Efficient closed-form approaches for pose estimation using Sylvester forms

Jana Vráblíková, Ezio Malis, Laurent Busé

Efficient closed-form approaches for pose estimation using Sylvester forms

Jana Vráblíková, Ezio Malis, Laurent Busé

***Abstract*—Solving non-linear least-squares problem for pose estimation (rotation and translation) is often a time consuming yet fundamental problem in several real-time computer vision applications. With an adequate rotation parametrization, the optimization problem can be reduced to the solution of a system of polynomial equations and solved in closed form. Recent advances in efficient closed form solvers utilizing resultant matrices have shown a promising research direction to decrease the computation time while preserving the estimation accuracy. In this paper, we propose a new class of resultant-based solvers that exploit Sylvester forms to further reduce the complexity of the resolution. We demonstrate that our proposed methods are numerically as accurate as the state-of-the-art solvers, and outperform them in terms of computational time. We show that this approach can be applied for pose estimation in two different types of problems: estimating a pose from 3D to 3D correspondences, and estimating a pose from 3D points to 2D points correspondences.**

I. Introduction

Pose estimation from geometric correspondences is a long-standing problem in computer vision, with classical solutions dating back to early formulations of 3D registration and camera resectioning. A substantial line of research has focused on developing closed-form solvers for various pose estimation settings. Concerning camera resectioning (i.e. the problem of pose computation from 3D points to 2D points correspondences), numerous minimal solvers (the P3p problem) [1]–[4] and non-minimal solvers (the Pnp problem) [5]–[8] have been proposed. Concerning 3D registration (i.e. the problem of pose computation from 3D to 3D correspondences), early solutions include the method proposed in [9], which provide exact solutions for noise-free point-to-point correspondences but do not naturally extend to point-to-line or point-to-plane constraints. More general treatments were later provided through unified optimization frameworks incorporating multiple geometric primitives, such as in [10], which introduced optimal closed-form solutions based on polynomial formulations of the registration problem. Polynomial formulations enabling closed-form solutions have become particularly influential due to their robustness and predictability compared to direct iterative techniques [11]–[14]. These approaches often rely on Gröbner basis methods or resultant constructions to eliminate unknowns and recover camera pose.

Recently, resultant-based solvers have emerged as a powerful solution for constructing efficient closed-form solvers capable of handling mixed 3D to 3D correspondences [13], [14]. These method exploit the algebraic structure of the polynomial systems arising from quaternion-based parametrisations of rotation to solve a least square optimization problem imposing the unit norm constraint on the quaternion with a Lagrangian. Using elimination matrices derived from multivariate resultants, one can obtain fast and accurate closed-form solvers. Despite their effectiveness, these solvers typically require working in high polynomial degrees. Since the larger is the polynomial degrees, the larger is the elimination matrix from which we obtain the solutions, obtaining resultant-based solvers with smaller degree will decrease computational cost.

Concurrently, advances in computational algebraic geometry have introduced more refined elimination techniques. In particular, the theory of Sylvester forms, initially introduced in [15, §3.10], has been recently revisited and further generalized to the multigraded setting in [16], with a view towards applications to the solving of zero-dimensional polynomial systems. It provides new tools for constructing compact elimination matrices with lower algebraic degrees, in comparison with the classical Macaulay elimination matrices. These techniques have shown promising results for reducing the computational cost of polynomial solvers while preserving their algebraic completeness and numerical stability.

The main contribution of our work is to integrate Sylvester forms with the hidden-variable formulation of the resultant in order to obtain new resultant-based methods that operate in degrees 7 and 8, significantly reducing the size of the elimination matrices compared to the degree 9 approach proposed in [14]. We give the theoretical foundations of our approach, relying on the concept of saturation of an ideal, and prove its validity. More specifically, other key contributions of this paper are (i) a detailed analysis of the rank of certain linear systems which allows us to prove the existence of our new elimination matrices (see Proposition 2), and which also explains properties stated in [14] (see Remark 3), (ii) a construction of Sylvester forms tailored to our setting, providing structural results on their coefficients that ease their evaluation (see Lemma 7).

To our knowledge, this is the first application of Sylvester forms to a large variety of pose estimation problems, and the first demonstration that such forms can be used to derive faster, more compact closed-form solvers without sacrificing accuracy. This establishes a new connection between advanced elimination theory and practical computer vision algorithms.

The authors are with Centre Inria d'Université Côte d'Azur, France.
Email:{first name, last name}@inria.fr

II. Theoretical background

A. Pose estimation from 3D to 3D correspondences

The registration of two sets of 3D points is typically formulated as a nonlinear optimisation problem after matching points to points, points to planes or points to lines. The objective of the problem is to estimate the pose (rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$) from measured points and corresponding points, lines and planes. In this section, we briefly review a unified formulation for those problems introduced in [10].

Let $\mathbf{m}_c \in \mathbb{R}^3$ be a point in a current frame $\mathcal{F}_c$. The current point $\mathbf{m}_c$ is obtained from a reference point $\mathbf{m}_r$ in a referent frame $\mathcal{F}_r$ as follows:

$$\mathbf{m}_c = \mathbf{R}\ \mathbf{m}_r + \mathbf{t} \tag{1}$$

where $\mathbf{R} \in \mathbb{SO}(3)$ is a 3×3 rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector.

Stacking the 3 rows $\mathbf{r}_k^\top = [r_{k1}, r_{k2}, r_{k3}]$ ($k \in \{1, 2, 3\}$) of $\mathbf{R}$ into a 9×1 vector

$$\mathbf{r} = [\mathbf{R}]_\vee = [\mathbf{r}_1; \mathbf{r}_2; \mathbf{r}_3] \tag{2}$$

and introducing a 3×9 matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{m}_r^\top & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{m}_r^\top & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{m}_r^\top \end{bmatrix} \tag{3}$$

we can write (1) as

$$\mathbf{m}_c = \mathbf{M}\ \mathbf{r} + \mathbf{t} \tag{4}$$

The point $\mathbf{m}_c$ correspond to a point $\mathbf{m} = \mathbf{m}_r$ in the current frame $\mathcal{F}_c$ if

$$\mathbf{m}_c - \mathbf{m} = \mathbf{M}\ \mathbf{r} + \mathbf{t} - \mathbf{m} = 0 \tag{5}$$

The point $\mathbf{m}_c$ lies on a line in the current frame $\mathcal{F}_c$ with a unit direction vector $\mathbf{d} \in \mathbb{R}^3$ that passes through a point $\mathbf{m} \in \mathbb{R}^3$ if

$$[\mathbf{d}]_\times^2(\mathbf{m}_c - \mathbf{m}) = [\mathbf{d}]_\times^2(\mathbf{M}\ \mathbf{r} + \mathbf{t} - \mathbf{m}) \tag{6}$$

where

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{7}$$

for any vector $\mathbf{v} = [v_1, v_2, v_3]^\top \in \mathbb{R}^3$. Finally, the point $\mathbf{m}_c$ belongs to a plane in the current frame $\mathcal{F}_c$ with unit normal vector $\mathbf{n} \in \mathbb{R}^3$ that passes though a point $\mathbf{m} \in \mathbb{R}^3$ if

$$\mathbf{n}^\top(\mathbf{m}_c - \mathbf{m}) = \mathbf{n}^\top(\mathbf{M}\ \mathbf{r} + \mathbf{t} - \mathbf{m}) = 0 \tag{8}$$

Given n_m point to point correspondences, n_l point to line correspondences and n_p point to plane correspondences, the optimisation problem can be written as

$$\min_{\mathbf{r},\mathbf{t}} \frac{1}{2}\sum_{i=1}^{n_m} w_{m_i}^2 d_{m_i}^2 + \frac{1}{2}\sum_{j=1}^{n_l} w_{l_j}^2 d_{l_j}^2 + \frac{1}{2}\sum_{k=1}^{n_p} w_{p_k}^2 d_{p_k}^2 \tag{9}$$

where w_m, w_l and w_p are weights and d_m, d_l and d_p are point to point, point to line and point to plane distances, respectively. Using a semi-definite weighting matrix $\mathbf{W} \in \mathbb{R}^{3\times3}$, we can write the square distances as follows

$$d^2 = (\mathbf{M}\ \mathbf{r} + \mathbf{t} - \mathbf{m})^\top \mathbf{W} (\mathbf{M}\ \mathbf{r} + \mathbf{t} - \mathbf{m}) \tag{10}$$

where we select the matrix $\mathbf{W}$ according to the type of correspondence,

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

The optimisation problem (9) can be therefore written as

$$\min_{\mathbf{r},\mathbf{t}} \sum_{i=1}^{n} (\mathbf{M}_i\ \mathbf{r} + \mathbf{t} - \mathbf{m}_i)^\top \mathbf{W}_i (\mathbf{M}_i\ \mathbf{r} + \mathbf{t} - \mathbf{m}_i) \tag{11}$$

where $n = n_m + n_l + n_p$.

B. Pose estimation from 3D to 2D correspondences

Another classical problem in computer vision is the estimation of the poste from the projection of 3D points into the image, the Pnp problem:

$$Z_c \mathbf{q}_c = \mathbf{R}\ \mathbf{m}_r + \mathbf{t} \tag{12}$$

where $Z_c = \mathbf{r}_3^\top \mathbf{m}_r + t_3$. Given n_q 3D point to 2D point correspondences, the weighted least squares optimisation problem can be written as:

$$\min_{\mathbf{r},\mathbf{t}} \frac{1}{2}\sum_{i=1}^{n_q} w_{q_i}^2 d_{q_i}^2 \tag{13}$$

where

$$d_q^2 = (\mathbf{M}\ \mathbf{r} + \mathbf{t} - Z_c \mathbf{q}_c)^\top \mathbf{W} (\mathbf{M}\ \mathbf{r} + \mathbf{t} - Z_c \mathbf{q}_c) \tag{14}$$

where $\mathbf{W} = w_q^2\ \mathbf{I}$, for a 3D point to 2D point correspondence. Since $Z_c \mathbf{q}_c = \mathbf{r}_3^\top \mathbf{m}_r + \mathbf{t}$, the optimisation problem (13) can be written as

$$\min_{\mathbf{r},\mathbf{t}} \sum_{i=1}^{n} (\mathbf{P}_i\ \mathbf{r} + \mathbf{Q}_i \mathbf{t})^\top \mathbf{W}_i (\mathbf{P}_i\ \mathbf{r} + \mathbf{Q}_i \mathbf{t}) \tag{15}$$

where $n = n_q$ and:

$$\mathbf{P} = \mathbf{M} - \begin{bmatrix} \mathbf{0}_{3x3} & \mathbf{0}_{3x3} & \mathbf{q}_c\ \mathbf{m}_r^\top \end{bmatrix} \tag{16}$$

$$\mathbf{Q} = \mathbf{I} - \begin{bmatrix} \mathbf{0}_{3x1} & \mathbf{0}_{3x1} & -\mathbf{q}_c \end{bmatrix} \tag{17}$$

C. Reduction to a polynomial problem

We parameterise the rotation by a unit quaternion:

$$\mathbf{q} = [q_r; \mathbf{q}_i] = [w; x; y; z] \tag{18}$$

The rotation matrix $\mathbf{R}$ can be then parameterised as follows:

$$\mathbf{R}(\mathbf{q}) = \mathbf{I} + 2q_r[\mathbf{q}_i]_\times + 2[\mathbf{q}_i]_\times^2 \tag{19}$$

where $\mathbf{q}^\top\mathbf{q} = 1$. Therefore, the vector $\mathbf{r}(\mathbf{q}) = [\mathbf{R}]_\vee$ is quadratic in the variables w, x, y, z.

The translation vector $\mathbf{t}$ can be eliminated from the equation (11) as shown in [10]. Similarly, it can be eliminated from the equation (15) as shown in [8]. Therefore, we can solve a new equivalent problem that depends only on the four variables

w, x, y, z. The new optimisation problem can be written in the following form:

$$\mathbf{q} = \operatorname{argmin}\{c(\mathbf{r}(\mathbf{q}))\} = \operatorname{argmin}\{\mathbf{r}^\top \mathbf{A}_r \mathbf{r} + 2\mathbf{b}_r^\top \mathbf{r} + c_r\} \quad (20)$$

subject to

$$\mathbf{q}^\top \mathbf{q} = 1 \quad (21)$$

where $\mathbf{A}_r \in \mathbb{R}^{9\times 9}$ is a symmetric matrix, $\mathbf{b}_r \in \mathbb{R}^{9\times 1}$ and $c_r \in \mathbb{R}$. Note that for the Pnp problem we have $\mathbf{b}_r = 0$ and $c_r = 0$.

We can impose the constraint (21) using the Lagrange multiplier method:

$$\min_{\lambda,\mathbf{q}} \mathcal{L}(\lambda, \mathbf{q}) = c(\mathbf{r}(\mathbf{q})) + \lambda(1 - \mathbf{q}^\top \mathbf{q}) \quad (22)$$

The solutions of (20) can be then obtained as solutions of the following polynomial system

$$\frac{\partial \mathcal{L}}{\partial \lambda} = 1 - \mathbf{q}^\top \mathbf{q} = 0 \quad (23)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}} = \mathbf{g}^\top(\mathbf{q}) - \lambda \mathbf{q}^\top = 0 \quad (24)$$

where $\mathbf{g}(\mathbf{q}) = [g_w(\mathbf{q}); g_x(\mathbf{q}); g_y(\mathbf{q}); g_z(\mathbf{q})]$ is the following 4×1 vector of degree 3 in $\mathbf{q}$:

$$\mathbf{g}(\mathbf{q}) = \left(\frac{\partial c(\mathbf{r})}{\partial \mathbf{r}} \frac{\partial \mathbf{r}(\mathbf{q})}{\partial \mathbf{q}} \right)^\top \quad (25)$$

We note that $\mathbf{g}(\mathbf{q})$ can be written as

$$\mathbf{g}(\mathbf{q}) = \mathbf{g}_3(\mathbf{q}) + \mathbf{g}_1(\mathbf{q}) \quad (26)$$

where $\mathbf{g}_3(\mathbf{q})$ is of homogeneous degree 3 in $\mathbf{q}$ and $\mathbf{g}_1(\mathbf{q})$ is linear in $\mathbf{q}$. Substituting $\mathbf{q}^\top \mathbf{q} = 1$ into equation (26) yields 4 polynomial equations which depend linearly on λ and are homogeneous of degree 3 with respect to $\mathbf{q}$:

$$\begin{aligned} \mathbf{e}(\mathbf{q}, \lambda) &= \mathbf{g}_3(\mathbf{q}) + (\mathbf{q}^\top \mathbf{q})\mathbf{g}_1(\mathbf{q}) - \lambda(\mathbf{q}^\top \mathbf{q})\mathbf{q} \\ &= \hat{\mathbf{g}}(\mathbf{q}) - \lambda(\mathbf{q}^\top \mathbf{q})\mathbf{q} = 0 \end{aligned} \quad (27)$$

The equations $\mathbf{e}(\mathbf{q}, \lambda)$ also depend linearly on a coefficient vector

$$\mathbf{c} = [c_{1,1}, c_{1,2}, \ldots, c_{4,20}]^\top \quad (28)$$

namely the coefficients of $\hat{\mathbf{g}}(\mathbf{q})$, and we sometimes write

$$\mathbf{e}(\mathbf{q}, \lambda, \mathbf{c}) = \mathbf{e}(\mathbf{q}, \lambda) \quad (29)$$

to emphasise the coefficients.

The problem can be therefore reduced into the problem of finding real solutions of the polynomial system

$$e_1(\mathbf{q}, \lambda) = e_2(\mathbf{q}, \lambda) = e_3(\mathbf{q}, \lambda) = e_4(\mathbf{q}, \lambda) = 0 \quad (30)$$

i.e., finding real points of the variety $\mathrm{V}(I) \subseteq \mathbb{P}^3 \times \mathbb{A}^1$ defined by the ideal

$$I = (e_1, e_2, e_3, e_4) \subset \mathbb{C}[\lambda][w, x, y, z]$$

which is graded with respect to the variables $\mathbf{q}$.

D. Eliminating λ

It turns out that the projection of $V(I)$ on $\mathbb{P}^3$, i.e. the elimination of the parameter λ from the equations (27), can be easily described. Indeed, taking exterior product we get

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

which is equivalent to the condition

$$\operatorname{rank} \begin{pmatrix} \hat{g}_w & \hat{g}_x & \hat{g}_y & \hat{g}_z \\ w & x & y & z \end{pmatrix} < 2 \quad (32)$$

where

$$[\hat{g}_w(\mathbf{q}); \hat{g}_x(\mathbf{q}); \hat{g}_y(\mathbf{q}); \hat{g}_z(\mathbf{q})] = \hat{\mathbf{g}}(\mathbf{q})$$

Therefore we obtain six polynomial equations, namely the $2\times$ 2-minors of the above matrix,

$$f_i(\mathbf{q}) = 0, \ i = 1, \ldots, 6 \quad (33)$$

that are homogeneous of degree 4 in $\mathbf{q}$. We notice that these equations depend linearly on the coefficients $\mathbf{c}$ of the equations (29), and we write $\mathbf{f}(\mathbf{q}, \mathbf{c})$ to emphasise the coefficients.

Denote by J the ideal of $\mathbb{C}[w, x, y, z]$ generated by the polynomials $f_i(\mathbf{q})$, so that $V(J) \subset \mathbb{P}^3$. For all $i = 1, \ldots, 6$ we have $f_i \in I$ (e.g. $x\hat{g}_w - w\hat{g}_x = xe_1 - we_2$), so that any point in $V(I)$ yields a point in $V(J)$. Conversely, given a point in $V(J)$ such that $\mathbf{q}^\top \mathbf{q} \neq 0$, there is a unique λ satisfying equations (27) at this point (observe that $\mathbf{q} \neq 0$ at any point of $\mathbb{P}^3$), hence a unique point in $V(I)$.

E. Closed-form solution via elimination matrices

In [14], a method to find the solutions of equations (27) based on the *hidden variable* approach (see [17, Chapter 3, §5]) is proposed; in this section, we briefly review it. The variable λ is considered as "hidden", that is to say that the polynomials (24) are seen as polynomials in $\mathbf{q}$, λ being interpreted as a parameter.

Notation 1. *For a given degree $d \in \mathbb{N}$, we denote the vector of homogeneous monomials of degree $d \in \mathbb{N}$ in w, x, y, z by*

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

The number of such monomials is $n_d = \binom{d+3}{d}$.

First, $4 \cdot n_6 = 4 \cdot 84 = 336$ equations constructed:

$$\mathbf{e}(\mathbf{q}, \lambda, \mathbf{c}) \otimes \mathbf{m}_6 = \mathbf{E}_9(\mathbf{c}, \lambda)\ \mathbf{m}_9 = 0 \quad (34)$$

where $\mathbf{E}_9(\mathbf{c}, \lambda)$ is a 336×220 coefficient matrix. For a general choice of λ, $\operatorname{rank}(\mathbf{E}_9(\mathbf{c}, \lambda)) = 220$.

Moreover, additional $6 \cdot n_5 = 6 \cdot 56 = 336$ equations that do not depend on λ are considered:

$$\mathbf{f}(\mathbf{q}, \mathbf{c}) \otimes \mathbf{m}_5 = \mathbf{F}_9(\mathbf{c})\ \mathbf{m}_9 = 0 \quad (35)$$

where $\mathbf{F}_9(\mathbf{c})$ is a 336×220 coefficient matrix and $\operatorname{rank}(\mathbf{F}_9(\mathbf{c})) = 180$.

From matrices $\mathbf{E}_9(\mathbf{c}, \lambda)$ and $\mathbf{F}_9(\mathbf{c})$, a 220×220 matrix $\mathbf{M}_9$ admitting the following structure is obtained (see Remark 3):

$$\mathbf{M}_9 = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_0 & \mathbf{D}_0 \end{bmatrix} - \lambda \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (36)$$

where the matrix $[\mathbf{A}\ \mathbf{B}]$ consists of a subset of rows of the matrix $\mathbf{E}_9(\mathbf{c}, \lambda)$, the matrix $[\mathbf{C}\ \mathbf{D}]$ consists of a subset of rows of the matrix $\mathbf{F}_9(\mathbf{c})$. In particular, $\text{size}(\mathbf{A}) = 40 \times 40$, $\text{size}(\mathbf{B}) = 40 \times 180$, $\text{size}(\mathbf{C}) = 180 \times 40$ and $\text{size}(\mathbf{D}) = 180 \times 180$, and furthermore, $\text{rank}(\mathbf{M}_9) = 220$, $\text{rank}(\mathbf{A}) = 40$, and $\text{rank}(\mathbf{D}) = 180$ for a general choice of λ.

The solution of the pose estimation problem is then to find λ such that

$$\mathbf{M}_9\ \mathbf{m}_9 = 0 \tag{37}$$

which is possible if

$$\det(\mathbf{M}_9) = 0 \tag{38}$$

This leads to solving the generalised eigenvalue problem

$$\det(\mathbf{Q}_0 + \lambda \mathbf{Q}_1) = 0 \tag{39}$$

where

$$\mathbf{Q}_i = \mathbf{A}_i - \mathbf{B}_i \mathbf{D}_0^{-1} \mathbf{C}_0,\ i = 0, 1 \tag{40}$$

We notice that the matrix $\mathbf{M}_9$ is built in such a way that its rows are filled by means of polynomial equations of degree 9 belonging to the ideal I; this is obvious for (34) and for (35) it follows from the fact that $f_i \in I$ for all $i = 1, \ldots, 6$. In what follows we construct similar matrices $\mathbf{M}_7$ and $\mathbf{M}_8$ corresponding to degrees 7 and 8 respectively. However, the extension to those degrees of the method described above is not straightforward: polynomial equations belonging to the ideal I (and hence J) are not enough and it is necessary to introduce new equations.

III. More efficient closed-form solutions using Sylvester forms

A. Number of solutions

The instances of the problem we are considering have finitely many (complex) solutions. In this section, we provide the number of points in $\mathbb{P}^3_{\mathbb{C}}$ defined by the ideal J (see Section II-D). It turns out that this counting is closely related to the analysis of the rank of matrices similar to (35), built in arbitrary degrees.

We recall that the ideal $J \subset \mathbb{C}[w, x, y, z]$ is generated by the polynomials (33) which are homogeneous polynomials. Therefore, J is a graded ideal and we denote by J_ν its graded component of degree ν, for all $\nu \in \mathbb{Z}$.

Given an integer $d \geq 4$, we consider the matrix $\mathbf{F}_d(c)$ built from $6 \cdot n_{d-4}$ polynomials that form a basis of the vector space $(J)_d$:

$$\mathbf{f}(\mathbf{q}, \mathbf{c}) \otimes \mathbf{m}_{d-4} = \mathbf{F}_d(\mathbf{c})\ \mathbf{m}_d^T \tag{41}$$

The matrix $\mathbf{F}_d(\mathbf{c})$ is a $(6 \cdot n_{d-4}) \times n_d$ coefficient matrix.

Proposition 2. *If $V(J) \subset \mathbb{P}^3_{\mathbb{C}}$ is finite, then it consists of 40 points counted with multiply. In addition,* $\text{rank}(\mathbf{F}_d(\mathbf{c})) = n_d - 40$ *if and only if $d \geq 7$.*

Proof. Set $R := \mathbb{C}[w, x, y, z]$. The proof relies on the analysis of the resolution of the quotient ring R/J by graded free R-modules (we refer to [17, Chapter 5 and 6] for an introduction to these concepts). Since $V(J)$ is finite, the ideal J is a determinantal ideal (i.e. it is defined by minors of a matrix; see (32)) which has maximal depth, here 3. As a consequence, it admits the following free resolution (known as Eagon–Northcott resolution; see [18, Theorem A.2.60]):

$$0 \to R(-10) \oplus R(-8) \oplus R(-6) \to R(-7)^4 \oplus R(-5)^4 \to R(-4)^6 \xrightarrow{\psi} R \tag{42}$$

where the notation $R(k)$, $k \in \mathbb{Z}$, denotes a shift in the grading: $R(k)_\nu = R_{k+\nu}$ for all integers ν and k.

The map ψ in (42) is defined by the generators (33) of J, which are of degree 4. Therefore, the transpose of $\mathbf{F}_d(\mathbf{c})$ is a matrix of the graded component

$$\psi_d : R_{d-4} \to R_d \tag{43}$$

of ψ in suitable monomial bases (namely $\mathbf{m}_d$ for the rows and $6\mathbf{m}_{d-4}$ for the columns). Now, the cokernel of ψ_d is equal to the Hilbert function of R/J in degree d (see [18, Chapter 1]). As J is a defining ideal of points in $\mathbb{P}^3_{\mathbb{C}}$ (observe that J is saturated because it has a free resolution of length 3), the Hilbert function of R/J is equal to the Hilbert polynomial of R/J if and only if d is greater or equal to the Castelnuovo-Mumford regularity of R/J (see [18, Theorem 4.2]). In view of (42), the regularity of R/J is equal to $10-3 = 7$. Moreover, the Hilbert polynomial of R/J is a constant which is equal to the number of points in $\mathbb{P}^3_{\mathbb{C}}$, counted with multiplicity, defined by J. In our setting, it is equal to the quantity

$$n_d - 6n_{d-4} + (4n_{d-5} + 4n_{d-7}) - (n_{d-6} + n_{d-8} + n_{d-10}) \tag{44}$$

for all $d \geq 7$, which is equal to 40. □

Remark 3. *Proposition 2 shows that the matrix $\mathbf{M}_9$ in* (36) *is indeed of rank $180 = 220 - 40$. This is a key property in Section II-E because it proves the claimed structure* (36) *of the matrix $\mathbf{M}_9$. Indeed, since equations* (33) *are contained in the ideal I, equations* (35) *can be found in equations* (34) *by linear operations on rows.*

B. Saturation and Sylvester forms

The closed-form solution reviewed in Section II-E is based on the equations (34) which correspond to the graded component I_9 of degree 9 of the graded ideal $I \subset R_\lambda := \mathbb{C}[\lambda][w, x, y, z]$ (the grading is with respect to the four variables w, x, y, z; λ being a parameter, it is considered to be of degree 0). To explain the choice of the degree 9, we need to introduce the ideal obtained from I by saturation with respect to the ideal $\mathfrak{m} := (w, x, y, z)$.

Definition 4. *The saturation of the graded ideal I with respect to $\mathfrak{m} := (w, x, y, z)$ is the ideal*

$$I^{\text{sat}} := \{p \in R_\lambda \ \ \textit{such that}\ \ \exists n \in \mathbb{N} : \mathfrak{m}^n p \subset I\} \tag{45}$$

Clearly $I \subset I^{\text{sat}}$. Moreover I and I^{sat} are equal after inversion in R_λ of w, x, y or z, which means that $V(I)$ and $V(I^{\text{sat}})$ are equal, including their local algebraic structures (e.g. multiplicities). In particular $(I)_d = (I^{\text{sat}})_d$ for $d \gg 0$.

Now, suppose given an integer d and consider the matrix $\mathbf{E}_d(\mathbf{c}, \lambda)$, built similarly to (34). We expect two properties for this matrix in order to solve the polynomial system (30):

- it is of (full) rank n_d for general values of $\mathbf{c}$ and λ,
- it is not full rank for some given $\mathbf{c}$ and λ if and only if the corresponding polynomial system has solutions in $\mathbb{P}^3_{\mathbb{C}}$.

It is a known result in elimination theory that these two properties hold for any d such that $(I)_d = (I^{\mathrm{sat}})_d$ (see e.g. [19, Theorem 3.20]). In addition, since I is generated by 4 equations of degree 3 in w, x, y, z, this latter property holds if $d \geq 4(3-1)+1 = 9$ (see e.g. [16, Lemma 2.2]).

To overcome the limitation $d \geq 9$ to obtain matrices $\mathbf{E}_d(\mathbf{c}, \lambda)$ with the expected properties, it is necessary to introduce new equations. Our strategy is to take those equations in I^{sat} so that the solution set $V(I)$ is unchanged. In addition, since we are targeting closed-form solutions, those equations must be given in closed-form in the coefficients $\mathbf{c}$. For that purpose, we will use Sylvester forms that have been initially introduced in [15] (see also [16, §2.10]).

Given $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4) \in \mathbb{N}^4$, we set $|\alpha| = \sum_{i=1}^4 \alpha_i$ and we define

$$\mathbf{q}^\alpha := [w^{\alpha_1+1}, x^{\alpha_2+1}, y^{\alpha_3+1}, z^{\alpha_4+1}] \tag{46}$$

Since the polynomials e_1, e_2, e_3 and e_4 are homogeneous of degree 3 with respect to $\mathbf{q}$, for any α such that $|\alpha| < 3$ it is possible to find decompositions

$$e_i = [h_{i,1}, h_{i,2}, h_{i,3}, h_{i,4}]\ (\mathbf{q}^\alpha)^T,\ i = 1, \ldots, 4 \tag{47}$$

where $h_{i,j} = h_{i,j}(\mathbf{q}, \lambda) \in \mathbb{C}[\lambda][w, x, y, z]$ are homogeneous polynomials of degree $3 - \alpha_j - 1$ in $\mathbf{q}$.

Definition 5. *For any $\alpha \in \mathbb{N}^4$ such that $|\alpha| < 3$, the determinant*

$$S_\alpha = \det\left(\begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} \\ h_{3,1} & h_{3,2} & h_{3,3} & h_{3,4} \\ h_{4,1} & h_{4,2} & h_{4,3} & h_{4,4} \end{bmatrix}\right) \tag{48}$$

is called a Sylvester form (of $e_1, \ldots, e_4$ with respect to $\mathbf{q}$).

By construction, Sylvester forms depend on $\mathbf{q}, \lambda$ and $\mathbf{c}$. More specifically, S_α is homogeneous of degree $8 - |\alpha|$ in $\mathbf{q}$, homogeneous of degree 4 in $\mathbf{c}$ and is of degree at most 4 in λ (notice that it is not homogeneous with respect to λ). Also, by construction Sylvester forms belong to I^{sat}.

Although Sylvester forms depend on decompositions (47), which are not unique, they are essentially unique in the following sense.

Proposition 6. *For any α such that $|\alpha| < 3$, the class of S_α modulo I is independent of the choices of decompositions* (47)*. Moreover, for any integer $\nu \in \{0, 1, 2\}$, the set*

$$\{S_\alpha \text{ such that } |\alpha| = \nu\} \tag{49}$$

form a basis of the free $\mathbb{C}[\lambda, \mathbf{c}]$-module $(I^{\mathrm{sat}}/I)_{8-\nu}$.

Proof. We refer to [16, Proposition 2.11] and the references therein. □

As a consequence of the above result, Sylvester forms can be added to the ideal I to obtain a new ideal that have the same saturation but being itself saturated in a smaller degree. In what follows, we exploit this property to build new closed-form solutions to our problem.

C. Closed-form solution in degree 8

Consider the ideal I' generated by the equations (30) and a Sylvester from $S_0 := S_{(0,0,0,0)}$, i.e. $I' := I + (S_0)$. According to Proposition 6,

$$(I')_d = (I^{\mathrm{sat}})_d = (I')_d^{\mathrm{sat}} \quad \text{for all } d \geq 8 \tag{50}$$

Therefore, as explained in Section III-B, the matrix $\mathbf{E}'_8(\mathbf{c}, \lambda)$ built from a basis of $(I')_8$ will have the expected properties. This matrix is actually the matrix $\mathbf{E}_8(\mathbf{c}, \lambda)$ to which an additional row corresponding to S_0 is added.

In practice, a key ingredient is the choice made to compute S_0. Its degree with respect to λ is of particular importance. To proceed, we consider a decomposition of the polynomials e_1, e_2, e_3, e_4 (corresponding to rows) with respect to $\mathbf{q}^{(1,1,0,0)} = [w^2, x^2, y, z]$ such that:

$$\begin{pmatrix} p_{1,1} - \lambda w & p_{1,2} - \lambda w & p_{1,3} - \lambda wy & p_{1,4} - \lambda wz \\ p_{2,1} - \lambda x & p_{2,2} - \lambda x & p_{2,3} - \lambda xy & p_{2,4} - \lambda xz \\ p_{3,1} - \lambda y & p_{3,2} - \lambda y & p_{3,3} - \lambda y^2 & p_{3,4} - \lambda yz \\ p_{4,1} - \lambda z & p_{4,2} - \lambda z & p_{4,3} - \lambda yz & p_{4,4} - \lambda z^2 \end{pmatrix} \tag{51}$$

where the $p_{i,j}$'s are polynomials in $\mathbf{q}$ and $\mathbf{c}$. The determinant of the above matrix is actually a Sylvester form $S_{(1,1,0,0)}$, which we denote by S_{wx}. Using classical rules of determinants, it appears that S_{wx} is linear in λ. Moreover, a similar decomposition with respect to $\mathbf{q}^{(0,0,0,0)} = [w, x, y, z]$ is easily obtained from (51) by multiplication of the first and second columns by w and x respectively. Therefore, by Definition 5, we may take $S_0 := wxS_{wx}$. It follows that S_0 can be written as

$$p^{(8,4)}(\mathbf{q}, \mathbf{c}) - \lambda p^{(8,3)}(\mathbf{q}, \mathbf{c}) \tag{52}$$

where $p^{(8,k)}$ is a polynomial of homogenous degree 8 in $\mathbf{q}$ and of homogenous degree k in $\mathbf{c}$, with $k = 3, 4$. The following result is important in practice for the efficiency of the evaluation of S_0.

Lemma 7. *With the above notation, we write*

$$p^{(8,k)} = \sum_{j=1}^{n_8} u_{j,k}\ m_j \quad \textit{with } m_j \in \mathbf{m}_8 \tag{53}$$

and we assume that the decomposition (51) *is chosen such that for all $j = 1, \ldots, 4$, the polynomials $p_{1,j}, \ldots, p_{4,j}$ have the same monomial supports (i.e. the same set of monomials with nonzero coefficient). Then, the coefficients $u_{j,k}$ are $k \times k$ minors of the matrix*

$$\mathbf{C} = \begin{bmatrix} c_{1,1} & c_{2,1} & \cdots & c_{1,20} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,20} \\ c_{3,1} & c_{3,2} & \cdots & c_{3,20} \\ c_{4,1} & c_{4,2} & \cdots & c_{4,20} \end{bmatrix} \tag{54}$$

of the coefficients of the equations (27).

Proof. We first assume that $\lambda = 0$ in the polynomials $e_1, \ldots, e_4$ and hence in the decomposition (51). Consider the action of the special linear group $\mathrm{SL}(4, \mathbb{C})$ (4×4 matrices with determinant equal to 1) on the matrix C given by matrix multiplication (on the left). By our assumptions on the choices of $p_{i,j}$, this action also corresponds to left multiplication on the matrix (51). Therefore, the determinant of this latter matrix is invariant under the action of $\mathrm{SL}(4, \mathbb{C})$. Applying the First Fundamental Theorem of Invariant Theory (see e.g. [20, Section 3.2]), we deduce that it is a polynomial in the 4×4-minors of (54) (these minors are themselves invariant under this action and actually generate the ring of invariants under this action).

Now, the case where λ is nonzero can be obtained from the previous case by substituting some coefficients $c_{i,j}$ by $c_{i,j} - \lambda$. We deduce that the determinant S_{wx} is a polynomial of the 4×4-minors of (54) after such substitutions. Expanding these minors, we obtain 4×4-minors of the plain matrix (54), then λ multiplied by 3×3-minors of the plain matrix (54) and terms depending on λ^2 up to λ^4. However, taking into account that S_{wx} is linear in λ, all the terms that are not linear in, or independent of, λ must cancel. □

We are now ready to state a new method to find the solutions of the system of equations (27). From a basis of $(I)_8$ we get $4 \cdot 56 = 224$ equations

$$\mathbf{e} \otimes \mathbf{m}_5 = \mathbf{E}_8(\mathbf{c}, \lambda)\ \mathbf{m}_8^T = 0 \tag{55}$$

The matrix $\mathbf{E}_8(\mathbf{c}, \lambda)$ is of size 224×165. We add to it a single row $\mathbf{S}$ from (53), i.e. such that $S_0 = \mathbf{S}\mathbf{m}_8^T$, to get the matrix $\mathbf{E}'_8(\mathbf{c}, \lambda)$ of size 225×165. In this way, the rows of $\mathbf{E}'_8$ correspond to a basis of $(I')_8$. From (50), we deduce that $\mathbf{E}'_8$ has full rank 165 for a general choice of λ.

From equations (33), we get the $6 \cdot 35 = 210$ equations

$$\mathbf{f} \otimes \mathbf{m}_4 = \mathbf{F}_8(\mathbf{c})\ \mathbf{m}_8^T = 0 \tag{56}$$

where $\mathrm{size}(\mathbf{F}_8(\mathbf{c})) = 210 \times 165$. By Proposition 2, $\mathrm{rank}(\mathbf{F}_8(\mathbf{c})) = 125$. Moreover, we already noticed that the rows of $\mathbf{F}_8$ can be obtained in $\mathbf{E}_8$ by linear operations on rows. Therefore, we proved that there exists a matrix $\mathbf{M}_8$ of the form

$$\mathbf{M}_8 = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_0 & \mathbf{D}_0 \end{bmatrix} - \lambda \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{57}$$

where the matrix $[\mathbf{A}\ \mathbf{B}]$ consists of 39 rows of the matrix $\mathbf{E}_8(\mathbf{c}, \lambda)$ and the row $\mathbf{S}$. The matrix $[\mathbf{C}\ \mathbf{D}]$ consists of 125 rows of the matrix $\mathbf{F}_8(\mathbf{c})$. In particular, $\mathrm{size}(\mathbf{A}) = 40 \times 40$, $\mathrm{size}(\mathbf{B}) = 40 \times 125$, $\mathrm{size}(\mathbf{C}) = 125 \times 40$ and $\mathrm{size}(\mathbf{D}) = 125 \times 125$. Furthermore, $\mathrm{rank}(\mathbf{M}_8) = 165$, $\mathrm{rank}(\mathbf{A}) = 40$, and $\mathrm{rank}(\mathbf{D}) = 125$ for a general choice of λ.

The solutions of the system (27) can be obtained by solving the generalised eigenvalue problem, following the procedure described in (37)-(40).

D. Closed-form solution in degree 7

Following the same strategy as in the previous section, we develop a closed-form solution by using Sylvester forms of degree 7 in $\mathbf{q}$. We choose four Sylvester forms S_α, for all α such that $|\alpha| = 1$, and consider the ideal I'' defined as the ideal I to which these four Sylvester forms are added. According to Proposition 6,

$$(I'')_d = (I^{\mathrm{sat}})_d = (I'')_d^{\mathrm{sat}} \quad \text{for all } d \geq 7 \tag{58}$$

In practice, to choose appropriately our four Sylvester forms, we first compute two Sylvester forms $S_{(1,0,1,0)}$ and $S_{(0,1,0,1)}$ of degree 6 using decompositions similar to (51) and then we set

$$S_w := yS_{(1,0,1,0)},\ S_x := zS_{(0,1,0,1)},\\ S_y := wS_{(1,0,1,0)},\ S_z := xS_{(0,1,0,1)} \tag{59}$$

The polynomials S_w, S_x, S_y, S_z are of homogenous degree 7 in $\mathbf{q}$, homogeneous of degree 4 in the coefficients $\mathbf{c}$ and linear in λ. Thus, we can write them similarly to (52); for instance

$$S_w(\mathbf{q}, \lambda) = p_w^{(7,4)}(\mathbf{q}, \mathbf{c}) - \lambda p_w^{(7,3)}(\mathbf{q}, \mathbf{c}) \tag{60}$$

where $p_w^{(7,k)}$ is a homogeneous polynomial of degree 7 in $\mathbf{q}$ and homogenous of degree k in $\mathbf{c}$ for $k = 3, 4$. Similar expressions hold for S_x, S_y and S_z. Lemma 7 also applies here so that the coefficients of these polynomials are $k \times k$ minors of the coefficient matrix $\mathbf{C}$, which improves the practical efficiency of their evaluation.

Putting all the above ingredients together, we obtain another closed-form solution to find the solutions of the system of equations (27). From a basis of $(I)_7$, we construct $4 \cdot 35 = 140$ equations

$$\mathbf{e} \otimes \mathbf{m}_4 = \mathbf{E}_7(\mathbf{c}, \lambda)\ \mathbf{m}_7^T = 0 \tag{61}$$

The matrix $\mathbf{E}_7(\mathbf{c}, \lambda)$ is of size 140×120. We add to it four rows obtained with the coefficients of S_w, S_x, S_y and S_z by means of expressions (60). We obtain the matrix $\mathbf{E}''_7(\mathbf{c}, \lambda)$ of size 144×120 which has full rank 120 for a general choice of λ (which is ensured by (58)). Now, equations (33) yields $6 \cdot 20 = 120$ equations

$$\mathbf{f} \otimes \mathbf{m}_3 = \mathbf{F}_7(\mathbf{c})\ \mathbf{m}_7^T = 0 \tag{62}$$

where $\mathrm{size}(\mathbf{F}_7(\mathbf{c})) = 120 \times 120$ and $\mathrm{rank}(\mathbf{F}_7(\mathbf{c})) = 80$ by Proposition 2. It follows that one can construct a matrix $\mathbf{M}_7$ of the form

$$\mathbf{M}_7 = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 \\ \mathbf{C}_0 & \mathbf{D}_0 \end{bmatrix} - \lambda \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{63}$$

where the matrix $[\mathbf{A}\ \mathbf{B}]$ consists of a 36 rows of the matrix $\mathbf{E}_7(\mathbf{c}, \lambda)$ and 4 rows of the coefficients of the polynomials S_w, S_x, S_y, S_z. The matrix $[\mathbf{C}\ \mathbf{D}]$ consists of 80 rows of the matrix $\mathbf{F}_7(\mathbf{c})$. In particular, $\mathrm{size}(\mathbf{A}) = 40 \times 40$, $\mathrm{size}(\mathbf{B}) = 40 \times 80$, $\mathrm{size}(\mathbf{C}) = 80 \times 40$ and $\mathrm{size}(\mathbf{D}) = 80 \times 80$. Furthermore, $\mathrm{rank}(\mathbf{M}_7) = 120$, $\mathrm{rank}(\mathbf{A}) = 40$, and $\mathrm{rank}(\mathbf{D}) = 80$ for a general choice of λ.

The solutions of the system (27) can be obtained by solving the generalised eigenvalue problem, following the procedure described in (37)-(40).

E. Constructing matrices $\boldsymbol{Q}_0$ and $\boldsymbol{Q}_1$

To solve the generalized eigenvalue problem, we need to construct matrices $\mathbf{Q}_0$ and $\mathbf{Q}_1$, see (40). In [14], these matrices are obtained by computing Schur complements of the matrix $\mathbf{M}_9$, which is composed of fixed subsets of rows of the coefficient matrices $\mathbf{E}_9$ and $\mathbf{F}_9$. This fast construction, however, can fail if the block $\mathbf{D}$ of the matrix $\mathbf{M}_9$ (see (36)) is not invertible. Here, we propose a procedure for the construction of the matrices $\mathbf{Q}_0$ and $\mathbf{Q}_1$, which outputs matrices with the best conditioning for a given choice of the monomial order. To unify the notation, here we set $\mathbf{E}'_7 := \mathbf{E}''_7$.

Algorithm 1: Optimized selection of rows and Schur complements

Input: Coefficient matrices $\mathbf{E}'_d$ and $\mathbf{F}_d$, monomials $\mathbf{m}_d$ $(d = 7, 8)$
Output: Matrices $\mathbf{Q}_0$ and $\mathbf{Q}_1$

1 Fix a monomial order on $\mathbf{m}_d$ and reorder the columns of $\mathbf{E}'_d$ and $\mathbf{F}_d$ (the first 40 monomials index the columns of $\overline{\mathbf{A}_0}, \overline{\mathbf{A}_1}$ and $\overline{\mathbf{C}_0}$)
2 Define blocks such that $[\overline{\mathbf{A}_0}\ \overline{\mathbf{B}_0}] + \lambda[\overline{\mathbf{A}_1}\ \overline{\mathbf{B}_1}] = \mathbf{E}'_d$ and $[\overline{\mathbf{C}_0}\ \overline{\mathbf{D}_0}] = \mathbf{F}_d$
3 Compute the QR decomposition $\overline{\mathbf{D}_0} = \mathbf{Q}_D\mathbf{R}_D$
4 Set $\mathbf{C}_0 = \mathbf{Q}_D^T\overline{\mathbf{C}}$ and $\mathbf{D}_0 = \mathbf{Q}_D^T\overline{\mathbf{D}} = \mathbf{R}_D$
5 Compute Schur complements $\overline{\mathbf{Q}_i} = \overline{\mathbf{A}_i} - \overline{\mathbf{B}_i}\mathbf{D}_0^{-1}\mathbf{C}_0$ for $i = 0, 1$
6 Compute the QR decomposition $\mathbf{Q}_Q\mathbf{R}_Q = \overline{\mathbf{Q}_1}$
7 Let $\mathbf{Q}_0 = \mathbf{Q}_Q^T\overline{\mathbf{Q}_0}$ and $\mathbf{Q}_1 = \mathbf{Q}_Q^T\overline{\mathbf{Q}_1} = \mathbf{R}_Q$

Remark 8. *Both matrices $\overline{\mathbf{D}_0}$ and $\overline{\mathbf{Q}_1}$ are tall matrices, i.e., they have more rows than columns. The* economical *QR decomposition (e.g.* qr($\overline{\mathbf{D}_0}$,"econ") *in Matlab) outputs an orthonormal matrix $\mathbf{Q}$ and a square, upper triangular matrix $\boldsymbol{R}$ with non-zero entries on the diagonal. The latter one can be inverted efficiently and is reused in the later steps of the procedure.*

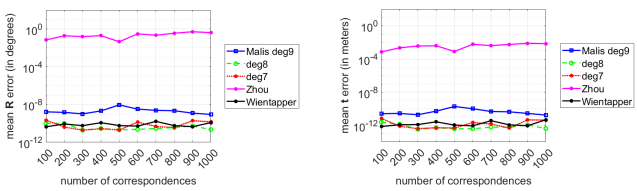
IV. Simulations and Experimental Results

We apply the methods to simulated and real data, comparing them in terms of accuracy and computational time with the state-of-the-art methods for the pose estimation problem from 3D to 3D as well as from 3D to 2D correspondences.

The solving process can be divided into three steps:

1) Derivation of the problem in the "canonical" form (20).
2) Computation of the solutions of the polynomial system given by equations (23) and (24).
3) Selection of the minimal solution, i.e. of the solution for which (20) is minimal.

The computational time presented in the results only reflects the time necessary for the second step, because the times needed for the first and final steps are identical for all the presented methods.



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

Fig. 1: Number of correspondences increases from 100 to 1000, the noise standard deviation is set to 0 m.

A. Pose estimation from 3D to 3D correspondences

We compare the proposed methods with the state-of-the-art closed-form h-resultant based algorithm by Malis [14] and the algorithms by Wientapper et al. [11] and by Zhou, Wang, and Kaess [12] (without the Newton-Raphson iterations to refine the results).

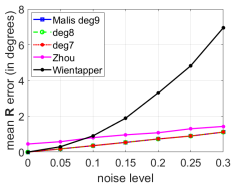
1) Experiments with simulated data: The data are generated using the simulation described in [14]. In particular, 3D points are randomly sampled on a 10m radius sphere. For lines and planes, unit direction vectors and normal vectors are generated randomly. Rotations are generated by uniformly sampling the Euler angles ϕ and θ, where $\phi, \theta \in [0, 2\pi]$ and $\theta \in [0, \pi]$. The translation vectors are uniformly distributed within the range $[-10\text{m}, 10\text{m}]$.

For n_m point-to-point, n_l point-to-line, and n_p point-to-plane correspondences, we denote $N = 3n_m + 2n_l + n_p$. Given N, a combination of point-to-point, point-to-line and point-to-plane correspondences is randomly generated.

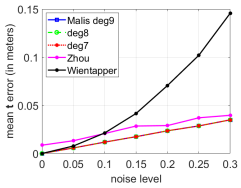
We compare the estimated rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{t}}$ to the ground truth rotation $\mathbf{R}$ and translation $\mathbf{t}$. The rotation error δr is the absolute value of the rotation angle computed as $\delta r = \|\log(\hat{\mathbf{R}}\mathbf{R}^T)\|_F$, where log denotes the logarithm of a rotation matrix and $\|\cdot\|_F$ denotes the Frobenius norm that gives the magnitude of the rotation angle. The translation error is $\delta t = \|\hat{\mathbf{t}} - \mathbf{t}\|$. The median computation time, the average rotation error $\mu(\delta r)$ and the average translation error $\mu(\delta t)$ are computed for 1000 trials.

Accuracy comparison. In the first simulation, we do not add any noise to the simulated data in order to check the numerical sensitivity of the algorithms. Figure 1 presents the average rotation and average translation errors for an increasing number of correspondences. In the absence of noise, the proposed methods in degree 7 and 8 are numerically more stable than the degree 9 method, perform similarly to [11] and significantly outperform [12]. Figure 2 shows the average rotation and translation error for 1000 correspondences and an increasing level of noise. Figure 3 presents the average rotation and translation error for increasing number of correspondences. The noise standard deviation is set to $\sigma = 0.2$m. In the simulations shown in Figures 2 and 3, the proposed methods perform as well as the algorithm from [14] and outperform methods [12] and [11].

Computational time comparison. In the proposed methods, it is necessary to evaluate the coefficients of the polynomials S_i.

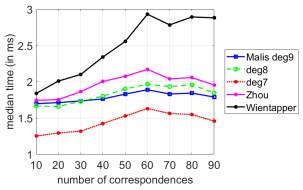


Rotation error (degrees).

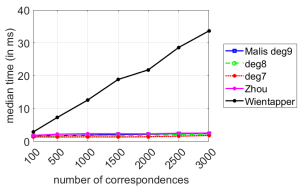


Translation error (m).

Fig. 2: Noise standard deviation increases from 0 to 0.3 m, the number of correspondences is set to 1000.

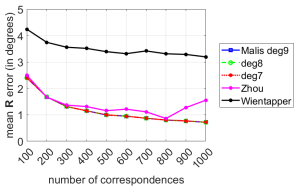


N from 10 to 90.

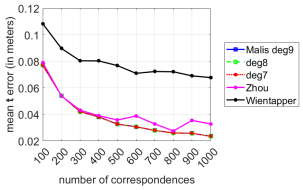


N from 100 to 3000.

Fig. 4: Comparison of computational times.



Rotation error (degrees).



Translation error (m).

Fig. 3: Number of correspondences increases from 100 to 1000, the noise standard deviation is set to 0.2 m.

Experimentally, we observed that there is no notable difference in terms of time or numerical stability for different choices of the Sylvester forms of degree 6 and the decompositions (47) of the polynomials (33).

Figure 4 shows the median computational time for N varying between 10 and 3000, with $\sigma = 0.2$m. For the sake of visibility, the results are separated into two figures. Our method with $d = 7$ outperforms the method by Malis [14], which uses a fixed subset of rows to construct the matrix $\mathbf{M}_9$. Both proposed methods are faster that the method [12] and [11]. We used a Matlab wrapper of the C++ implementation of the algorithm [11] provided by the authors, and Matlab implementations of the other methods.

2) Experiments with real data: Similarly to the previous works [12], [14], the KITTI dataset [21] is used for the experimental evaluation. The current set of 3D points (LiDAR scan $k+1$) is segmented to extract planar structures and matched with the closest 3D points from the reference set (LiDAR scan k). Planes are detected in the point cloud by performing a least-squares fitting on the 8-nearest neighbors of each point. If the resulting least-squares residual is below a predefined threshold, the neighboring points are considered to belong to the same plane. Point-to-plane correspondences are then obtained using an Iterative Closest Point (ICP) strategy, where the estimated pose is ignored and only the correspondences are retained. Once the point-to-plane correspondences have been determined, 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 is employed to compute the weights of the weighted least-squares problem derived from (11). Finally, the pose estimated with state-of-the-art approaches and the methods proposed in this paper are compared with the ground truth, and a translation error δt (in meters) and a rotation error δr (in degrees) are computed for each frame.

Table I shows the mean and standard deviation of translation and rotation errors and computation time obtained on sequences 03, 04, and 07 of the KITTI dataset. The proposed methods outperform the other methods in terms of both accuracy and computational time.

KITTI sequence 03 (800 frames)			
method	rotation (°) $\mu(\delta r) \pm \sigma(\delta r)$	translation (m) $\mu(\delta t) \pm \sigma(\delta t)$	time (ms) $\mu(t) \pm \sigma(t)$
Malis [14]	0.2741±6.3519	0.0237±0.2125	3.3211±0.9909
Wientapper [11]	0.0521±0.0365	0.0163±0.011	62.893±12.574
Zhou [12]	0.0746±0.6813	0.018±0.0481	2.9928±0.9365
deg8	**0.0493±0.0162**	**0.0181±0.0115**	3.0317±0.744
deg7	**0.0493±0.0162**	**0.0181±0.0115**	**2.5318±0.4278**

KITTI sequence 04 (270 frames)			
method	rotation (°) $\mu(\delta r) \pm \sigma(\delta r)$	translation (m) $\mu(\delta t) \pm \sigma(\delta t)$	time (ms) $\mu(t) \pm \sigma(t)$
Malis [14]	**0.0329±0.0191**	**0.0163±0.0093**	3.3927±0.8963
Wientapper [11]	0.0373±0.0274	0.0165±0.0094	66.125±13.96
Zhou [12]	0.5892±9.1232	0.0755±0.9701	2.8593±1.0025
deg8	**0.0329±0.0191**	**0.0163±0.0093**	2.9417±0.5468
deg7	**0.0329±0.0191**	**0.0163±0.0093**	**2.4667±0.4254**

KITTI sequence 07 (1100 frames)			
method	rotation (°) $\mu(\delta r) \pm \sigma(\delta r)$	translation (m) $\mu(\delta t) \pm \sigma(\delta t)$	time (ms) $\mu(t) \pm \sigma(t)$
Malis [14]	1.1871±14.299	0.0866±0.9645	3.5282±0.9402
Wientapper [11]	0.0467±0.0391	0.0121±0.07	49.726±16.979
Zhou [12]	0971±1.0427	0.0136±0.0297	2.8219±0.6593
deg8	**0.0428± 0.0335**	**0.012±0.007**	3.498±1.0466
deg7	**0.0428± 0.0335**	**0.012±0.007**	**2.5303±0.7546**

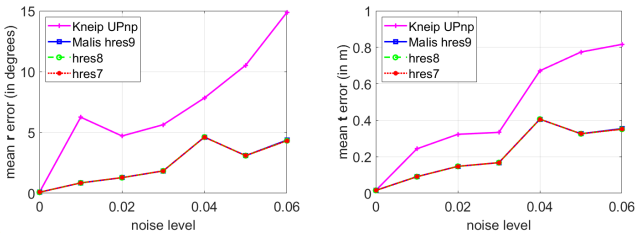
TABLE I: Comparison on KITTI datasets.

B. Pose estimation from 3D to 2D correspondences

The proposed methods can also be applied to the Pnp problem after derivation of the "canonical form" (20). We compare them with the following solvers:

UPnp [8] is an efficient solver that does not use the Lagrangian to solve the constrained least square problem. The problem solved by the UPnP is equivalent to set $\lambda = 0$ in (22). This reduces the number of solutions and hence the computational time. However, setting $\lambda = 0$ is a valid approximation only if the noise is (close to) zero.

Similarly, optDLS [22] and SRPnP [23] do not recover all the possible solutions. Furthermore, these methods use Cayley parameterization of the rotation (optDLS, SRPnP) do not perform well for rotations with an angle close to π around any axis (see [24]).



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

Fig. 5: Pnp problem on ETH3D dataset with increasing noise.

SQPnP [25] is an iterative method. Unlike closed-form methods, there is no theoretical guarantee that the method will reach the global minimizer.

OPnP [24] uses quaternion representation of the rotation and therefore is stable for rotation angles close to π and it finds all 40 solutions. Our method finds the same number of solutions with the same accuracy, but is about 5 times faster.

1) Experiments with simulated data: In Table II, we compare the proposed methods with more solvers. It shows the average and maximal rotation and translation error and the average computation time over 10000 trials. We set the number of correspondences $N = 10$ and the noise standard deviation $\sigma = 1$ pixel.

To further test the accuracy of the proposed methods, we used images from the ETH3D dataset [26]. The dataset contains original images, the coordinates of 3D points and the corresponding 2D points, the pose of cameras associated with each image and the intrinsic parameters of the cameras. We compare the proposed methods with the algorithm from [14] and the UPnp method [8] on the 25 available datasets. Figure 5 shows the mean rotation and translation error averaged across all datasets. In each dataset, we chose one image and 10 correspondences, where we perturbed the 2D and 3D ground truth points by increasing noise. Methods solving the least squares problem exactly, including ours, show better accuracy with increasing noise. The proposed approaches were implemented in Matlab and are slower than the UPnp method. Indeed, we used the UPnp implementation in C available in the OpenGV library [27]. However, more accurate approaches may be useful in non-real time application.

2) Experiments with real data: To test our methods on real data, we again used the ETH3D dataset [26]. The dataset contains 25 sequences of images, together with the coordinates of 3D points and the corresponding 2D points, the pose of cameras associated with each image, and the intrinsic parameters of the cameras. For every sequence, we chose one reference image $\mathbf{im}_0^i$, where $i = 1, \ldots, 25$. In every other image $\mathbf{im}_j^i$ of the i-th sequence, we matched points with points in the reference image. The preimages of the matched points in the reference image $\mathbf{im}_0^i$ are the 3D points $\mathbf{m}_r$ from equation (12). In image $\mathbf{im}_j^i$, we filter out matched points, that are farther than 5 pixels from the true matches. The remaining points are the image points $\mathbf{p}_c$ in the image $\mathbf{im}_j^i$. We estimated the pose $\hat{\mathbf{R}}, \hat{\mathbf{t}}$, and compared it with the ground truth pose $\mathbf{R}, \mathbf{t}$, such that equation (12) holds.

We did not consider images $\mathbf{im}_j^i$ that do not overlap with the reference image $\mathbf{im}_0^i$, nor those with fewer than 4 correspondences.

We compared the proposed methods with the algorithm by Malis [14] and the UPnp method by Kneip, Li and Seo [8] (without the final Newton step). Table III shows the rotation and translation error averaged over the 25 sequences. Tables IV and V show the average rotation and translation error (respectively) for a given number of correspondences. The results are again averaged over the 25 sequences. All the compared algorithms improve with growing number of correspondences. The resultant-based methods outperform the state-of-the-art UPnp method and, for low number of correspondences, the proposed deg8 and deg7 methods often perform better than the deg9 algorithm by Malis.

V. Conclusion

In this work, we showed how to integrate Sylvester forms in resultant-based methods and proved the validity of our approach. We obtained new resultant-based methods that operate in degrees 7 and 8, significantly reducing the size of the elimination matrices. This has a significant impact on the computation time outperforming previous approaches.

An important open question concerns the selection of the monomial ordering, which affects the conditioning of the blocks of the elimination matrix from which the solutions are computed. Fixing the block construction in advance is computationally more efficient, whereas selecting it online can improve numerical accuracy. Understanding how to choose this ordering optimally remains an interesting direction for future work.

References

[1] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, 2003.

[2] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Colorado Springs, USA, June 2011.

[3] T. Ke and S. I. Roumeliotis, "An efficient algebraic solution to the perspective-three-point problem," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4618–4626.

[4] Y. Ding, J. Yang, V. Larsson, C. Olsson, and K. Åström, "Revisiting the p3p problem," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[5] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *Int Journal on Computer Vision*, vol. 81, pp. 155–166, 2008.

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

[7] Y. Zheng, Y. Kuang, S. Sugimoto, K. Åstrom, and M. Okutomi, "Revisiting the pnp problem: A fast, general and optimal solution," in *International Conference on Computer Vision (ICCV)*, 2013.

[8] L. Kneip, H. Li, and Y. Seo, "Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability," in *European Conference on Computer Vision (ECCV)*, 2014.

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

	rotation (°)		translation (m)		time (ms)
	$\mu(\delta r) \pm \sigma(\delta r)$	$\max(\delta \mathrm{r})$	$\mu(\delta t) \pm \sigma(\delta t)$	$\max(\delta \mathrm{t})$	$\mu(t) \pm \sigma(t)$
UPnP	0.254 ± 0.630	33.07	0.005 ± 0.007	0.380	0.734 ± 0.332 (in C)
optDLS	0.265 ± 1.604	71.61	0.008 ± 0.099	5.029	2.000 ± 0.578
OPnP	0.207 ± 0.105	0.843	0.004 ± 0.003	0.021	11.75 ± 2.747
SRPnP	0.214 ± 0.155	10.79	0.004 ± 0.003	0.133	0.759 ± 0.409
SQPnP	0.629 ± 6.373	177.8	0.014 ± 0.159	3.190	0.085 ± 0.546 (in C++)
deg9	0.207 ± 0.102	0.844	0.004 ± 0.003	0.021	2.561 ± 1.004
deg8	0.207 ± 0.102	0.844	0.004 ± 0.003	0.021	2.417 ± 0.856
deg7	0.207 ± 0.102	0.844	0.004 ± 0.003	0.021	2.204 ± 0.768

TABLE II: Comparison to other PnP solvers.

	rotation (°)	translation (m)
	$\mu(\delta r) \pm \sigma(\delta r)$	$\mu(\delta t) \pm \sigma(\delta t)$
Kneip UPnp	6.9055 ± 18.2656	0.80875 ± 2.8761
Malis deg9	2.1398 ± 5.9209	0.44969 ± 1.2829
deg8	2.2221 ± 6.3222	**0.4363 ± 1.2553**
deg7	**2.1263 ± 5.8971**	0.46607 ± 1.3665

TABLE III: Comparison of the developed methods with UPnp method by Kneip, Li and Seo [8], and the deg9 method by Malis [14]. The results are averaged over all sequences in the ETH3D dataset.

n	Kneip UPnp	Malis deg9	deg8	deg7
4	35.1584	20.0303	18.4840	**15.3291**
5	45.7671	21.0113	20.7705	**18.9686**
6	53.4829	14.4955	14.4299	**14.0446**
7	44.3037	**10.0844**	11.0800	10.1060
8	28.4090	**13.0101**	13.0107	**13.0101**
9	24.9650	7.8337	7.8338	**7.8331**
10	2.2087	0.7268	**0.7267**	0.7268
11 - 25	12.4584	**4.0925**	**4.0925**	**4.0925**
26 - 50	10.7784	**2.7428**	**2.7428**	**2.7428**
51 - 100	11.5497	**1.9820**	**1.9820**	**1.9820**
101 - 200	4.3847	**0.3016**	**0.3016**	**0.3016**
$\geq$ 200	8.7278	**0.5582**	**0.5582**	**0.5582**

TABLE IV: Mean rotational error (in degrees) for given number n of correspondences. The results are averaged over all sequences in the ETH3D dataset.

n	Kneip UPnp	Malis deg9	deg8	deg7
4	8.2631	6.6374	6.8478	**3.7874**
5	1.5255	3.5584	3.3815	**2.7821**
6	11.5245	2.4866	2.4848	**2.4597**
7	1.4194	**2.0547**	2.1253	2.0578
8	0.6908	1.4533	**1.4531**	1.4533
9	5.8250	**1.9149**	**1.9149**	**1.9149**
10	0.6147	**0.1549**	**0.1549**	**0.1549**
11 - 25	12.4584	**4.0925**	**4.0925**	**4.0925**
26 - 50	1.0293	**0.6142**	**0.6142**	**0.6142**
51 - 100	0.8434	**0.1951**	**0.1951**	**0.1951**
101 - 200	4.3847	**0.3016**	**0.3016**	**0.3016**
$\geq$ 200	0.6028	**0.0429**	**0.0429**	**0.0429**

TABLE V: Mean translation error (in meters) for given number n of correspondences. The results are averaged over all sequences in the ETH3D dataset.

[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] 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.

[12] 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 (ICRA)*, 2020.

[13] E. Malis, "Complete closed-form and accurate solution to pose estimation from 3d correspondences," *IEEE Robotics and Automation Letters*, vol. 8, pp. 1786 – 1793, 2023.

[14] ——, "A novel closed-form approach for enhancing efficiency in pose estimation from 3d correspondences," *IEEE Robotics and Automation Letters*, vol. 9, pp. 1843–1850, 2024.

[15] J. P. Jouanolou, "Formes d'inertie et résultant: un formulaire," *Adv. Math.*, vol. 126, no. 2, pp. 119–250, 1997. [Online]. Available: https://doi.org/10.1006/aima.1996.1609

[16] L. Busé, M. Chardin, and N. Nemati, "Multigraded sylvester forms, duality and elimination matrices," *Journal of Algebra*, vol. 609, pp. 514–546, 2022.

[17] D. A. Cox, J. B. Little, and D. O'Shea, *Using algebraic geometry*, 2nd ed., ser. Graduate texts in mathematics. New York: Springer, 2005.

[18] D. Eisenbud, *The geometry of Syzygies a second course in commutative algebra and algebraic geometry*, ser. Graduate texts in mathematics, 229. New York, N.Y., [etc: Springer, 2005.

[19] L. Busé, F. Catanese, and E. Postinghel, *Algebraic curves and surfaces: a history of shapes*, ser. SISSA Springer Series. Springer, 2023, vol. 4.

[20] B. Sturmfels, *Algorithms in Invariant Theory*. Springer-Verlag, Vienna, 1993.

[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[22] G. Nakano, "Globally optimal dls method for pnp problem with cayley parameterization," in *British Machine Vision Conference*, 2015.

[23] P. Wang, G. Xu, Y. Cheng, and Q. Yu, "A simple, robust and fast method for the perspective-n-point problem," *Pattern Recognition Letters*, vol. 108, pp. 31–37, 2018.

[24] Y. Zheng, Y. Kuang, S. Sugimoto, K. Åström, and M. Okutomi, "Revisiting the pnp problem: A fast, general and optimal solution," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 2344–2351.

[25] G. Terzakis and M. Lourakis, "A consistently fast and globally optimal solution to the perspective-n-point problem," in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*. Springer-Verlag, 2020, p. 478–494.

[26] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, "A multi-view stereo benchmark with high-resolution images and multi-camera videos," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[27] L. Kneip and P. Furgale, "Opengv: A unified and generalized approach to real-time calibrated geometric vision," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1–8.