



A Complete Workflow for Automatic Forward Kinematics Model Extraction of Robotic Total Stations Using the Denavit-Hartenberg Convention

Christoph Klug¹ · Dieter Schmalstieg² · Thomas Gloor³ · Clemens Arth²

Received: 8 November 2017 / Accepted: 6 September 2018
© The Author(s) 2018

Abstract

Development and verification of real-time algorithms for robotic total stations usually require hard-ware-in-the-loop approaches, which can be complex and time-consuming. Simulator-in-the-loop can be used instead, but the design of a simulation environment and sufficient detailed modeling of the hardware are required. Typically, device specification and calibration data are provided by the device manufacturers and are used by the device drivers. However, geometric models of robotic total stations cannot be used directly with existing ro-bot simulators. Model details are often treated as company secrets, and no source code of device drivers is available to the public. In this paper, we present a complete workflow for automatic geometric model extraction of robotic total stations using the Denavit-Hartenberg convention. We provide a complete set of Denavit-Hartenberg parameters for an exemplary ro-bot total station. These parameters can be used in existing robotic simulators without modifications. Furthermore, we analyze the difference between the extracted geometric model, the calibrated model, which is used by the device drivers, and the standard spherical representation for 3D point measurements of the device.

Keywords Robot control · Robot kinematics · Forward kinematics · Robotic total station · Denavit-Hartenberg parameters

1 Introduction

Robotic total stations (RTS) are commonly used for measuring 3D points with high precision and accuracy. Applications vary from land surveying over building and road

construction to as-built verification and deformation monitoring. Modern devices allow the measurement of distances and angles, provide image data, and automatically compensate various system effects, such as inaccuracies in production, sensor drifts and environmental influences [38]. While measured 3D points are usually represented in spherical coordinates, these devices are calibrated with extended geometric models by the manufacturers to reduce systematic errors, to combine multiple sensors and to achieve higher precision and accuracy. Manufacturers usually provide a software development kit (SDK) to access and convert the measured data. However, the detailed mathematical formulation of the corrections is often a company secret and is not available to the public.

The contribution of this work is a detailed description of the forward kinematics model parameter estimation for an (RTS) using the Denavit-Hartenberg (DH) convention [8], which can hardly be found in the literature in a closed form. Based on this description, we provide the estimated (DH) parameters for an exemplary (RTS). These parameters can be used for custom or existing robotic simulators, such as Roboanalyzer [27], WorkcellSimulator [37], ABB

¹ VRVis Research Center, Donau-City-Strasse 11, 1220 Wien, Austria

² Graz University of Technology, Inffeldgasse 16/2, 8010 Graz, Austria

³ Hilti Corporation, Feldkircherstrasse 100, 9494 Schaan, Liechtenstein

RoboStudio [5] or OpenSim [7]. The introduced model estimation workflow can also be applied to open chain robots with comparable actuators and sensors in general. In particular, we extended the methods described by Barker [2] and Rajeevlochana et al. [28] to extract the relationship between the robot control parameters, one or more cameras and the electronic distance meter (EDM) of an (RTS) automatically. Furthermore, we compared three different kinematics model representations for the exemplary (RTS): (a) the geometrically extracted kinematics model, (b) an simplified model using a spherical coordinate system and (c) the numerically optimized kinematics model.

This paper is structured as follows. Related work on (RTS), (DH) parameter estimation and robotic modeling are described in Section 2. In Section 3, the kinematics modeling workflow is described. In Section 4, experimental results for the kinematics model of an exemplary RTS are presented, including the comparison of the error for the geometrically extracted, analytically simplified, and numerically optimized models. A brief discussion about findings, limitations and possible future work is provided in Section 5, while concluding remarks are given in Section 6.

2 Related Work

Most robotic simulation environments for open kinematic chains use DH parameters for geometric linkage simulations [3, 5, 7, 22, 27, 30, 35, 37]. While DH parameter based models are quite common in robotic theory, RTS specifications do not provide this kind of information. Without the claim to completeness, this section provides an overview of published work in the field of RTS, the particular device used in this paper, the DH convention and related kinematics modeling approaches.

Uren and Price [38] provide an introduction to surveying devices and methods, including standardized mathematical models, measurement methods, error calculations and common workflows. Forward kinematics modeling or device simulation are not described, however.

Klug et al. [21] use a spherical model for the RTS in the context of human structure measurements. However, the applied model is an idealized representation of the device. The full kinematics model and the influence of the simplification are not analyzed (see Fig. 1).

Kinematics modeling and model identification has been addressed extensively in the literature, using different models and notations [6, 8, 12, 13, 33, 36]. Reuleaux [29] introduced the concept of kinematic chains in 1876. Denavit and Hartenberg [8] presented a systematic notation for kinematic chains, later called DH convention, which is still the most common method to identify rigid kinematic chains. Veitschegger and Wu [39] added the base and tool transform

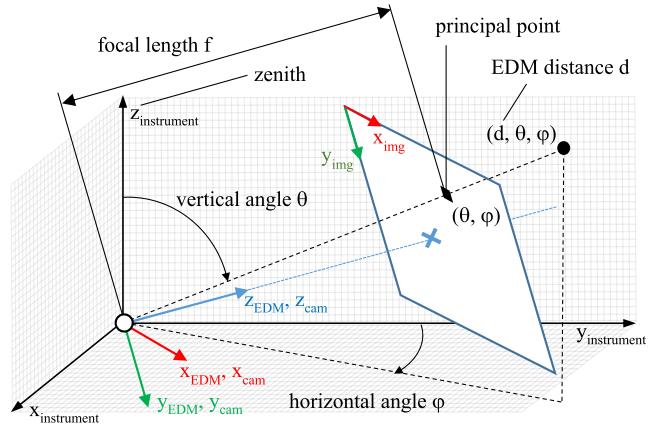


Fig. 1 Simplified kinematics model used in [21]

for complete kinematics description. A general introduction to the field of robotics, including a detailed description of the DH convention and its variants, can be found in the book by Siciliano and Khatib [34]; a comparison of different DH convention variants and an extensive study of geometric linkages can be found in book by McCarthy and Soh [25]. Despite the long history of kinematics modeling of robotic devices, to the best of our knowledge, no complete description for kinematics model extraction of an RTS with camera has been presented so far.

Barker [2] describes a vector-algebra approach for extracting geometric properties of assembled robotic arms. Rajeevlochana et al. [28] present a description for automatic model parameter estimation using a modified version of the algorithm based on line geometry. More details about their workflow, data acquisition, model extraction, and modeling error evaluation are given in the work of Hayat et al. [15] and in the work of Chittawadigi et al. [4]. However, their work is not tailored to RTS, hence device-specific algorithm steps are missing. Furthermore, numerical optimization of the geometrically extracted models was not addressed by the authors.

In our work, we describe a workflow based on DH parameters, which is tailored to kinematics modeling and simulation of modern RTS. To the best of our knowledge, we are the first to describe such a workflow in self-contained form. Furthermore, we show the significance of numerical optimization with a global cost function in addition to the geometric approach. Finally, instead of focusing on a single application case, we analyze the difference between the geometrically extracted model, the numerically optimized model and the spherical approximation.

3 Forward Kinematics Modeling of RTS

An RTS is an electrical theodolite, which consists of an optical telescope, an electrical distance meter and one or

more cameras [38]. Modern devices support teleoperation; hence, the optical telescope of the instrument is completely replaced by a camera module without any eyepiece. While the idealized model of the robotic module defines a spherical coordinate system, this model is violated by inevitable inaccuracies in manufacturing and physical device calibration. Manufacturing details are usually trade secrets, and only limited information is available to the public. Software-side calibration can reduce these errors, but requires an extended kinematics model of the RTS.

By extracting a combined model for hardware and software, a more accurate simulation environment can be implemented, device specifications can be verified, and less in-depth knowledge of the physical device is required.

The linear geometric properties of the rigid body can be described using a series of Euclidean transformation matrices with six degree of freedom (DOF) each. In the field of robotics, more systematic approaches with fewer DOF are known for estimating kinematic and dynamic properties, such as the DH convention. In this section, we provide a detailed description of how to derive a geometric model for RTS using the DH convention. This allows to estimate device properties when specification are missing, to verify calibration results, and to re-use existing simulation environments for robotic devices.

3.1 Forward Kinematics

The dynamic parts of an RTS can be described as a spatial kinematic chain. *Forward kinematics* describes the pose of an end effector using *links* and *joints* without consideration of driving forces or moments [23]. Joints are rigidly connected by links and can be affected at run-time, using the related *joint control variables*. *End effectors* or *tools* are sensors or actuators attached to the last link of a robotic device.

The DH convention was originally introduced to describe the geometric relationship of an open kinematic chain with $M - 1$ joints and M links, using a series of homogeneous joint displacement matrices \mathbf{Z}_i and link displacement matrices \mathbf{X}_i [8]. In this work, we used the (DH) notation proposed by Rajeevlochana and Saha [27, 31].

By convention, joint i connects link $i - 1$ and link i . The displacement matrices \mathbf{Z}_i and \mathbf{X}_i define the local coordinate frame i by $\{\mathbf{o}_i, \mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}$ where \mathbf{o}_i is the frame origin, and \mathbf{x}_i , \mathbf{y}_i , and \mathbf{z}_i are the normalized \mathbf{x} , \mathbf{y} , and \mathbf{z} axes, respectively. Frame i is rigidly attached to the end of link $i - 1$ and must satisfy the following conditions: (i) The \mathbf{x}_i axis is perpendicular to the \mathbf{z}_{i-1} axis; (ii) the \mathbf{x}_i axis intersects with the \mathbf{z}_{i-1} axis.

The DH convention defines two types of joints: (a) *revolute joints* and (b) *prismatic joints*. A revolute joint allows a rotation around the \mathbf{z} axis of frame i by the angle

γ_i . A prismatic joint allows the translation along the \mathbf{z} axis of frame i by the distance d_i .

The relative pose $\mathbf{M}_{i,j}$ of frame j with respect to frame i is given by

$$\mathbf{M}_{i,j} = \prod_{n=i}^j \mathbf{M}_{n,n+1} \quad (1)$$

$$\mathbf{M}_{i,i+1} = \mathbf{Z}_i \cdot \mathbf{X}_i \quad (2)$$

The joint matrix \mathbf{Z}_i describes a screw displacement along the \mathbf{z} axis of frame i

$$\mathbf{Z}_i = \mathbf{T}_z(d_i) \cdot \mathbf{R}_z(\gamma_i) \quad (3)$$

where d_i and γ_i are the control variables for joint i .

The link matrix \mathbf{X}_i describes a rigid screw displacement along the \mathbf{x} axis of frame i

$$\mathbf{X}_i = \mathbf{T}_x(a_i) \cdot \mathbf{R}_x(\alpha_i) \quad (4)$$

where a_i and α_i are used to define the static properties of link i . Figure 2 shows the relative pose $\mathbf{M}_{i,i+1}$ of frame j in respect to frame i using the DH convention.

The set Q_i contains the control variables for link i , which are called the *DH parameters*:

$$Q_i = \{d_i, \gamma_i, \alpha_i, a_i\} \quad (5)$$

The rigid twist of link $i - 1$ is given by the rotation matrix \mathbf{R}_x , while the rigid length of link $i - 1$ is given by the translation matrix \mathbf{T}_x :

$$\mathbf{T}_x(a_i) = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R}_x(\alpha_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha_i & -\sin \alpha_i & 0 \\ 0 & \sin \alpha_i & \cos \alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

The rotation of link i around joint i is given by the rotation matrix \mathbf{R}_z , the translation of joint i along the \mathbf{z}_i axis is given by the translation matrix \mathbf{T}_z :

$$\mathbf{T}_z(d_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R}_z(\gamma_i) = \begin{bmatrix} \cos \gamma_i & -\sin \gamma_i & 0 & 0 \\ \sin \gamma_i & \cos \gamma_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

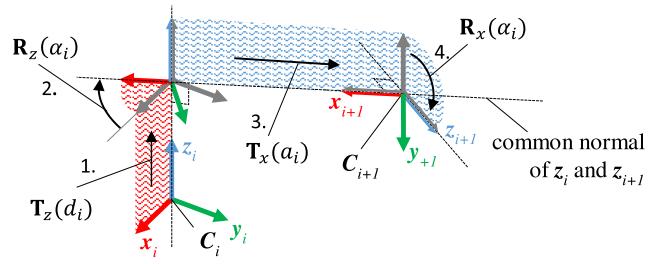
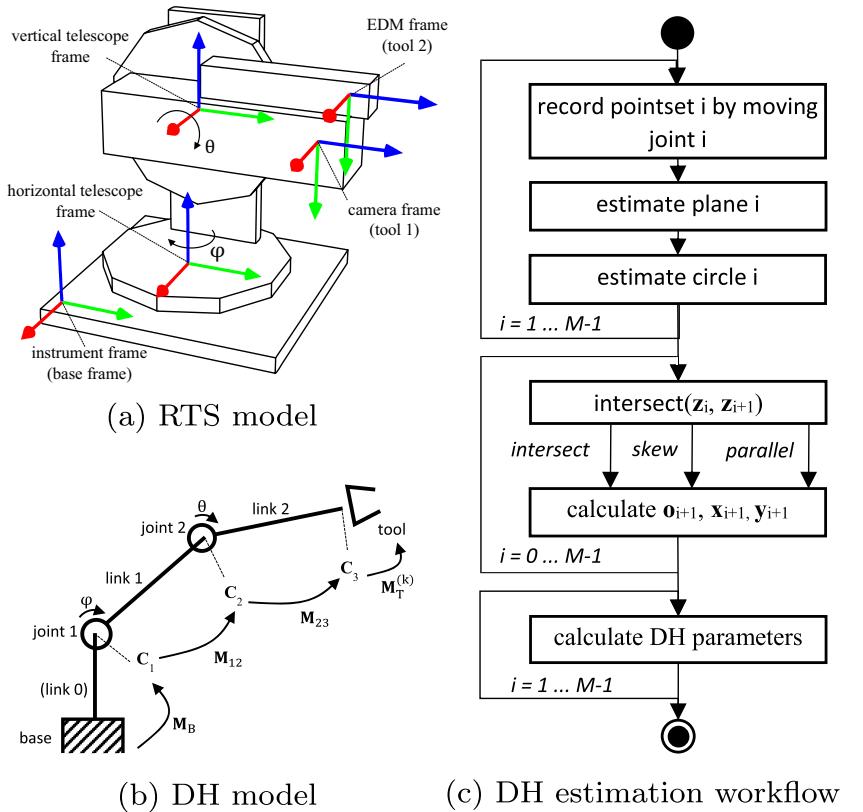


Fig. 2 Relative pose $\mathbf{M}_{i,i+1}$ of frame $i + 1$ with respect to frame i as defined by the DH convention

Fig. 3 **a** Coordinate frames of an exemplary RTS. For better visualization, the position of the frames are drawn non-overlapping. **b** DH model of an RTS, showing joints and links with related frames and transformations. In literature, this is also called an *RR open chain* [25]. **c** Workflow for extracting DH parameters



The pose of frame $i+1$ with respect to frame i can be written as

$$\mathbf{M}_{i,i+1} = \begin{bmatrix} \cos \gamma_i & -\sin \gamma_i \cos \alpha_i & \sin \gamma_i \sin \alpha_i & a_i \cos \gamma_i \\ \sin \gamma_i & \cos \gamma_i \cos \alpha_i & -\cos \gamma_i \sin \alpha_i & a_i \sin \gamma_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \left[\begin{array}{c|c} \mathbf{R}_{i,i+1} & \mathbf{t}_{i,i+1} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] \quad (8)$$

where $\mathbf{R}_{i,i+1}$ is a 3×3 rotation matrix, and $\mathbf{t}_{i,i+1}$ is a 3×1 translation vector.

3.1.1 Forward Kinematics Modeling Workflow for RTS

Estimating the forward kinematics model of an RTS can be reduced to the problem of deriving the DH parameters for a robotic device.

Figure 3a shows the coordinate frames of an RTS with a single camera.¹ The pose of the RTS with respect to the reference frame \mathbf{C}_0 could be defined by the base transform \mathbf{M}_B , which allows arbitrary placement of the robot in the scene. Multiple tools which are rigidly attached to the last

link can be described by adding individual tool transforms $\mathbf{M}_T^{(k)}$. The pose of tool k with respect to the reference frame can be written as

$$\mathbf{M}_{B,T}^{(k)} = \mathbf{M}_B \cdot \mathbf{M}_{1,2} \dots \mathbf{M}_{M-1,M} \cdot \mathbf{M}_T^{(k)} \quad (9)$$

The transformation of homogeneous points from tool space k to the reference space is given by

$$\hat{\mathbf{x}}_{i,0} = \mathbf{M}_{B,T}^{(k)} \hat{\mathbf{x}}_{i,T}^{(k)} \quad (10)$$

where $\hat{\mathbf{x}}_{i,T}^{(k)}$ defines a homogeneous point in the tools space k , and $\hat{\mathbf{x}}_{i,0}$ defines the same point in the reference space.

The steps required for estimating the DH parameters according the workflow of Chittawadgi et al. [4] are: 1. identify joint count and type of joints, 2. identify end effectors, 3. record end effector poses, while varying a single joint, and 4. calculate the DH parameters. Figure 3b shows the model of an exemplary RTS. The system can be described by a reference frame $\mathbf{C}_0 = \mathbf{C}_B$, two revolute joints, and two end effectors $\mathbf{C}_T^{(EDM)}$ and $\mathbf{C}_T^{(CAM)}$.

The first and second joint describe the horizontal rotation φ and the vertical rotation θ , respectively. The rigidly attached end effectors are described by the (EDM) frame $\mathbf{C}_T^{(EDM)}$ and the camera frame $\mathbf{C}_T^{(CAM)}$, respectively.²

¹Coordinate frames for different RTS are not standardized and vary between individual devices.

²Note that only the end-to-end transformations between base frame and tool frames matches the device manual, the inner frames of the model are defined differently by the DH convention.

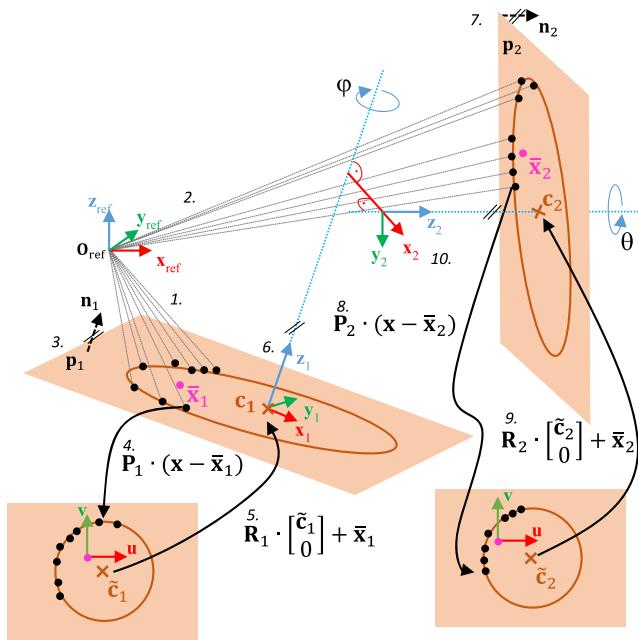


Fig. 4 Circular feature extraction. The algorithm steps are labeled in ascending order: 1,2: measure points; 3,7: fit plane; 4,8: project to 2D; 5,9: fit circle and transform result to 3D; 6,10: define frames

The rotation and orientation of a joint can be estimated by recording end effector positions with respect to the reference frame, while rotating only one joint at a time. Figure 3c shows the flow chart for measuring end effector positions and extracting DH parameters.

The recorded 3D points describe a planar circular trajectory. The center of the circles and the plane of rotation can be used to estimate the DH parameters. Figure 4 shows the concept for circular feature extraction.

3.1.2 Data Acquisition for DH Parameter Estimation

The estimation of DH parameters usually requires an external measurement setup for measuring the pose of the end effectors. However, for forward kinematics model estimation of an RTS, the required data can be fetched from the application programming interface (API) of the device without any external devices.³

For each joint, a trajectory is defined by recording end effector positions, while varying the related joint control parameter and keeping other control parameters constant. A linear trajectory of the recorded end effector positions indicates a prismatic joint. A circular trajectory of the recorded points indicates a revolute joint.

If the recorded data does neither describe a linear or circular movement, the affected joint type is either not prismatic or revolute, or the end effector coincides with the

rotation axis of a revolute joint. To avoid singularities during DH parameter extraction, the recording for the affected joint has to be repeated using different fixed joint settings or different API parameters.⁴

Figure 5a shows the required measurement API. Joint $i = 1$ is the horizontal rotation, joint $i = 2$ is the vertical rotation, and $\mathcal{F}(\varphi, \theta)$ is the control function for the corresponding joints. Data recording for joint 1 can be done by varying φ in the range $[0, 2\pi]$, while keeping $\theta = \frac{\pi}{2}$ constant. Data recording for joint 2 can be done by varying θ between $[0, \pi]$, while keeping $\varphi = 0$ constant. The device API function

$$\mathbf{x}_i = \mathcal{F}^{(\text{EDM})}(\varphi_i, \theta_i, d_{\text{EDM}}) \quad (11)$$

applies the control variables $\{\varphi_i, \theta_i\}$ and converts a 1D distance d_{EDM} from (EDM) space to a 3D point \mathbf{x}_i in the reference frame. The device API function

$$\mathbf{x}_i = \mathcal{F}^{(\text{CAM})}(\varphi_i, \theta_i, \mathbf{u}, d_{\text{CAM}}) \quad (12)$$

applies the control variables $\{\varphi_i, \theta_i\}$ and converts a 2D image pixel \mathbf{u} to a 3D point \mathbf{x}_i in the reference frame. The two functions $\mathcal{F}^{(\text{EDM})}$ and $\mathcal{F}^{(\text{CAM})}$ are sufficient for forward kinematics modeling. They must be provided by the API of the RTS. Figure 5b shows the parameter space of the angle control variables $\{\varphi_i, \theta_i\}$; Fig. 5c and d show the positions of the recorded EDM and camera end effectors, respectively.

3.1.3 Estimating Circular Features

A plane \mathbf{p} can be defined as

$$\mathbf{p} = \begin{bmatrix} \mathbf{n} \\ -\mathbf{n}^T \cdot \bar{\mathbf{x}} \end{bmatrix} \quad \|\mathbf{n}\| = 1 \quad (13)$$

where $\mathbf{n} = [n_x \ n_y \ n_z]^T$ is the normalized plane normal. Given a set of 3D points $S_p = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$, the center of mass $\bar{\mathbf{x}}$ is given by

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \quad (14)$$

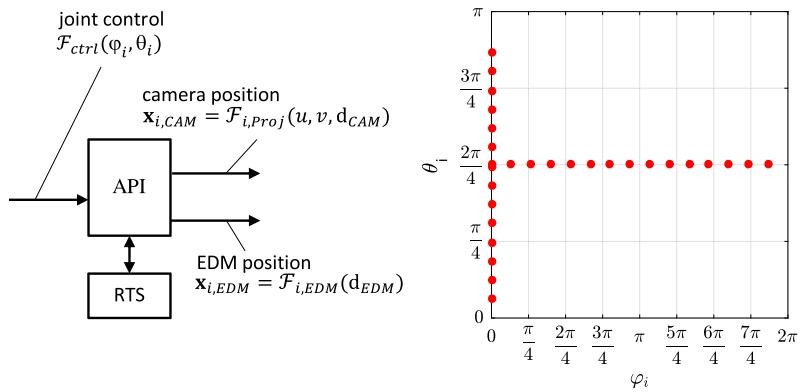
The plane normal can be fitted to the point cloud S_p using Singular Value Decomposition (SVD):

$$\text{SVD}(\tilde{\mathbf{A}}) = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^T \quad (15)$$

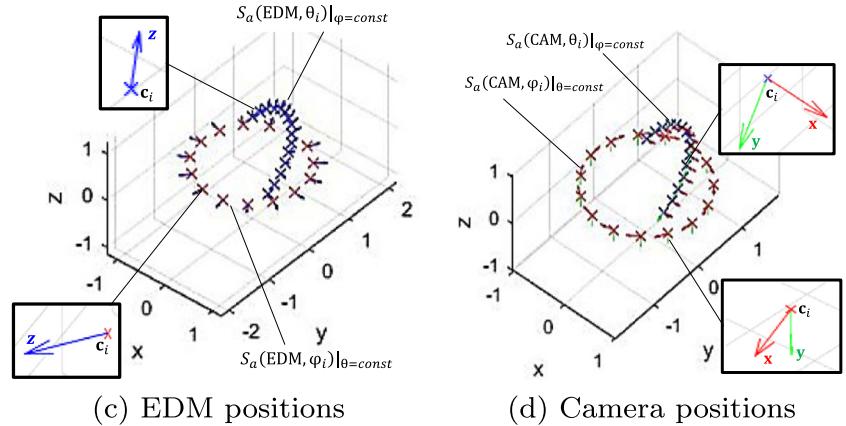
³For device calibration, external measurements would still be required.

⁴If all link lengths are close to zero, an artificial end effector offset must be applied. This can be done by recording 3D points which do not coincide with the end effector origin.

Fig. 5 **a** Control behavior for recording 3D points of the end effector during DH parameter estimation. **b** Angle control parameter space coverage for model estimation. **c** Recorded EDM end effector positions \mathbf{c}_i with artificial link length by setting EDM distance measurement to 1 m. **d** Recorded camera end effector positions \mathbf{c}_i with artificial link length by setting back-projected pixel ray length to 1 m



(a) RTS data recording API (b) Control parameter space



(c) EDM positions (d) Camera positions

where \mathbf{U} is a 3×3 matrix, \mathbf{S} is a $3 \times N$ matrix, \mathbf{V} is a $N \times N$ matrix. $\tilde{\mathbf{A}}$ is a $3 \times N$ matrix containing the normalized and centered measurement samples.

$$\mathbf{A} = [\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_N] \quad (16)$$

$$\bar{\mathbf{A}} = [(\mathbf{x}_1 - \bar{\mathbf{x}}) (\mathbf{x}_2 - \bar{\mathbf{x}}) \dots (\mathbf{x}_N - \bar{\mathbf{x}})] \quad (17)$$

$$\tilde{\mathbf{A}} = \frac{1}{k} \bar{\mathbf{A}} \quad k = \max_{a_{ij} \in \bar{\mathbf{A}}} |a_{ij}| \quad (18)$$

The normalization k is the absolute maximum value of all elements in $\bar{\mathbf{A}}$. A vector parallel to the plane normal \mathbf{n} is given by the eigenvector of $\tilde{\mathbf{A}}$, which corresponds to the smallest eigenvalue, which is simply the last column \mathbf{u}_3 of $\mathbf{U} = [\mathbf{u}_1 \mathbf{u}_2 \mathbf{u}_3]$. Plane \mathbf{p} is then fully defined by Eqs. 13 and 14.

It is convenient to define a right-handed orthogonal basis \mathbf{B} for each plane such that the plane normal is aligned with the \mathbf{z} axis, and the plane contains two additional orthogonal vectors according to

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{U}) \end{bmatrix} \mathbf{U} \quad (19)$$

$$\mathbf{B} = [\mathbf{b}_1 \mathbf{b}_2 \mathbf{b}_3] \quad \mathbf{n} = \mathbf{b}_3 \quad (20)$$

The determinant of \mathbf{U} is either equal to 1 if no reflection happens or equal to -1 in case of a reflection. Therefore \mathbf{B} can be interpreted as a reflection-free rotation of the plane with respect to the reference frame.

All points must be transformed to the xy plane and projected to a 2D Euclidean space before fitting a circle to the planar measurements. This can be formalized as projection of the centered measurement matrix $\tilde{\mathbf{A}}$ given in Eq. 17:

$$\mathbf{A}_c = \mathbf{P}_c \cdot \tilde{\mathbf{A}} \quad \mathbf{P}_c = \begin{bmatrix} \mathbf{b}_1^T \\ \mathbf{b}_2^T \end{bmatrix} \quad (21)$$

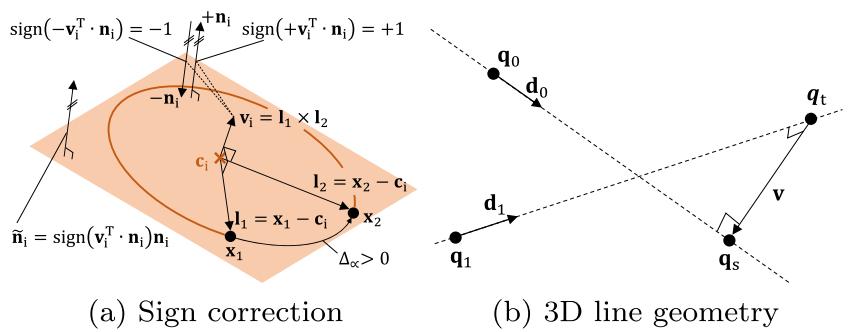
where \mathbf{A}_c is a $[2 \times N]$ matrix, and \mathbf{P}_c is a $[2 \times 3]$ projection matrix which applies the inverse plane rotation \mathbf{B}^T and the projection of the stacked 3D points $\tilde{\mathbf{A}}$ to the 2D space.⁵

A circle in 2D is given by the implicit equation

$$(u_i - c_1)^2 + (v_i - c_2)^2 = r^2 \quad (22)$$

⁵No homogeneous coordinates are required, since \mathbf{P}_c does not include any 3D translation.

Fig. 6 **a** Sign correction of circular features: Δ_α is the difference of the angular control parameters of two consecutive measurement samples, $\pm \mathbf{n}_i$ is the extracted plane normal, defined up to sign. **b** Distance between two lines in 3D



where r is the radius of the circle, $\tilde{\mathbf{c}} = [c_1 \ c_2]^T$ is the center of the circle, and $[u_i \ v_i]^T$ is a point on the circle in 2D. Rearranging (22) leads to

$$2 \cdot u_i \cdot c_1 + 2 \cdot v_i \cdot c_2 + k_3 = u_i^2 + v_i^2 \quad (23)$$

$$k_3 = r^2 - c_1^2 - c_2^2 \quad (24)$$

which is linear in the unknown parameters c_1 , c_2 and k_3 . This can be written as an inhomogeneous linear system

$$\tilde{\mathbf{A}}_c^T \cdot \mathbf{x}_c = \mathbf{b}_c \quad \tilde{\mathbf{A}}_c = \begin{bmatrix} 2 \cdot \mathbf{A}_c \\ \mathbf{1}_{1 \times N} \end{bmatrix} \quad \mathbf{x}_c = \begin{bmatrix} c_1 \\ c_2 \\ k_3 \end{bmatrix} \quad (25)$$

$$\mathbf{b}_c = \begin{bmatrix} \xi_1 \\ \xi_2 \\ \dots \\ \xi_N \end{bmatrix} \quad \xi_i = u_i^2 + v_i^2 \quad (26)$$

where $\tilde{\mathbf{A}}_c$ is a $3 \times N$ matrix, and ξ_i is the sum of squared 2D coordinate values of point i . SVD can be used to solve for the unknown circle parameters:

$$\text{SVD}(\tilde{\mathbf{A}}_c^T) = \mathbf{U}_c \cdot \mathbf{S}_c \cdot \mathbf{V}_c^T \quad (27)$$

$$\mathbf{x}_c = \mathbf{V}_c \cdot \mathbf{S}_c^+ \cdot \mathbf{U}_c^T \cdot \mathbf{b}_c \quad (28)$$

$$\mathbf{S}_c^+ = \left[\begin{array}{ccc|c} 1/S_{c,11} & 0 & 0 & \mathbf{0}_{3 \times (N-3)} \\ 0 & 1/S_{c,22} & 0 & \\ 0 & 0 & 1/S_{c,33} & \end{array} \right] \quad (29)$$

where \mathbf{U}_c is a $N \times N$ matrix, \mathbf{S}_c is a $N \times 3$ matrix and \mathbf{V}_c^T is a 3×3 matrix.

The 3D circular feature Φ_c for a single joint is fully defined by $\Phi_c = \{\mathbf{c}, r\}$, using

$$\mathbf{c} = \mathbf{B} \cdot \begin{bmatrix} c_1 \\ c_2 \\ 0 \end{bmatrix} + \bar{\mathbf{x}} \quad r = \sqrt{k_3 + c_1^2 + c_2^2} \quad (30)$$

where \mathbf{B} is the plane rotation according to Eq. 20, \mathbf{c} is the center of the circle with respect to the reference frame, and r is the radius of the circle.

3.1.4 Correcting the Sign of Circular Features

A plane normal \mathbf{n} as given in Eq. 13 is defined up to sign. This leads to sign ambiguities in the DH control parameters.

For revolute joints, this can lead to an inverse rotation, for prismatic joints, this can lead to an inverse translation. The ambiguity can be resolved using two consecutive measurement points \mathbf{x}_1 and \mathbf{x}_2 of the circular feature i

$$\tilde{\mathbf{n}}_i = \text{sign}\left((\mathbf{x}_1 - \mathbf{c}_i) \times (\mathbf{x}_2 - \mathbf{c}_i)^T \mathbf{n}_i\right) \mathbf{n}_i \quad (31)$$

where \mathbf{c}_i is the center of the circle, and $\text{sign}(x)$ is the sign of the scalar value x according to

$$\text{sign}(x) = \begin{cases} -1, & x < 0 \\ 1, & \text{otherwise} \end{cases} \quad (32)$$

The corrected plane normal $\tilde{\mathbf{n}}_i$ ensures a positive rotation direction for increasing control parameter values, if point \mathbf{x}_1 was recorded with a smaller control parameter than point \mathbf{x}_2 .⁶ Figure 6a shows the concept for sign correction of circular features.

3.1.5 Link Constellation and Frame Alignment

The link constellation and frame alignment are based on the spatial relationship of the \mathbf{z}_{i-1} and the \mathbf{z}_i axes. An intersection is treated as special case of skewed lines.

Rajeevlochana et al. used Plücker coordinates and Dual Vector Algebra for estimating the line constellation [28]. Plücker coordinates allow closed form line intersection testing. However, for kinematic chains with low link counts, we did not observe any computational benefits when using Plücker coordinates instead of simple vector algebra as described in the work by Barker [2]. Thus, for the sake of simplicity, we describe line constellations using simple vector algebra.

Figure 6b shows the distance of two lines in the 3D space. Given two lines \mathcal{L}_0 and \mathcal{L}_1 in their parametric form

$$\mathcal{L}_0(s) = \mathbf{q}_0 + s\mathbf{d}_0 \quad \mathcal{L}_1(t) = \mathbf{q}_1 + t\mathbf{d}_1 \quad (33)$$

we wish to find the minimum distance between the two lines. Let \mathbf{q}_s and \mathbf{q}_t define points on line \mathcal{L}_0 and \mathcal{L}_1 which minimizes the length of the vector

$$\mathbf{v} = \mathbf{q}_s - \mathbf{q}_t \quad (34)$$

⁶For revolute joints, it is advisable to use control angles between $\{0 \dots \pi\}$ for sign correction to avoid errors caused by periodicity.

using

$$\mathbf{q}_s = \mathbf{q}_0 + s_s \mathbf{d}_0 \quad \mathbf{q}_t = \mathbf{q}_1 + t_s \mathbf{d}_1 \quad (35)$$

The minimum Euclidean distance between the lines is given by $\|\mathbf{v}\|$, if \mathbf{v} has the same direction as the common normal, which can be written as

$$\mathbf{d}_0^T \mathbf{v} = 0 \quad \mathbf{d}_1^T \mathbf{v} = 0 \quad (36)$$

If we substitute Eqs. 34 and 35 in Eq. 36 we can solve for the two unknown parameters s_s and t_s

$$s_s = \frac{be - cd}{ac - b^2} \quad t_s = \frac{ae - bd}{ac - b^2} \quad (37)$$

where

$$a = \|\mathbf{d}_0\|^2 \quad b = \mathbf{d}_0^T \mathbf{d}_1 \quad c = \|\mathbf{d}_1\|^2 \quad (38)$$

$$d = \mathbf{d}_0^T (\mathbf{q}_0 - \mathbf{q}_1) \quad e = \mathbf{d}_1^T (\mathbf{q}_0 - \mathbf{q}_1) \quad (39)$$

The constellation type of two lines $c_c \in \{\text{parallel, skewed}\}$ can be determined from the divisor of Eq. 37:

$$c_c = \begin{cases} \text{parallel, } (ac - b^2) = 0 \\ \text{skewed, otherwise} \end{cases} \quad (40)$$

The parameters s_c and t_c for calculating the closest points \mathbf{q}_s and \mathbf{q}_t on line \mathcal{L}_0 and \mathcal{L}_1 , respectively, are given by

$$s_c = \begin{cases} 0, \text{ parallel} \\ s_s, \text{ skewed} \end{cases} \quad t_c = \begin{cases} d/b, \text{ parallel, } b \geq c \\ e/c, \text{ parallel, } b < c \\ t_s, \text{ skewed} \end{cases} \quad (41)$$

Finally, the points \mathbf{q}_s and \mathbf{q}_t can be calculated by substituting the parameters from Eq. 41 into the line equations given in Eq. 33:

$$\mathbf{q}_s = \mathcal{L}_0(s_c) \quad \mathbf{q}_t = \mathcal{L}_1(t_c) \quad (42)$$

3.1.6 First Link Frame

The \mathbf{z} direction of the first frame is aligned with the plane normal of the first circular feature $\tilde{\mathbf{n}}_1$

$$\mathbf{z}_1 = \tilde{\mathbf{n}}_1 \quad (43)$$

The origin of the first frame \mathbf{o}_1 must lie on the first rotational axis $\mathcal{L}_{z1}(s) = \mathbf{c}_1 + s \tilde{\mathbf{n}}_1$; the \mathbf{x} axis of the first frame must lie on the plane defined by \mathbf{o}_1 and \mathbf{z}_1 . The rotation of \mathbf{x}_1 around \mathbf{z}_1 can be arbitrary defined. The first frame is fully defined by

$$\mathbf{o}_1 = \mathbf{c}_1 \quad \mathbf{B}_1 = [\mathbf{x}_1 \mathbf{y}_1 \mathbf{z}_1] \quad (44)$$

where \mathbf{B}_1 is a right-handed orthogonal base according to Eq. 20.

However, this approach will lead to different DH parameters for different point sets of the first circular feature.

A more convenient approach is to further align the \mathbf{x} direction of the reference frame \mathbf{x}_0 with the \mathbf{x} direction of

the first frame \mathbf{x}_1 by projecting \mathbf{x}_0 onto plane \mathbf{p}_1 :

$$\mathbf{x}_1 = \frac{\mathbf{x}_0 - (\mathbf{x}_0^T \mathbf{z}_1) \mathbf{z}_1}{\|\mathbf{x}_0 - (\mathbf{x}_0^T \mathbf{z}_1) \mathbf{z}_1\|} \quad \mathbf{y}_1 = \mathbf{z}_1 \times \mathbf{x}_1 \quad (45)$$

If plane \mathbf{p}_1 is parallel to the yz plane of the reference frame, the alignment of the \mathbf{y} directions can be used instead.

3.1.7 Middle Link Frames

The frames $i = 2, \dots, M - 1$ can be defined iteratively using

$$\mathbf{z}_i = \tilde{\mathbf{n}}_i \quad \mathbf{z}_{i+1} = \tilde{\mathbf{n}}_{i+1} \quad \mathbf{o}_{i+1} = \mathbf{q}_t^{(i)} \quad (46)$$

$$\mathbf{x}_{i+1} = \begin{cases} \frac{\mathbf{z}_i \times \mathbf{z}_{i+1}}{\|\mathbf{z}_i \times \mathbf{z}_{i+1}\|}, \text{ } \mathcal{L}_i \text{ and } \mathcal{L}_{i+1} \text{ intersect} \\ \frac{\mathbf{q}_t^{(i)} - \mathbf{q}_s^{(i)}}{\|\mathbf{q}_t^{(i)} - \mathbf{q}_s^{(i)}\|}, \text{ are skewed or parallel} \\ \mathbf{x}_i, \text{ are identical} \end{cases} \quad (47)$$

$$\mathbf{y}_{i+1} = \frac{\mathbf{z}_{i+1} \times \mathbf{x}_{i+1}}{\|\mathbf{z}_{i+1} \times \mathbf{x}_{i+1}\|} \quad (48)$$

where $\mathbf{q}_s^{(i)}$ and $\mathbf{q}_t^{(i)}$ are the common normal intersections of \mathcal{L}_i and \mathcal{L}_{i+1} according to Eq. 35, respectively. Based on the calculated coordinate frames, the (DH) parameters for a revolute joint i can be derived by

$$\left. \begin{array}{l} b_i = (\mathbf{o}_{i+1} - \mathbf{o}_i)^T \mathbf{z}_i \\ \theta_i = \arctan2((\mathbf{x}_i \times \mathbf{x}_{i+1})^T \mathbf{z}_i, \mathbf{x}_i^T \mathbf{x}_{i+1}) \\ a_i = (\mathbf{o}_{i+1} - \mathbf{o}_i)^T \mathbf{x}_i \\ \alpha_i = \arctan2((\mathbf{z}_i \times \mathbf{z}_{i+1})^T \mathbf{x}_{i+1}, \mathbf{z}_i^T \mathbf{z}_{i+1}) \end{array} \right\} \quad (49)$$

3.1.8 Last Link Frame

The last frame can be used to define the tool pose. However, a general pose of the tool would require six DOF whereas a single frame of the DH framework is limited to four DOF. One solution to this problem is to use multiple DH frames to describe the tool pose. In this work, we used a more general approach by extracting a six DOF tool matrix $\mathbf{M}_T^{(k)}$ for each tool k separately.

By including individual tool matrices, the rotation of the last link frame around the \mathbf{z}_N axis is arbitrary. However, it is convenient to align the last frame with the previous one:

$$\mathbf{x}_M = \mathbf{x}_{M-1} \quad \mathbf{y}_M = \mathbf{y}_{M-1} \quad \mathbf{z}_M = \mathbf{z}_{M-1} \quad (50)$$

The DH parameters are then calculated according to Section 3.1.7.

3.1.9 Base Transform

The base transform defines the pose of the first coordinate frame $\{\mathbf{o}_1, \mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1\}$ with respect to the reference frame $\{\mathbf{o}_0, \mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0\}$ using the six DOF Euclidean transformation \mathbf{M}_B . This problem can be described as estimating 3D rigid transformations between two point sets and has been studied

extensively [1, 9, 17, 18]. In this work, we follow the SVD approach from reference [1].

Let $S_a = \{\mathbf{a}_1, \mathbf{a}_2 \dots \mathbf{a}_N\}$ and $S_b = \{\mathbf{b}_1, \mathbf{b}_2 \dots \mathbf{b}_N\}$ be two point sets with corresponding points. To determine the Euclidean transformation three or more correspondences are needed. We are looking for the best transformation in the least squares sense:

$$(\mathbf{R}, \mathbf{t}) = \mathcal{F}(S_a, S_b) = \underset{\mathbf{R}, \mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^N \|\mathbf{R}\mathbf{a}_i + \mathbf{t} - \mathbf{b}_i\|^2 \quad (51)$$

The centroids $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ of the point clouds are given by

$$\bar{\mathbf{a}} = \frac{1}{N} \sum_{i=1}^N \mathbf{a}_i \quad \bar{\mathbf{b}} = \frac{1}{N} \sum_{i=1}^N \mathbf{b}_i \quad (52)$$

The cross-covariance matrix of the two point sets is given by

$$\mathbf{H} = (\mathbf{S}_a - \bar{\mathbf{a}})(\mathbf{S}_b - \bar{\mathbf{b}})^T \quad (53)$$

where the measurement matrices \mathbf{S}_a and \mathbf{S}_b contains the stacked points of the two point sets

$$\begin{aligned} \mathbf{S}_a &= [\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_N]^T \\ \mathbf{S}_b &= [\mathbf{b}_1 \mathbf{b}_2 \dots \mathbf{b}_N]^T \end{aligned} \quad (54)$$

The SVD of the cross-covariance matrix \mathbf{H} is

$$\text{SVD}(\mathbf{H}) = \mathbf{U}_P \cdot \mathbf{S}_P \cdot \mathbf{V}_P^T \quad (55)$$

The rotation \mathbf{R} can be calculated by

$$\mathbf{R} = \mathbf{V}_P \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{V}_P \mathbf{U}_P^T) \end{bmatrix} \mathbf{U}_P^T \quad (56)$$

Finding the optimal rotation \mathbf{R} is also known as *Kabsch algorithm* [20]. The optimal translation \mathbf{t} is given by

$$\mathbf{t} = -\mathbf{R}\bar{\mathbf{a}} + \bar{\mathbf{b}}. \quad (57)$$

The rigid transform \mathbf{M}_B which describes the pose of the first link frame $\{\mathbf{o}_1, \mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1\}$ with respect to the reference frame $\{\mathbf{o}_0, \mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0\}$ can be calculated according to Eq. 51:

$$\mathbf{S}_a = [\mathbf{c}_0 + \mathbf{x}_0 \ \mathbf{c}_0 + \mathbf{y}_0 \ \mathbf{c}_0 + \mathbf{z}_0] \quad (58)$$

$$\mathbf{S}_b = [\mathbf{c}_1 + \mathbf{x}_1 \ \mathbf{c}_1 + \mathbf{y}_1 \ \mathbf{c}_1 + \mathbf{z}_1] \quad (59)$$

$$(\mathbf{R}_B, \mathbf{t}_B) = \mathcal{F}(\mathbf{S}_a, \mathbf{S}_b) \quad (60)$$

$$\mathbf{M}_B = \begin{bmatrix} \mathbf{R}_B & \mathbf{t}_B \\ \hline 0 & 0 & 0 & 1 \end{bmatrix} \quad (61)$$

3.1.10 Camera Tool Frame

The pose of the end effector k with respect to the last link frame $\{\mathbf{o}_M, \mathbf{x}_M, \mathbf{y}_M, \mathbf{z}_M\}$ is defined by a six DOF rigid transform. If two point sets with $N \geq 3$ correspondences

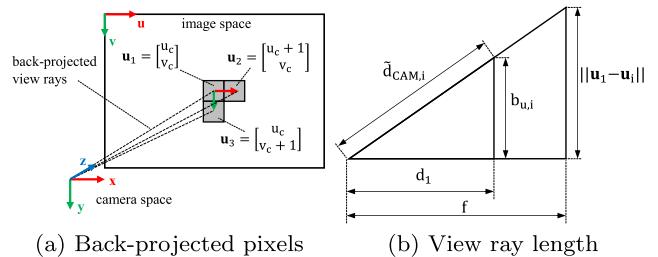


Fig. 7 **a** Recorded back-projected pixels for camera tool pose estimation. **b** Simplified view ray relation between principal ray and back projected pixel to validate ray length approximation during the camera tool pose estimation

can be obtained, the point set alignment algorithm as described in Section 3.1.9 can be applied.

The pose of the camera end effector can be defined by the end effector position, two orthogonal vectors describing the \mathbf{x} and \mathbf{y} direction, and the right-handed coordinate system constraint. The end effector position was already used for the DH parameter extraction. The \mathbf{x} and \mathbf{y} directions of the camera are aligned with the \mathbf{u} and \mathbf{v} direction of the image space, respectively. By extending the set of back-projected pixel coordinates during the measurement flow for the DH parameters, the \mathbf{x} and \mathbf{y} direction of the end effector with respect to the reference frame can be extracted. Figure 5c and d shows the end effector recordings, including the observable frame axis of the end effectors; Fig. 7a shows the back-projection method and the alignment of camera and image frame.

Given the camera intrinsics, the camera projection matrix \mathbf{P} can be written as

$$\hat{\mathbf{u}} = \mathbf{P}_{\text{CAM}} \mathbf{M}_{B,T}^{-1} \hat{\mathbf{x}}_i \quad (62)$$

$$\mathbf{P}_{\text{CAM}} = [\mathbf{K} | \mathbf{0}] \quad \mathbf{K} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \quad (63)$$

where $\hat{\mathbf{u}}$ is a homogeneous pixel point, $\hat{\mathbf{x}}_i$ is a homogeneous world space point, s is a skew factor, $\{f_u, f_v\}$ describe the focal lengths and $\mathbf{c} = [c_u \ c_v]^T$ describe the principal point offset in 2D.⁷

Matrix $\mathbf{M}_{B,T}$ is the camera tool pose with respect to the reference frame, using $\mathbf{M}_T^{(k)} = \mathbf{M}_T^{(\text{CAM})}$ in Eq. 10. By defining the tool space as camera space, the relationship between a homogeneous point $\hat{\mathbf{x}}_T$ in the camera space and a homogeneous point $\hat{\mathbf{x}}_0$ in the reference space is given by

$$\begin{aligned} \hat{\mathbf{x}}_0 &= \mathbf{M}_{B,T} \hat{\mathbf{x}}_T \\ &= (\mathbf{M}_B \mathbf{M}_{1,M} \mathbf{M}_T) \hat{\mathbf{x}}_T \end{aligned} \quad (64)$$

⁷For the sake of simplicity, we set $s = 0$ and $f_u = f_v = f$, and we do not include lens distortion parameters.

where $\mathbf{M}_{1,M}$ is the relative pose of frame M with respect to frame 1 as defined in Eq. 9. Rearranging Eq. 64 leads to

$$\mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} \hat{\mathbf{x}}_0 = \mathbf{M}_T \hat{\mathbf{x}}_T \quad (65)$$

where $\mathbf{M}_{1,M}$ and \mathbf{M}_B are already known. The substitution of $\tilde{\mathbf{x}}_0 = \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} \hat{\mathbf{x}}_0$ in Eq. 65 leads to

$$\tilde{\mathbf{x}}_0 = \mathbf{M}_T \hat{\mathbf{x}}_T \quad (66)$$

which can be solved for \mathbf{M}_T using $N \geq 3$ point correspondences. Finding point correspondences $\hat{\mathbf{x}}_T$ and $\tilde{\mathbf{x}}_0$ can be done using back-projection of pixels into the camera space. Let $S_u = \{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ be a set of 2D image points using

$$\mathbf{u}_1 = \mathbf{c} \quad \mathbf{u}_2 = \mathbf{c} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \mathbf{u}_3 = \mathbf{c} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (67)$$

where \mathbf{c} is the principal point of the camera. The image coordinates given in S_u are back-projected by using the API function given in Eq. 12

$$\mathbf{x}_i = \mathcal{F}^{(\text{CAM})}(\varphi_j, \theta_j, \mathbf{u}_i, d_{\text{CAM}}) \quad (68)$$

using $d_{\text{CAM}} = 1$ and constant parameters $\{\varphi_j, \theta_j\}$ for the set S_u .

The measured points are then converted to a local coordinate frame using \mathbf{x}_1 as origin

$$\left. \begin{aligned} \hat{\mathbf{x}}_1 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} [\mathbf{x}_1 \ 1]^T \\ \hat{\mathbf{x}}_2 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} \left[\mathbf{x}_1 + \frac{\mathbf{x}_2 - \mathbf{x}_1}{\|\mathbf{x}_2 - \mathbf{x}_1\|} \ 1 \right]^T \\ \hat{\mathbf{x}}_3 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} \left[\mathbf{x}_1 + \frac{\mathbf{x}_3 - \mathbf{x}_1}{\|\mathbf{x}_3 - \mathbf{x}_1\|} \ 1 \right]^T \end{aligned} \right\} \quad (69)$$

The rigid transformation $\mathbf{M}_T^{(\text{CAM})}$ can be extracted according to Section 3.1.6 using the two point sets

$$S_a = \left\{ \mathbf{c}_t, \mathbf{c}_t + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathbf{c}_t + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right\} \quad (70)$$

$$S_b = \{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \tilde{\mathbf{x}}_3\} \quad (71)$$

where $\tilde{\mathbf{x}}_i$ is the Euclidean representation of the homogeneous coordinate $\hat{\mathbf{x}}_i$ given in Eq. 69 and \mathbf{c}_t is given by $[0 \ 0 \ 1]^T$.

A constant distance $d_{\text{CAM}} = 1$ for all back-projected points in Eq. 68 will lead to a systematic error, which is negligible for the tool transform estimation. This can be shown by calculating the corrected length $d_{\text{CAM},i}$ of the back projected ray for the image coordinates $\mathbf{u}_{i,i \in \{2,3\}}$ in Eq. 67. Assuming \mathbf{u}_1 is aligned with the principal ray and the focal length f is known, the ray length $\tilde{d}_{\text{CAM},i}$ is given by triangle similarity and the Pythagorean theorem according to

$$\tilde{d}_{\text{CAM},i} = \sqrt{d_1^2 + b_{u,i}^2} \quad \frac{b_{u,i}}{d_1} = \frac{\|\mathbf{u}_1 - \mathbf{u}_i\|}{f} \quad (72)$$

$$\tilde{d}_{\text{CAM},i} = d_1 \sqrt{1 + \left(\frac{\|\mathbf{u}_1 - \mathbf{u}_i\|}{f} \right)^2} \quad (73)$$

where $\|\mathbf{u}_1 - \mathbf{u}_i\|$ describes the distance of two pixels $\{\mathbf{u}_1, \mathbf{u}_i\}$ in the image space and $\{b_{u,i}, d_1, \tilde{d}_{\text{CAM},i}\}$ define a right-angled triangle. For $f \gg \|\mathbf{u}_1 - \mathbf{u}_i\|$ and $d_1 = d_{\text{CAM}} = 1$ we can set $\tilde{d}_{\text{CAM},i} \approx d_i$. Figure 7b shows the simplified relationship of back projected rays with respect to the principal ray.

3.1.11 EDM Tool Frame

Similar to the camera tool, the pose of the EDM end effector can be described by a six DOF Euclidean transformation. Distance measurements can be modelled by a 3D ray in the Euclidean space. This leads to a tool pose defined up to an arbitrary rotation around the measurement axis. However, it is convenient to align the EDM frame with the camera frame.

The pose of the EDM end effector can be defined by a ray which describes the distance measurement in 3D, a related orthogonal vector, and the right-handed coordinate system constraint. The end effector position, which defines a point on the ray, was already used for the DH parameter extraction.

We define the **z** axis of the EDM as the distance measurement direction. By extending the set of back-projected distances during the measurement flow for the (DH) parameters, the **z** direction of the end effector with respect to the reference frame can be extracted.

A 1D distance measurement can be back-projected using

$$\hat{\mathbf{x}}_i = \mathbf{M}_{B,T}^{(\text{EDM})} \hat{\mathbf{d}}_i \quad (74)$$

$$\mathbf{M}_{B,T}^{(\text{EDM})} = \mathbf{M}_B \mathbf{M}_{1,M} \mathbf{M}_T^{(\text{EDM})} \quad (75)$$

where $\hat{\mathbf{x}}_i$ is a homogeneous point in the reference frame, and $\hat{\mathbf{d}}_i$ is a homogeneous point in the (EDM) space. The homogeneous point $\hat{\mathbf{d}}_i$ for a distance measurement d_i along the **z** axis of the (EDM) frame is defined as

$$\hat{\mathbf{d}}_i = [0 \ 0 \ d_i \ 1]^T \quad (76)$$

Rearranging Eq. 74 leads to

$$\mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} \hat{\mathbf{x}}_i = \mathbf{M}_T^{(\text{EDM})} \hat{\mathbf{d}}_i \quad (77)$$

which is of the same form as Eq. 65. We construct three correspondences for solving (74) for the rigid transform $\mathbf{M}_T^{(\text{EDM})}$. Let $S_d = \{d_1, d_2\}$ be a set of two distances where $d_1 = 1$ and $d_2 = 2$. The back-projection of the distances d_i to 3D points in the reference frame can be done with the API function given in Eq. 11

$$\mathbf{d}_i = \mathcal{F}^{(\text{EDM})}(\varphi_j, \theta_j, d_i) \quad (78)$$

using constant parameters $\{\varphi_j, \theta_j\}$ for the set S_d . The **x** direction of the (EDM) frame can be found by projecting

the \mathbf{x} direction⁸ of the camera frame onto the plane defined by point \mathbf{d}_1 and plane normal $\mathbf{n} = \mathbf{z}_{EDM} = \frac{\mathbf{d}_2 - \mathbf{d}_1}{\|\mathbf{d}_2 - \mathbf{d}_1\|}$

$$\mathbf{x}_{EDM} = \frac{\mathbf{x}_{CAM} - (\mathbf{z}_{CAM}^T \mathbf{z}_{EDM}) \mathbf{z}_{EDM}}{\|\mathbf{x}_{CAM} - (\mathbf{z}_{CAM}^T \mathbf{z}_{EDM}) \mathbf{z}_{EDM}\|} \quad (79)$$

The points are then converted to an intermediate coordinate frame with \mathbf{d}_1 as origin

$$\left. \begin{aligned} \hat{\mathbf{d}}_1 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} [\mathbf{d}_1 \ 1]^T \\ \hat{\mathbf{d}}_2 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} [\mathbf{d}_1 + \mathbf{x}_{EDM} \ 1]^T \\ \hat{\mathbf{d}}_3 &= \mathbf{M}_{1,M}^{-1} \mathbf{M}_B^{-1} [\mathbf{d}_1 + \mathbf{z}_{EDM} \ 1]^T \end{aligned} \right\} \quad (80)$$

with $M = 3$. The rigid transformation $\mathbf{M}_T^{(EDM)}$ can be extracted according to Section 3.1.6, where we define the two point sets as

$$S_a = \left\{ \mathbf{c}_t, \mathbf{c}_t + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathbf{c}_t + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\} \quad (81)$$

$$S_b = \left\{ \tilde{\mathbf{d}}_1, \tilde{\mathbf{d}}_2, \tilde{\mathbf{d}}_3 \right\} \quad (82)$$

where $\tilde{\mathbf{d}}_i$ is the Euclidean representation of the homogeneous coordinate $\hat{\mathbf{d}}_i$, and \mathbf{c}_t is given by $[0 \ 0 \ d_1]^T$.

4 Experimental Results for the Forward Kinematics Model of an Exemplary (RTS)

In this section, we apply the modeling method which we introduced in Section 3 to the HILTI PLT 300 [16]. To cover a wide spectrum of possible applications, we compare the DH results to the related spherical model, which represents simplified and idealized relationships between actuators and sensors, and the numerically optimized model of the exemplary RTS.

The RTS HILTI PLT 300 is a high precision measurement device which contains one EDM and supports RGB and infrared (IR) image and video streaming using a single camera. The device driver provides access of basic control and measurement variables, calibration values, data streams and seven manufacturer-side calibrated optical zoom positions of the camera. While the device can be geometrically described by standard kinematics models of theodolites and robotic total stations [38], no detailed kinematics model of the device is available.

⁸If the \mathbf{x} direction of the camera frame and the \mathbf{z} direction of the (EDM) frame are parallel, the \mathbf{y} direction of the camera frame can be used instead.

The data set for the DH model extraction of the PLT 300 contained 15 points for each joint, which have been recorded using the API functions introduced in Eqs. 11 and 12. End point positions for the first joint were recorded by setting the control variable θ to the fixed value $\theta = \frac{\pi}{2}$, while varying φ from 0 to 2π . End point positions for the second joint were recorded by setting the control variable φ to the fixed value $\varphi = 0$ while varying θ from 0 to π . The angular parameter space coverage is shown in Fig. 5b. For each position, the length of the back projected EDM ray and the length of the back-projected camera view ray were set to one meter ($d_1 = 1$). The pixel position for the back-projected camera view ray was aligned with the principal point of the camera, which was also provided by the API of the device. For EDM tool pose extraction, a second ray at each control position was extracted using a constant ray length of two meter ($d_2 = 2$). For camera tool pose extraction, two additional rays at each position were used, for which the back-projected image coordinates were shifted by one pixel in \mathbf{x} and one pixel in \mathbf{y} direction, respectively.

Table 1 shows the extracted DH parameters, Table 2 shows the base and tool transforms for both, the EDM and camera end effectors. The complete extracted kinematics model, including all coordinate frames and transformation matrices, is shown in Fig. 8. We show results for only one of the seven fixed zoom levels.

4.1 Model Simplification

The extracted model parameters in Section 4 are sufficient for building a device simulator. However, an idealized device simulation is often preferred. Idealized models can be used for analytic system and algorithm design and verification, for calculating or simulating desired system behaviours or for generating reference data sets.

The device used for this work shows similarities in the tool transforms, translation parameters, which are close to zero, and rotation parameters, which can be approximated by multiples of $\frac{\pi}{2}$. In this section, we apply numerical approximations to relate the extracted kinematics model with the simplified and idealized spherical model. We represent a rotation with a 3D rotation matrix according to

$$\mathbf{R} = \mathbf{R}_z(\gamma) \mathbf{R}_y(\beta) \mathbf{R}_x(\alpha) \quad (83)$$

where $\{\alpha, \beta, \gamma\}$ are the Euler angles [32].

Table 1 Estimated DH parameters for the PLT 300

joint	α_i	a_i	γ_i	d_i
1	1.5708	0.0	$\varphi_i - 1.5708$	0.0
2	0.0	0.0	θ_i	0.0

Table 2 Estimated base and end effector transforms for the PLT 300

transform	α	β	γ	t_x	t_y	t_z
\mathbf{M}_B	3.1416	0.0	0.0	0.0	0.0	0.0
$\mathbf{M}_T^{(\text{CAM})}$	-1.5020	1.5646	3.2103	-0.0003	0.0	0.0003
$\mathbf{M}_T^{(\text{EDM})}$	-1.5020	1.5646	3.2103	-0.0003	0.0	0.0003

By setting the translation parameters to 0 and replacing the rotational components by the nearest multiple of $\frac{\pi}{2}$, we get following forward kinematics model of the PLT 300:

$$\mathbf{M}_{B,T} = \mathbf{M}_B \mathbf{M}_{1,2} \mathbf{M}_{2,3} \mathbf{M}_T \quad (84)$$

$$\mathbf{M}_{1,2} = \mathbf{R}_z \left(\varphi_i - \frac{\pi}{2} \right) \mathbf{R}_x \left(\frac{\pi}{2} \right) \quad (85)$$

$$\mathbf{M}_{2,3} = \mathbf{R}_z(\theta_i) \quad (86)$$

$$\mathbf{M}_B = \mathbf{R}_x(\pi) \quad (87)$$

$$\mathbf{M}_T = \mathbf{R}_z(\pi) \mathbf{R}_y \left(\frac{\pi}{2} \right) \mathbf{R}_x \left(-\frac{\pi}{2} \right) \quad (88)$$

The rotation matrices \mathbf{R}_x and \mathbf{R}_z are defined in Eqs. 6 and 7, respectively. The rotation matrix \mathbf{R}_y is defined as

$$\mathbf{R}_y(\beta_i) = \begin{bmatrix} \cos \beta_i & 0 & \sin \beta_i & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \beta_i & 0 & \cos \beta_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (89)$$

Multiplying and simplifying matrix $\mathbf{M}_{B,T}$ leads to

$$\mathbf{M}_{B,T} = \begin{bmatrix} \cos(\varphi_i) & \cos(\theta_i) \sin(\varphi_i) & \sin(\varphi_i) \sin(\theta_i) & 0 \\ -\sin(\varphi_i) & \cos(\varphi_i) \cos(\theta_i) & \cos(\varphi_i) \sin(\theta_i) & 0 \\ 0 & -\sin(\theta_i) & \cos(\theta_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (90)$$

which can be written as an Euler rotation according to Eq. 83:

$$\mathbf{M}_{B,T} = \mathbf{R}_z(-\varphi_i) \mathbf{R}_y(0) \mathbf{R}_x(-\theta_i) \quad (91)$$

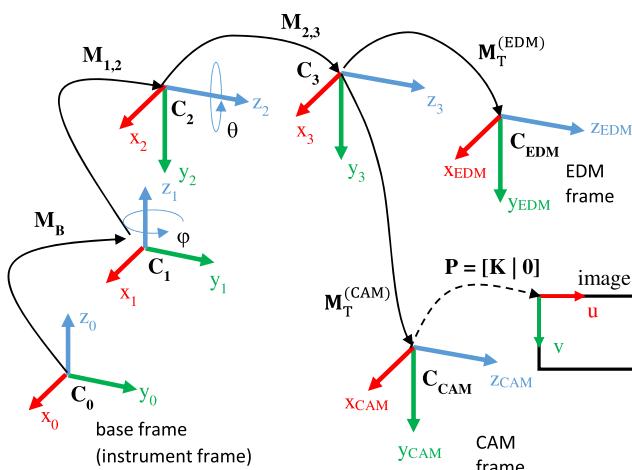


Fig. 8 Coordinate frame relations of the RTS PLT 300 using the DH convention

4.2 Model Error Estimation

The (DH) model error can be expressed using the average point distance \bar{d} and the unbiased standard deviation $\hat{\sigma}$ between recorded and calculated point sets according to

$$\bar{d} = \frac{\sum_{i=1}^N ||\mathbf{x}_{i,\text{meas}} - \mathbf{x}_{i,\text{calc}}||}{N} \quad (92)$$

$$\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^N (||\mathbf{x}_{i,\text{meas}} - \mathbf{x}_{i,\text{calc}}|| - \bar{d})^2}{N-1}} \quad (93)$$

where $\mathbf{x}_{i,\text{meas}}$ are points of the measurement set and $\mathbf{x}_{i,\text{calc}}$ are points of the calculated point set. The measured point sets for the (EDM) tool and for the camera tool have been created using Eqs. 11 and 12, respectively. The calculated point set for the (EDM) tool is given by Eq. 74.

The calculated point set for the camera tools can be derived from Eq. 62. A back-projected homogeneous image coordinate $\hat{\mathbf{u}}$ defines a view ray $\mathcal{X}_T(\hat{\mathbf{u}}, \lambda)$ in the reference frame [14]:

$$\mathcal{X}_T(\hat{\mathbf{u}}, \lambda) = \mathbf{P}^\dagger \cdot \hat{\mathbf{u}} + \lambda \cdot \mathbf{c} \quad (94)$$

where \mathbf{P}^\dagger is the pseudo-inverse of the projection matrix \mathbf{P} and \mathbf{c} is the camera center. The 3×4 projection matrix \mathbf{P} is defined by

$$\mathbf{P} = \mathbf{P}_{\text{CAM}} \mathbf{M}_{B,T}^{-1} \quad (95)$$

which projects points from the reference frame to the camera image.

While SVD can be used to reduce numerical instabilities for estimating the pseudo-inverse \mathbf{P}^\dagger [14], Eq. 94 has two further problems: 1. A camera center $\mathbf{c} = 0$ does not result in a valid ray equation, and 2. λ is not a linear parameterization of the ray length.

Therefore, the following alternative view ray calculation should be used for finite cameras [14]:

$$\mathcal{X}_T(\hat{\mathbf{u}}, \mu) = \frac{\mu}{||\mathbf{M}^{-1} \cdot \hat{\mathbf{u}}||} \begin{pmatrix} \mathbf{M}^{-1} \cdot \hat{\mathbf{u}} \\ 0 \end{pmatrix} + \begin{bmatrix} -\mathbf{M}^{-1} \cdot \mathbf{p}_4 \\ 1 \end{bmatrix} \quad (96)$$

where $\hat{\mathbf{u}}$ is a homogeneous coordinate vector, and μ is the length of the back-projected ray. The decomposition of the

Table 3 Evaluation taxonomy, showing the most suitable choice for different applications

evaluation aspect	variant	usage
end effector type	EDM only	non-vision based measurements
	camera only	vision-based measurements
	both (EDM, camera)	
end effector error	position only (A)	non-vision based measurements
	position and rotation around principal ray (B)	vision-based measurements
optimization	none	DH modeling workflow evaluation, start values for optimization, comparison with previous work evaluate influence of parameter space coverage
	reduced angle parameter space	
	complete angle parameter space	

projection matrix \mathbf{P} into the 3×3 matrix \mathbf{M} and the 3×1 column vector \mathbf{p}_4 is given by

$$\mathbf{P} = [\mathbf{M} | \mathbf{p}_4]. \quad (97)$$

Equation 96 can be used to calculate the evaluation point set for the camera tool frame, using the ray length $\mu = d_{\text{CAM}}$.

We analyzed different aspects of the modeling error and the influence of the individual end effectors. Traditionally, the camera of the (RTS) is used by the operator to measure a certain 3D point, but the actual measurement does not use image based measurement (IBM) methods. Scanning applications may not use the camera at all. Ideally, the \mathbf{z} axes of the camera and the EDM are aligned, the tool rotations around the principal ray and the EDM ray do not influence the result. For such applications, the tool rotations around their respective \mathbf{z} axis should be excluded in the error analysis (method A).⁹ Advanced applications use IBM methods; hence, errors of all tool poses have to be considered (method B). The different aspects of the evaluation are shown in Table 3.

The results in Table 4 show that the simplified, spherical model, compared to the geometrically extracted (DH)

Table 4 Kinematics model error of the extracted (DH) model, e_{DH} , and of the simplified model, $e_{\text{simplified}}$

method	end effector	$e_{\text{DH}} [m]$	$e_{\text{simplified}} [m]$
(A)	EDM	$8.16e^{-4} \pm 1.43e^{-6}$	$1.24e^{-17} \pm 3.39e^{-17}$
	camera	$2.79e^{-6} \pm 1.73e^{-6}$	$8.14e^{-4} \pm 1.15e^{-6}$
	compound	$4.09e^{-4} \pm 4.07e^{-4}$	$4.07e^{-4} \pm 4.07e^{-4}$
(B)	EDM	$1.04e^{-3} \pm 2.19e^{-4}$	$3.80e^{-17} \pm 8.05e^{-17}$
	camera	$1.65e^{-4} \pm 1.15e^{-4}$	$4.38e^{-3} \pm 2.52e^{-3}$
	compound	$5.13e^{-4} \pm 4.57e^{-4}$	$2.63e^{-3} \pm 2.90e^{-3}$

The individual and compound errors for the end effectors are provided. (A) Tool rotations around principal ray and (EDM) ray ignored. (B) Tool rotations around principal ray and (EDM) ray considered

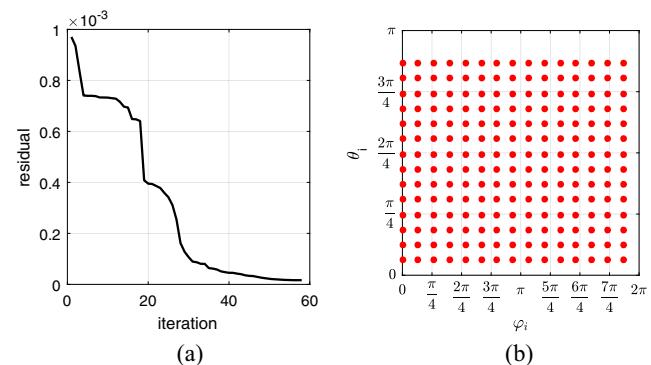
model, is a good approximation for the (EDM) end effector; the (DH) model shows lower modeling error for the complete system.

4.3 Model Optimization

The kinematics modeling method proposed in Section 3 is a greedy algorithm, which only optimizes local cost functions. While this is sufficient for many applications, the estimated model can be refined using nonlinear optimization techniques to decrease the error of the model. In this section, we briefly discuss model optimization using a global error function. However, this is only a proof of concept, while a detailed optimization analysis is beyond the scope of this work.

Finding an optimal model can be formalized as nonlinear optimization problem with boundary conditions and linear scalarization

$$\min_{S_o} (\omega_d \bar{d}_d + \omega_\sigma \hat{\sigma}_d) \quad \left. \begin{array}{l} t_{\min} < t_i < t_{\max} \\ r_{\min} < r_i < r_{\max} \end{array} \right\} \quad (98)$$

**Fig. 9** **a** Residual evolution of SQP optimization. **b** Angle control parameter space for cross-validation

⁹Previous work in (DH) modeling does not address the complete tool pose, but only the position of the end effector [2, 4, 15, 28, 31].

Table 5 Errors of optimized model, with tool rotations around principal ray and EDM ray considered (method B)

end effector	$e_{\text{optimized}} [m]$	$e_{\text{optimized,cross}} [m]$	$e_{\text{optimized,full}} [m]$
EDM	$1.72e^{-5} \pm 5.19e^{-6}$	$2.12e^{-5} \pm 7.64e^{-6}$	$1.99e^{-5} \pm 6.88e^{-6}$
camera	$1.98e^{-5} \pm 8.20e^{-6}$	$2.16e^{-5} \pm 9.50e^{-6}$	$1.41e^{-5} \pm 7.79e^{-6}$
compound	$1.87e^{-5} \pm 7.26e^{-6}$	$2.15e^{-5} \pm 8.81e^{-6}$	$1.64e^{-5} \pm 7.95e^{-6}$

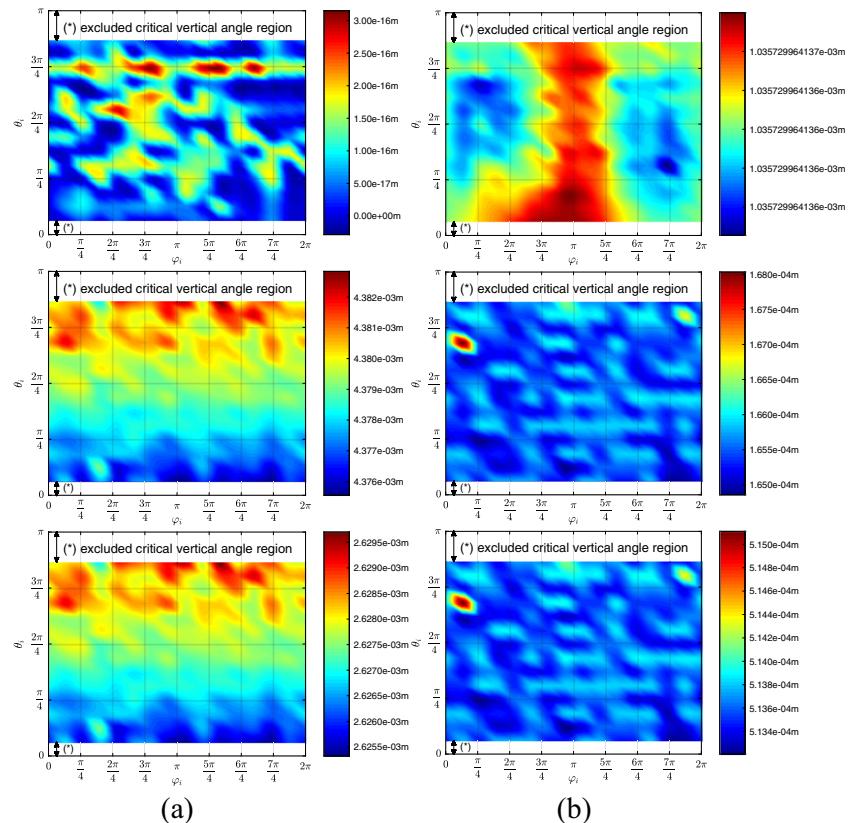
Table 6 DH parameter results of SQP optimization

i	α_i	a_i	γ_i	d_i
1	1.570808	0.0	$\varphi_i - 1.5203$	0.0
2	$0.2055e^{-3}$	$-0.1117e^{-3}$	$\theta_i - 0.4432e^{-3}$	$0.0684e^{-3}$

Table 7 Base and tool transformation results of SQP optimization

transform	α	β	γ	t_x	t_y	t_z
\mathbf{M}_B	3.1416	0.0	0.0504	$5.915e^{-9}$	$7.216e^{-9}$	$3.312e^{-9}$
$\mathbf{M}_T^{(\text{CAM})}$	-1.4632	1.5646	3.2103	$-0.1451e^{-3}$	$-2.152e^{-9}$	$0.1505e^{-3}$
$\mathbf{M}_T^{(\text{EDM})}$	-1.4219	1.5693	3.2910	$0.1421e^{-3}$	$1.415e^{-6}$	$-9.014e^{-5}$

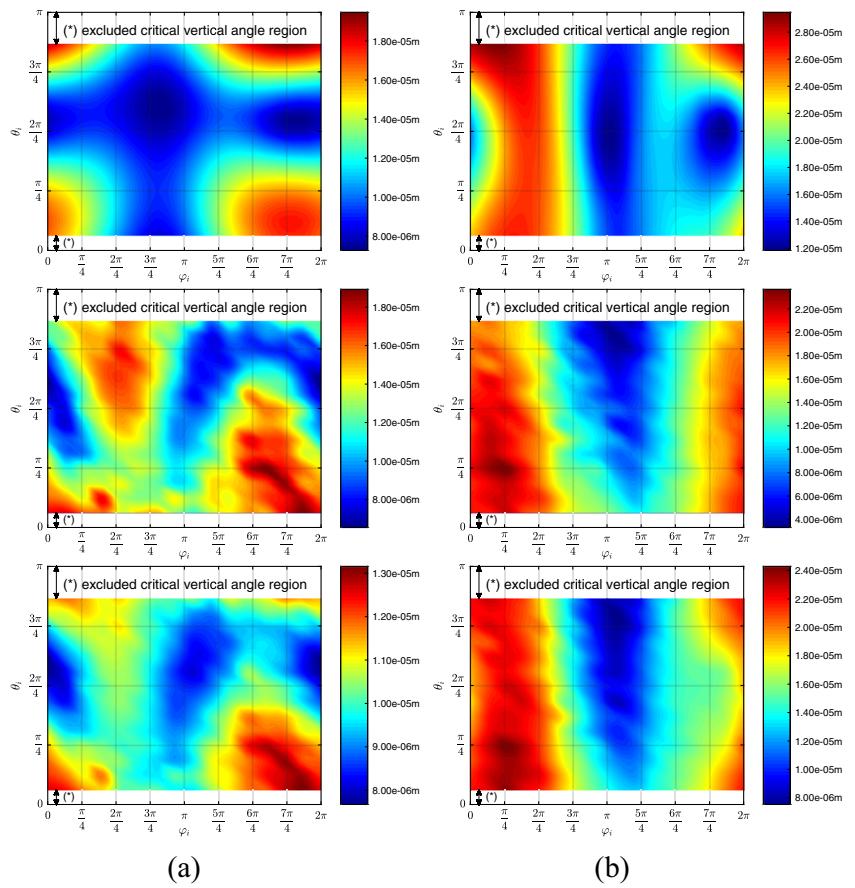
Fig. 10 Modeling error distribution with respect to the angle control parameter space, using Eq. 92 for each control point $\{\varphi_i, \theta_i\}$. **a** Simplified model. **b** Geometrically extracted model. End effector error (method B), encoded as color: EDM (top row), camera (middle row), compound (bottom row)



(a)

(b)

Fig. 11 Modeling error distribution with respect to the angle control parameter space, using Eq. 92 for each control point $\{\varphi_i, \theta_i\}$. **a** Optimized with reduced parameter space. **b** Optimized with full parameter space. End effector error (method B), encoded as color: EDM (top row), camera (middle row), compound (bottom row)



where ω_d and ω_σ are weighting factors and where the mean value \bar{d}_d and the unbiased standard deviation $\hat{\sigma}_d$ are given in Eqs. 92 and 93, respectively. The set S_o contains the 26 model parameters:¹⁰

$$S_o = \{\mathbf{r}_B, \mathbf{t}_B, Q_1, Q_2, \mathbf{r}_T^{(CAM)}, \mathbf{t}_T^{(CAM)}, \mathbf{r}_T^{(EDM)}, \mathbf{t}_T^{(EDM)}\} \quad (99)$$

The two sets Q_1 and Q_2 contain the eight (DH) parameters of the model. The rigid base transform is described by the 3×1 vector \mathbf{r}_B , containing the Euler angles, and the 3×1 translation vector \mathbf{t}_B . Analogously, $\{\mathbf{r}_T^{(CAM)}, \mathbf{t}_T^{(CAM)}\}$ and $\{\mathbf{r}_T^{(EDM)}, \mathbf{t}_T^{(EDM)}\}$ describe the rigid end effector transforms. We used the lower boundary $t_{\min} = -1e^{-2}m$ and the upper boundary $t_{\max} = 1e^{-2}m$ for all translational model parameters. Additionally, we used the lower boundary $r_{\min} = -\pi$ and the upper boundary $r_{\max} = +\pi$ for all rotational model parameters.¹¹

¹⁰We do not include the camera intrinsics, (EDM) calibration parameters or scale factors of the system control parameters $\{\varphi, \theta\}$.

¹¹We mapped the result to the range $[0, 2\pi]$ for better comparison between optimized and non-optimized model.

Sequential quadratic programming (SQP) was used to refine the initial kinematics model, which is a gradient based iterative numerical optimization method. Details about SQP can be found in the book by Nocedal and Wright [26].

Usually the weights of the objectives are normalized to one, hence $\omega_d + \omega_\sigma = 1$ [10]. In this work, however, we weighted the mean value with $\omega_d = 1.0$ and the unbiased standard deviation with $\omega_\sigma = 1.0$ for better comparison between the non-optimized result given in Table 4 and the evolution of the residual over the optimization iterations shown in Fig. 9a.

The optimization was implemented and executed with MATLAB 2017 and the MATLAB Optimization Toolbox [24] using four parallel sub-processes, Microsoft Windows 10, an Intel Core i7 processor with 64GB RAM. The runtime of the SQP based optimization was $1.326e^3 s$ ($\approx 0.4h$).

The error results of the optimized model $e_{\text{optimized}}$ are shown in Table 5; the optimized model parameters are shown in Tables 6 and 7, respectively. Error distributions with respect to the angle control parameter space are shown in Figs. 10 and 11, respectively.¹²

¹²The high dynamic range of the modeling errors does not allow for a common heat map encoding.

5 Discussion

The model error $e_{\text{optimized}}$ of Table 5 was calculated using the reduced control parameter set $\{\varphi_i, \theta_i\}$ as shown in Fig. 5b for both, optimization and evaluation. For simple cross-validation, the optimized model was applied to all samples of the full control parameter space, as shown in Fig. 9b, which corresponds to the error $e_{\text{optimized,cross}}$ in Table 5. Cross-validation is explained in the book by Witten and Frank [40]. In Table 5, $e_{\text{optimized,full}}$ shows the error of the optimized model, where the full control parameter space for model fitting and validation was used. The errors for all three methods $\{e_{\text{optimized}}, e_{\text{optimized,cross}}, e_{\text{optimized,full}}\}$ were calculated using Eqs. 92 and 93. The error values of all three methods are in the same order of magnitude, which shows that the sub-set of the recorded samples is sufficient for RTS model optimization. However, a slight decrease in the modeling error can be observed when using samples from the full angle control parameter space.¹³

The comparison of Table 4 with Table 5 shows a decrease in the modeling error by one order of magnitude when applying numerical optimization subsequent to the geometric DH parameter estimation. Hence, numerical optimization is essential for kinematics modeling.

Figures 10 and 11 show the distribution of the modeling error with respect to the angle control parameter space for the non-optimized and the optimized model, respectively. Sample points in the *critical vertical angle regions* were excluded throughout this work for stability reasons. In Fig. 11, samples of the full control parameter space were used for optimization and validation. The critical regions exclude samples near the poles of the spherical model from the calculations. The poles are defined by $\theta_i = \{0, \pi\}$. In particular, the critical region around $\theta_i = \pi$ also excludes the non-measurable area of a physical (RTS), as shown in Fig. 12. In this work, we set critical regions to $|\theta_i| = \{0, \frac{3}{4}\pi \dots \pi\}$.

All calculations were carried out with 64-bit precision arithmetic. In particular, a 64-bit value has a precision of 16 decimal digits. The integer part of the values of all distances and angles used in this work can be represented with one decimal digit; the fractional parts are represented by 15 decimal digits. Details about floating point arithmetic are given in the IEEE 754-2008 standard for floating-point arithmetic [19]. The range of the EDM error distribution of the simplified model, shown in the top left diagram of Fig. 10, is in the magnitude of the round-off effects of the 64-bit floating point arithmetic. Hence, the error distribution can be considered as noise, introduced by round-off effects. The error distribution of the geometrically estimated DH

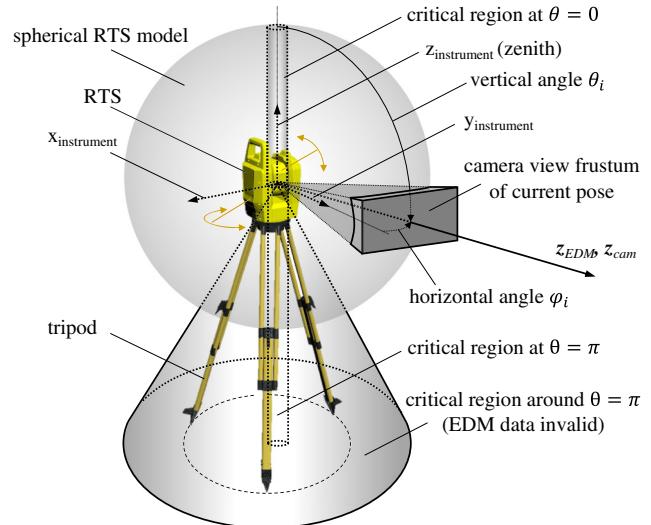


Fig. 12 Critical vertical angle regions of an RTS

model of the EDM end effector is shown in the top right diagram of Fig. 10; accuracy and precision are in the magnitude of $1e - 3m$ and $1e - 16m$, respectively. The distribution indicates an EDM pose error of the model, which appears as increasing error between $|\varphi| = \{0 \dots \pi\}$ when applying horizontal rotations. The approximately uniform distribution over the vertical parameter space $\theta = \{0 \dots \pi\}$ indicates that the major offset is along the y-axis of the instrument frame. A scale and offset error of the angle control parameter φ could also lead to a similar error distribution, but is contradicted by the error analysis of the simplified EDM model.

For each model in Figs. 10 and 11, the error distributions of the camera end effector are given in the middle row; the compound error distributions are given in the bottom row.

When comparing all analyzed models, the simplified model is the best match for the device, if only the EDM end effector is considered. In particular, the EDM modeling error distribution, shown in the top left diagram of Fig. 10, is in the magnitude of numerical round-off effects. This indicates that the driver uses a spherical coordinate system to convert angle and distance sensor data $\{\varphi_i, \theta_i, d_i\}$ to Euclidean points \mathbf{x}_i .

When only considering the camera end effector, the accuracy of the geometrically estimated model is by a order lower, compared to the simplified model; the error distribution is more uniform with respect to the angle parameter space. The distribution of the simplified model indicates a translational component of the camera pose, which cannot be modeled by a single spherical coordinate system.

The compound errors of the individual models are mainly influenced by sample points of the camera frame. This indicates that the camera model used in this work is too

¹³The analysis of the effect of using a reduced distance parameter space is beyond the scope of this work.

simplistic. To lower the modeling accuracy and precision, a more general camera model can be applied.

The results presented in Section 4 show that a spherical representation of the RTS is sufficient for idealized kinematics simulation of the system. If a more detailed model is required, the system can be described by DH parameters using the method we introduced in Section 3. The results proposed in Section 4.3 show the significance of the downstream numerical optimization.

While the analyzed models cover a wide range of possible applications, additional aspects can be addressed to improve the simulation of the device properties:

Optical Zoom Levels An RTS usually provides different optical zoom levels. The exemplary device used for the project supports seven optical zoom levels [16]. In this work, only the first optical zoom level was presented.

A more detailed description would include individual tool matrices $\mathbf{M}_T^{(k)}$ and projection matrices $\mathbf{P}^{(k)}$ for each zoom level.

Faces An RTS supports measurement in both *faces* to reduce systematic measurement errors [38]. Changing the face for a particular measurement from face one to face two means to change the control variables by $\varphi_2 = \varphi_1 - \pi$ and $\theta_2 = -\theta_1$, where $\{\varphi_1, \theta_1\}$ and $\{\varphi_2, \theta_2\}$ are the control variables for face one and face two, respectively. Depending on the particular device specific API, face control could be either coded into the control variables or accessed explicitly. Hence, either an unique kinematics model for both faces or a separate kinematics model for each face has to be applied. In this work, we used only one face for model estimation. Furthermore, the image orientation of the second face might be corrected implicitly by the device driver or explicitly by the application software.

Simulation of Complete Model A *complete* kinematics model allows arbitrary rigid placement of the robot base and tools [11]. The complete kinematics description used in this work requires base and tool transforms in addition to the (DH) parameters, as proposed by Veitschegger and Wu [39]. However, some existing kinematics simulators might not support this extension without further modifications [27]. For end-to-end simulation, it might be required to express base and tool transforms as (DH) joints.

Nonlinear Kinematics Model Including non-linear system properties could further increase the model accuracy. For example, signal conditioning for angular control variables can be included in the circular feature extraction method of Section 3 or in the numerical optimization method of Section 4.3.

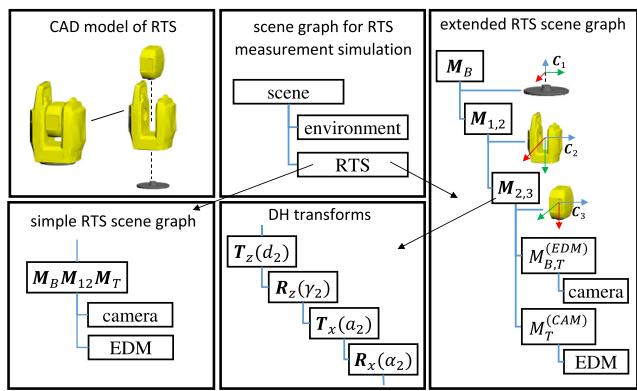


Fig. 13 Exemplary scene graphs for RTS simulators, including following nodes: Scene node (reference frame), environment node (measurement targets), RTS (device model). Simple scene graph: no visualization of the kinematics of the individual RTS components. Extended scene graph: DH parameters can be applied to the individual RTS components

Dynamic Model of the Robot The RTS prototype used in this work provides only limited access to dynamic control parameters, hence no dynamic model was derived. We will further work on modeling the dynamic behavior of the PLT 300.

Alternative Error Formulation The model error used in this work is based on Euclidean distances between recorded and calculated point sets. A more detailed error formulation could separate translational and angular errors.

The estimated models can be used for RTS simulation, using custom or standard robotic simulators. Figure 13 shows the scene graph of an exemplary RTS simulator, Fig. 14 shows a custom RTS simulator in Unity3D. An early version of the simulator was used by Klug et al. for the design of interactive RTS algorithms [21]. A detailed description of the simulator is beyond the scope of this work.

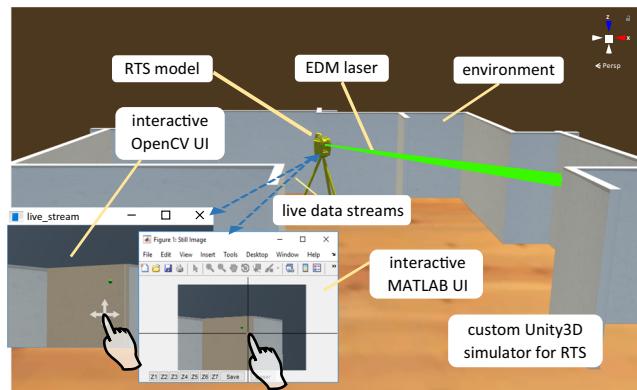


Fig. 14 Exemplary RTS simulator in Unity3D. Klug et al. [21] used the simplified model for the design of interactive RTS algorithms

6 Conclusion

In this work, we presented a complete workflow for forward kinematics modeling of an RTS using the DH convention. The workflow is tailored to RTS and includes a detailed formalization and description of all individual steps.

Furthermore, we provided the forward kinematics model of the PLT 300, for which we validated the influence of simplifying and optimizing the extracted kinematics model. The applied error analysis clearly shows the significance of the downstream optimization.

The choice between simplified and optimized model depends on the application. The simplified model represents an ideal device without systematic errors, which can be used for basic algorithm design, development and verification. A typical application of this model is the error propagation analysis [38]. The optimized model matches the analyzed device more precisely. Typical applications of this model are device simulation, test data generation and conversion of sensor data. By using (DH) parameters, existing robotic simulators could be used, or custom simulation environments could be implemented.

The modeling error distribution shown in Figs. 10 and 11 indicates that further enhancements could be achieved by applying more general camera models.

In general, we believe that the DH parameters and the proposed workflow is feasible for RTS simulation. We will further work on including more complete models of the sensors, modeling dynamics of RTS, and on comparing the usability of existing simulators for RTS simulation.

Acknowledgements This work was enabled by the Competence Center VRVis.

VRVis is funded by BMVIT, BMWFW, Styria, SFG and Vienna Business Agency under the scope of COMET - Competence Centers for Excellent Technologies (854174) which is managed by FFG.

Funding Information Open access funding provided by Graz University of Technology.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

1. Arun, K.S., Huang, T.S., Blostein, S.D.: Least-squares fitting of two 3-D point sets. *IEEE Trans. Pattern Anal. Mach. Intell. PAMI* **9**(5), 698–700 (1987). <https://doi.org/10.1109/TPAMI.1987.4767965>
2. Barker, L.K.: Vector-algebra approach to extract Denavit-Hartenberg parameters of assembled robot arms. In: NASA STI. NASA-TP-2191, L-15621, NAS, vol. 1, no. 60, p. 2191. NASA Langley Research Center, Hampton (1983)
3. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: USARSim: a robot simulator for research and education. In: Proc. 2007 IEEE Int. Conf. Robot. Autom., pp. 1400–1405 (2007). <https://doi.org/10.1109/ROBOT.2007.363180>
4. Chittawadigi, R.G., Hayat, A.A., Saha, S.K.: Geometric model identification of a serial robot. *IFToMM Int. Symp. Robot. Mechatronics*, vol. 3 (2013)
5. Connolly, C.: Technology and applications of ABB RobotStudio. *Ind. Robot An Int. J.* **36**(6), 540–545 (2009). <https://doi.org/10.1108/01439910910994605>
6. Crossley, F.R.E.: The Permutations of Kinematic Chains of Eight Members Or less from the Graph Theoretic Viewpoint. In: Shaw, W.A. (ed.) *Dev. Theor. Appl. Mech.*, vol. 2, pp. 467–486. Pergamon Press, Oxford (1964)
7. Delp, S.L., Anderson, F.C., Arnold, A.S., Loan, P., Habib, A., John, C.T., Guendelman, E., Thelen, D.G.: OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE Trans. Biomed. Eng.* **54**(11), 1940–1950 (2007). <https://doi.org/10.1109/TBME.2007.901024>
8. Denavit, J., Hartenberg, R.S.: A kinematic notation for lower-pair mechanisms based on matrices. *Trans. ASME. J. Appl. Mech.* **22**, 215–221 (1955)
9. Eggert, D., Lorusso, A., Fisher, R.: Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Mach. Vis. Appl.* **9**(5–6), 272–290 (1997). <https://doi.org/10.1007/s001380050048>
10. Ehrgott, M.: Multicriteria Optimization. Lecture Notes in Economics and Mathematical Systems. Springer, Berlin (2005)
11. Everett, L., Driels, M., Mooring, B.: Kinematic modelling for robot calibration. In: Proc. 1987 IEEE Int. Conf. Robot. Autom., vol. 4, pp. 183–189 (1987). <https://doi.org/10.1109/ROBOT.1987.1087818>
12. Harrisberger, L.: A number synthesis survey of three-dimensional mechanisms. *J. Eng. Ind.* **87**(2), 213–218 (1965)
13. Hartenberg, R.S., Denavit, J.: Kinematic Synthesis of Linkages. McGraw-Hill Series in Mechanical Engineering McGraw-Hill (1964)
14. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2003)
15. Hayat, A.A., Chittawadigi, R.G., Udai, A.D., Saha, S.K.: Identification of Denavit-Hartenberg parameters of an industrial robot. In: Proc. Conf. Adv. Robot., AIR '13, pp. 1–6. ACM, New York (2013). <https://doi.org/10.1145/2506095.2506121>
16. Hilti Corporation: PLT 300. <https://www.hilti.com/measuring-systems/optical-tools/r4728599> [Online; Accessed 28] (2017)
17. Horn, B.K.P.: Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A* **4**(4), 629 (1987). <https://doi.org/10.1364/JOSAA.4.000629>
18. Horn, B.K.P., Hilden, H.M., Negahdaripour, S.: Closed form solutions of absolute orientation using orthonormal matrices. *J. Opt. Soc.* **5**, 1127–1135 (1988)
19. IEEE Std 754-2008: IEEE standard for floating-point arithmetic (Revision of IEEE Std 754-1985). Standard, IEEE Computer Society (2008)
20. Kabsch, W.: A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. Sect. A* **32**(5), 922–923 (1976). <https://doi.org/10.1107/s0567739476001873>
21. Klug, C., Schmalstieg, D., Arth, C.: Measuring human-made corner structures with a robotic total station using support points, lines and planes. In: Proc. 12th Int. Jt. Conf. Comput. Vision, Imaging Comput. Graph. Theory Appl. - Vol. 6 VISAPP, (VISIGRAPP 2017), pp. 17–27. INSTICC, SciTePress (2017). <https://doi.org/10.5220/0006096800170027>

22. Koenig, N., Howard, A.: Design and use paradigms for Gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), vol. 3, pp. 2149–2154 (2004). <https://doi.org/10.1109/IROS.2004.1389727>
23. Kucuk, S., Bingul, Z.: Robot kinematics: Forward and inverse kinematics. In: Ind. Robot. Theory, Model. Control. Pro Literatur Verlag, Germany / ARS, Austria (2006). <https://doi.org/10.5772/5015>
24. MathWorks Inc.: MATLAB optimization toolbox. <https://mathworks.com/products/optimization.html> [Online; Accessed 28] (2017)
25. McCarthy, J.M., Soh, G.S. Geometric Design of Linkages, 2nd edn. Springer, New York (2011). Interdisciplinary applied mathematics
26. Nocedal, J., Wright, S.: Numerical Optimization. Springer Series in Operations Research and Financial Engineering. Springer, New York (2006)
27. Rajeevlochana, C.G., Saha, S.K.: RoboAnalyzer : 3D model based robotic learning software. In: Int. Conf. Multi Body Dyn. vol. 2011, pp. 3–13. KL University, Vijayawada (2011)
28. Rajeevlochana, C.G., Saha, S.K., Kumar, S.: Automatic extraction of DH parameters of serial manipulators using line geometry. Int. Conf. Multibody Syst Dyn (2012)
29. Reuleaux, F.: The kinematics of machinery: Outlines of a theory of machines macmillan and CO (1876)
30. Rohmer, E., Singh, S.P.N., Freese, M.: V-REP: A versatile and scalable robot simulation framework (2013). 2013 IEEE/RSJ Int. Conf. Intell. Robot. Syst., pp. 1321–1326. <https://doi.org/10.1109/IROS.2013.6696520>
31. Saha, S.K.: Introduction to Robotics, 2nd edn, p. 599. McGraw Hill, New Delhi (2014). ISBN:9789332902817
32. Schneider, P., Eberly, D.: Geometric Tools for Computer Graphics. Boston Morgan Kaufmann Publishers, Amsterdam (2003)
33. Sheth, P.N., Uicker, J.J.: A generalized symbolic notation for mechanisms. *J. Eng. Ind.* **93**(1), 102–112 (1971)
34. Siciliano, B., Khatib, O.: Springer Handbook of Robotics. Springer Handbooks Springer International Publishing (2016)
35. Siemens PLM Software: Robcad tecnomatix. <https://www.plm.automation.siemens.com/en/products/tecnomatix/manufacturing-simulation/robotics/robcad.shtml> (2011) [Online; Accessed 28] (2017)
36. Stone, H.W.: Kinematic Modeling, Identification, and Control of Robotic Manipulators. The Springer International Series in Engineering and Computer Science. Springer, New York (2012)
37. Tonello, S., Zanetti, G.P., Finotto, M., Bortoleto, R., Tosello, E., Menegatti, E.: WorkCellSimulator: A 3D simulator for intelligent manufacturing. In: Proc. 3rd Int. Conf. Simulation, Model. Program. Auton. Robot., SIMPAR'12, pp. 311–322. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-34327-8_29
38. Uren, J., Price, B.: Surveying for Engineers. Palgrave Macmillan, Basingstoke (2010)
39. Veitschegger, W.K., Wu, C.H.: Robot calibration and compensation. *IEEE Trans. Robot. Autom.* **4**(6), 643–656 (1988)
40. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Elsevier, San Francisco. The Morgan Kaufmann Series in Data Management Systems, ISBN: 9780080477022 (2005)

Christoph Klug received the M.S. degree in Telematics from the Technical University of Graz (2012). After suspending his university career to work in the Automotive industry as an embedded software designer for several years, he is now working as a research assistant at the Virtual Reality and Visualisation Research Centre GmbH (VRVis) in cooperation with the Technical University of Graz, where he is finishing his Ph.D. degree in computer sciences. He is currently putting all his energy in enhancing workflows and algorithms for robotic total stations using Computer Vision methods with special regard to usability and accuracy. Further research interests lie in Computer Vision and Graphics, geometric modeling and human-computer interaction.

Dieter Schmalstieg is full professor and head of the Institute of Computer Graphics and Vision at Graz University of Technology, Austria. His current research interests are augmented reality, virtual reality, computer graphics, visualization and human-computer interaction. He received Dipl.-Ing. (1993), Dr. techn. (1997) and Habilitation (2001) degrees from Vienna University of Technology. He is author and co-author of over 300 peer-reviewed scientific publications with over 13,000 citations, with over twenty best paper awards and nominations. His organizational roles include associate editor in chief of IEEE Transactions on Visualization and Computer Graphics, member of the editorial advisory board of computers & graphics and of the Springer Virtual Reality journal, member of the steering committee of the IEEE International Symposium on Mixed and Augmented Reality, chair of the EUROGRAPHICS working group on Virtual Environments (1999–2010), key researcher of the K-Plus Competence Center for Virtual Reality and Visualization in Vienna and key researcher of the Know-Center in Graz. In 2002, he received the START career award presented by the Austrian Science Fund. In 2012, he received the IEEE Virtual Reality technical achievement award for seminal contributions to the field of Augmented Reality. He was elected as a senior member of IEEE, as a member of the Austrian Academy of Sciences and as a member of the Academia Europaea. In 2008, he founded the Christian Doppler Laboratory for Handheld Augmented Reality.

Thomas Gloor received his PhD in theoretical physics from Ecole Polytechnique Fédérale de Lausanne (EPFL) in 2005. Since 2009 he is a permanent R&D engineer at Hilti corporation. His main interest are computer vision algorithms, digital signal processing and software engineering.

Dr. Clemens Arth is senior scientist at Graz University of Technology and CEO/Founder of AR4 GmbH. He received Dipl.-Ing. (2004) and Dr. techn. (2008) degrees from Graz University of Technology. He is author and co-author of numerous peer-reviewed publications at ISMAR, VISAPP, CVPR, ACCV, ICPR and VR, while he also authored more than 15 patents and patent applications. His main research interests are Computer Vision algorithms, Augmented Reality technology and Machine Learning.