithm.



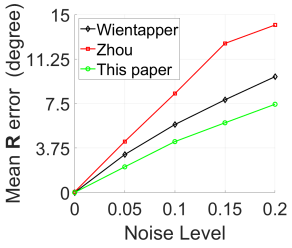
(a) Comparison with iterative methods, translation error (m).



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

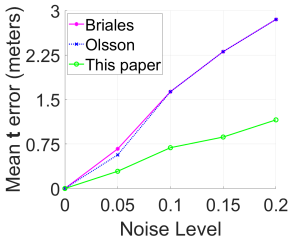


(c) Comparison with closed-form methods, translation error (m).

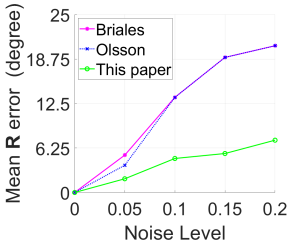


(d) Comparison with closed-form methods, rotation error (degrees).

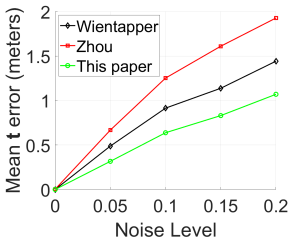
Fig. 3: Simulation with a rotation angle fixed to $\theta = 180°$ around a random rotation axis. The number of correspondences is fixed to $N = 7$.



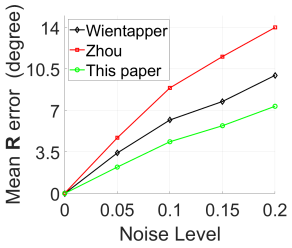
(a) Comparison with iterative methods, translation error (m).



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



(c) Comparison with closed-form methods, translation error (m).

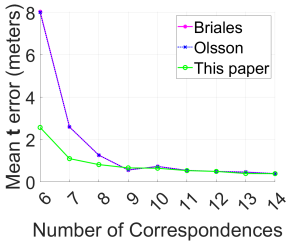


(d) Comparison with closed-form methods, rotation error (degrees).

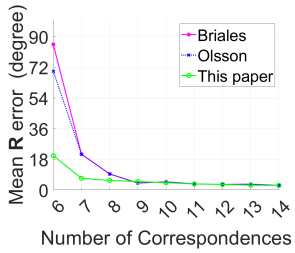
Fig. 4: Simulation with a random rotation. The number of correspondences is fixed to $N = 7$.

Figure (4) shows the results of a simulation where the rotation and the translation are random. The number of correspondences is fixed to N=7. The proposed approach, provides more accurate results than iterative and closed-form methods for all noise levels.

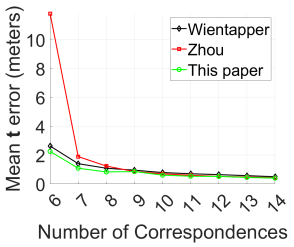
2) Fixed noise, increasing number of correspondences : In the second set of simulations the noise standard deviation is fixed to 0.2 m while the number of correspondences increases from 6 to 14. Figure (5) shows the results when the rotation angle is fixed to $\theta = 180°$ around a random axis.



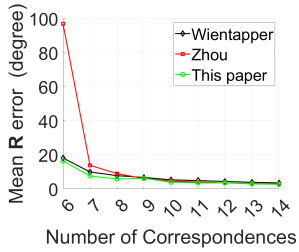
(a) Comparison with iterative methods, translation error (m).



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

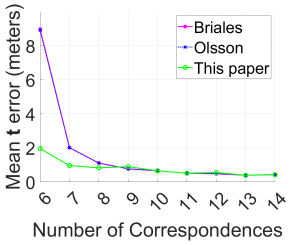


(c) Comparison with closed-form methods, translation error (m).

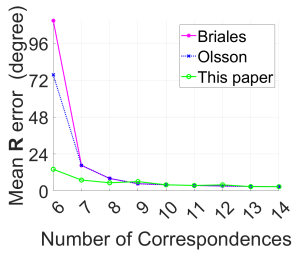


(d) Comparison with closed-form methods, rotation error (degrees).

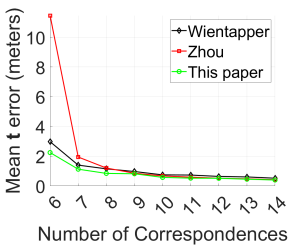
Fig. 5: Simulation with number of correspondences increasing from 6 to 14. The rotation angle is fixed to $\theta = 180°$ and the noise standard deviation is fixed to 0.2 m.



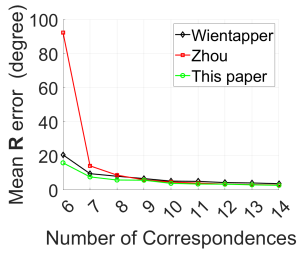
(a) Comparison with iterative methods, translation error (m).



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



(c) Comparison with closed-form methods, translation error (m).



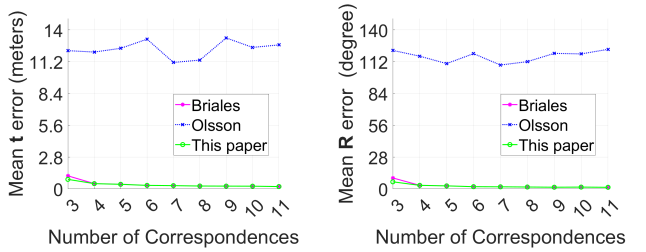
(d) Comparison with closed-form methods, rotation error (degrees).

Fig. 6: Simulation with number of correspondences increasing from 6 to 14. The rotation is random. The noise standard deviation is fixed to 0.2 m.

In this case the C-G-R rotation parametrisation provide the worst results, especially when 6 to 8 correspondences are considered. Note that the 180° degeneracy can be avoided by randomly pre-rotating the points [23]. However, the requirement to run the algorithm multiple times increases the computation time and the number of solutions. Iterative algorithms also provide poor results for few correspondences confirming previous experiments. The proposed algorithm provide the best results over all the algorithms even when only few correspondences are used.

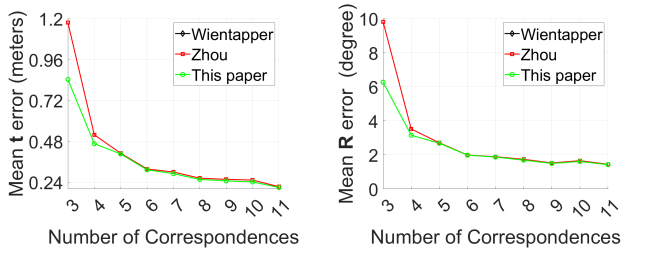
Figure (6) shows the results when the rotation is random. The results are very similar to the previous experiment. In particular, there is no improvement for Zhou's algorithm suggesting again that high level of noise in the data prevents the C-G-R rotation parametrisation from failing. It must be noticed that the difference from the results presented in [24] is certainly due to the fact that the Newton-Raphson refinement is not performed here.

3) Special case with points-to-points correspondences: Simulations with points-to-points correspondences only confirmed that Wientapper's algorithm is not able to handle this special case. In Figure (7) the results corresponding to this method are not displayed since the Matlab function provided by the authors does not give any results. Indeed, the polynomials equations goes from degree 4 to degree 2 (see section III-B). Note that Zhou's algorithm does not suffer from this problem and provide good results. On the contrary, Olsson's algorithm provided poor results. The proposed approach applied to this case provided the best results similarly to Briales's algorithm.



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

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



(c) Comparison with closed-form methods, translation error (m).

(d) Comparison with closed-form methods, rotation error (degrees).

Fig. 7: Simulation with number of 3D point correspondences increasing from 3 to 11. The rotation and translation are random. The noise standard deviation is fixed to 0.2 m.

4) Comparison of computation time: In this experiment, we compare the computational time of closed-form algorithms. The number of correspondences vary from 100 to 3000. Figure 8 illustrates the results averaging the running time over 100 trials. In the current Matlab implementation the proposed approach is slower than the others closed-form methods below 1000 correspondences and faster above (e.g. around 12 milliseconds for 1000 correspondences).

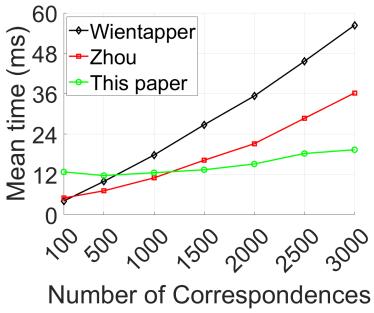


Fig. 8: Computational time of closed-form algorithms (ms)

B. Experiments with real data

Similarly to previous work [24], the KITTI dataset [33] 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 are 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 (i.e. points-to-points and points-to-lines correspondences are only 1% of the found correspondences). The planes are extracted from the point cloud by performing least squares fitting on k-neighbors (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 [26] is used to compute the weights of the weighted least-square problem in equation (13). 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 obtained on sequences 03, 04, and 07 of the KITTI dataset like in [24]. Iterative algorithms are not considered here since they have a much higher computational cost as pointed out in [24]. Finally, the proposed approach is more accurate and robust than state-of-the-art closed-form approaches.

method	Seq. 03 (800 frames)		Seq. 04 (270 frames)		Seq. 07 (1100 frames)	
	$\mu(\delta t) \pm \sigma(\delta t)$ (m)	$\mu(\delta r) \pm \sigma(\delta r)$ (°)	$\mu(\delta t) \pm \sigma(\delta t)$ (m)	$\mu(\delta r) \pm \sigma(\delta r)$ (°)	$\mu(\delta t) \pm \sigma(\delta t)$ (m)	$\mu(\delta r) \pm \sigma(\delta r)$ (°)
Wientapper [23]	0.020 ± 0.031	0.077 ± 0.126	0.091 ± 0.255	0.060 ± 0.080	0.012 ± 0.007	0.046 ± 0.034
Zhou [24]	0.018 ± 0.015	0.068 ± 0.090	0.030 ± 0.021	0.044 ± 0.035	0.012 ± 0.007	0.044 ± 0.038
This paper	0.017 ± 0.012	0.050 ± 0.036	0.026 ± 0.018	0.036 ± 0.022	0.011 ± 0.007	0.041 ± 0.029

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

V. Conclusions

This paper presents a complete and accurate solution for pose estimation from points-to-points, points-to-line and points-to plane correspondences. The proposed approach is also able to provide an accurate and robust solution to the weighted least squares problem even when considering a minimum number of correspondences and for any rotation. Simulated and experimental results show that the proposed algorithm outperforms state-of-the-art algorithms at the price of a higher computational complexity in the current Matlab implementation. Future research directions would be (i) to study if the superior accuracy comes from the choice of solver (Macaulay u-resultant instead of Gröebner basis) or from the correct handling of the quaternion constraint with the Lagrange multiplier, and (ii) to study how to reduce the computational complexity of the solver.

Acknowledgement

This work was partially supported by the Robocortex company. 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] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4758–4765.

[3] A. Comport, E. Malis, and P. Rives, "Real-time quadrifocal visual odometry," *International Journal of Robotic Research*, vol. 29, no. 2-3, pp. 245–266, 2010.

[4] R. Wang, M. Schwrer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *International Conference on Computer Vision*, 2017, pp. 3923–3931.

[5] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, A. Kim, D. abd Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*, 2011.

[6] B. Canovas, M. Rombaut, A. Ngre, D. Pellerin, and S. Olympieff, "Speed and memory efficient dense rgb-d slam in dynamic scenes," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 4996–5001.

[7] P. F. Proen ca and Y. Gao, "Probabilistic rgb-d odometry based on points, lines and planes under depth uncertainty," *Robotics and Autonomous Systems*, vol. 104, p. 2539, 2018.

[8] 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, pp. 5182–5189.

[9] 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, pp. 1249–1255.

[10] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[11] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1987.

[12] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1991.

[13] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation*, vol. 3, 1991, pp. 2724–2729.

[14] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registration," Department of Computer Science, University of North Carolina, Tech. Rep. TR04-004, 2004.

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

[16] F. M. Mirzaei and S. I. Roumeliotis, "Globally optimal pose estimation from line correspondences," in *International Conference on Robotics and Automation*, 2011.

[17] C. Olsson, F. Kahl, and M. Oskarsson, "The registration problem revisited: Optimal solutions from points, lines and planes," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 1206–1213.

[18] ——, "Branch-and-bound methods for euclidean registration problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 783–794, 2009.

[19] C. Olsson and A. Eriksson, "Solving quadratically constrained geometrical problems using lagrangian duality," in *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–5.

[20] J. Briales and J. Gonzalez-Jimenez, "Convex global 3d registration with lagrangian duality," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5612–5621.

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

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

[23] 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, pp. 57–75, 2018.

[24] 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, pp. 1308–1314.

[25] W. Auzinger and H. J. Stetter, *An Elimination Algorithm for the Computation of All Zeros of a System of Multivariate Polynomial Equations*. Singapore: International Series of Numerical Mathematic, 1988, vol. 86, ch. 2, pp. 11–30.

[26] E. Malis and E. Marchand, "Experiments with robust techniques in real-time robotic vision," in *IEEE/RSJ International Conference on Intelligent Robots Systems*, 2006, pp. 223–228.

[27] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, 1996.

[28] J. Canny, "Generalised characteristic polynomials," *Journal of Symbolic Computation*, vol. 9, pp. 241–250, 1990.

[29] E. Ask, Y. Kuang, and K. strm, "Exploiting p-fold symmetries for faster polynomial equation solving," in *International Conference on Pattern Recognition*, 2012, pp. 3232–3235.

[30] 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, pp. 438–445.

[31] 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.

[32] J. R. Cardoso and F. S. Leite, "Exponentials of skew-symmetric matrices and logarithms of orthogonal matrices," *Journal of Computational and Applied Mathematics*, vol. 233, no. 11, pp. 2867–2875, 2010.

[33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, vol. 32, no. 11, pp. 1231–1237, 2013.