Robotics and Computer–Integrated Manufacturing 98 (2026) 103148

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length Article

From perception to precision: Vision-based mobile robotic manipulation for assembly screwdriving

Aleksandar Stefanov *, Miha Zorman, Sebastjan Šlajpah, Janez Podobnik, Matjaž Mihelj, Marko Munih

University of Ljubljana, Faculty of Electrical Engineering, Slovenia

ARTICLE INFO

Keywords:
Mobile manipulation
Vision-based control
Screwdriving
Robotics
Perception algorithms
Assembly automation

ABSTRACT

Flexible manufacturing demands automation that is both precise and adaptable. However, tasks such as screwdriving are typically automated using costly, rigid robotic cells, making this approach impractical for low-volume, high-mix production. As a scalable solution, mobile manipulators offer a flexible alternative, but achieving the required precision for screwdriving remains challenging due to localization uncertainties. This paper addresses these limitations by presenting a vision-guided mobile robotic manipulation system that performs high-precision screwdriving using only monocular RGB imagery. The proposed pipeline integrates stationary and onboard cameras with perception algorithms for object identification and segmentation, pose estimation, and CAD-based screw hole localization, compensating for base misalignment and object placement variability. Experimental validation using ISO 9283 standard's metrics demonstrates a translational accuracy between 0.21 mm and 0.50 mm across multiple screw positions. Additionally, the system achieves angular estimation errors as low as 0.07° to 0.20°, verifying its capability for sub-degree precision in orientation estimation. In 50 independent experiments involving a total of 400 screw insertions, the system achieved a 100 % success rate, confirming its reliability in practical conditions. These results confirm the feasibility of using RGB-only vision for precision screwdriving and highlight the mobile manipulation system's scalability for real-world semi-structured manufacturing environments.

1. Introduction

Screwdriving is a fundamental component of every assembly process in manufacturing [1,2]. Traditionally, these operations have relied on manual labor, which is prone to inconsistencies in torque application and alignment errors — factors that can compromise manufacturing quality. To address these issues and enhance precision, efficiency, and consistency, manual operations have increasingly been replaced by automated approaches. Despite the benefits of automation, conventional robotic solutions are typically confined to fixed cells with dedicated workstations, offering limited flexibility while still incurring high costs. However, with the advent of Industry 4.0, the robotics field is experiencing a paradigm shift through the emergence of systems that integrate mobility with manipulation capabilities [3]. These mobile manipulators represent a transition from traditional mechatronic systems to more autonomous and cognitively advanced solutions. Such solutions enable automation in manufacturing sectors that produce a diverse range of products in low volumes, where the cost of traditional automation would otherwise be prohibitively high [4,5]. However, automating low-volume production poses significant challenges, since solutions must remain adaptable and robust to unexpected changes [6]. By combining the mobility of mobile robots with the dexterity of robotic arms, they can perform complex manipulation tasks in diverse and dynamically changing work environments. Furthermore, their design prioritizes safe operation in human-centric settings [7], aligning with the broader goals of smart manufacturing and collaborative robotics. Especially in flexible, high-mix low-volume manufacturing environments, collaborative interaction between humans and robots can take multiple forms, including direct cooperation or collaboration, spatial coexistence, and sequential task sharing. These scenarios often require dynamic task allocation and physical workspace reuse, where the boundaries between human and robotic operations are less rigid than in traditional fixed automation. As such, ensuring safe and reliable operation becomes a fundamental requirement, even when direct physical collaboration is not present. Modern mobile manipulation systems must therefore integrate appropriate safety mechanisms tailored to the intended form of interaction, particularly when operating in dynamic or partially shared environments. While general robotic assembly strategies have been extensively surveyed [8], few works have addressed

* Corresponding author.
E-mail address: aleksandar.stefanov@fe.uni-lj.si (A. Stefanov).

https://doi.org/10.1016/j.rcim.2025.103148
Received 27 May 2025; Received in revised form 10 September 2025; Accepted 21 September 2025
Available online 30 September 2025

the specific challenges of mobile, vision-guided screwdriving in flexible production environments.

The integration of mobility to robotic arms introduces a set of challenges that extend beyond traditional fixed-base systems [9]. Such challenges can be especially prominent in tasks that require precise manipulation, where the manipulated object is being perceived and grasped given the task requirements [10–12]. For example, in mobile manipulation, precise alignment between the robot's end-effector and its environment is often compromised by inherent positioning errors of the mobile platform [13]. These errors can lead to deviations from a consistent spatial pose, meaning that even when the robot is intended to align with a workstation or part, its actual base may be misaligned relative to the fixed world coordinate system. This misalignment complicates the transformation from the robot's base frame to the end-effector's tool frame, resulting in errors that directly affect tasks requiring high precision, such as screwdriving or fine assembly. To mitigate these challenges, it is essential to integrate advanced perception and calibration modules that can compensate for these errors, ensuring that the end-effector maintains the accuracy necessary for reliable manufacturing operations. Such needs highlight how perception sensors represent a core component for precise mobile manipulation in industrial assembly task.

On that note, we propose a vision-guided mobile robotic manipulation system for performing screwdriving operations in assembly processes that require high degree of precision. We demonstrate and evaluate the practical applicability of mobile manipulation in real-world use cases, such as screwdriving, which remains a prevalent manual operation requiring high precision and is often distributed across multiple workstations — making it a natural candidate for automation through mobile robotic systems. The main contributions of this paper are as follows.

- We present a comprehensive pipeline for addressing screwdriving tasks using mobile manipulation, offering flexibility to accommodate variations in the pose of assembly objects to a certain extent.
- By integrating a perception algorithm for vision-guided control solely based on RGB data, we tackle the challenge of low pose repeatability in mobile robots, which inherently arises from localization limitations. While depth sensors have gained popularity in robotic perception tasks, our system deliberately adopts a monocular RGB-based approach due to its better suitability for high-precision screwdriving on metallic parts. RGB cameras offer greater flexibility in optical configuration and are not affected by depth sensing artifacts common on reflective surfaces. Additionally, they enable higher spatial resolution at lower cost, supporting the sub-millimeter and sub-degree accuracy required for our application.
- The proposed methodology is thoroughly evaluated in terms of accuracy and repeatability using standardized metrics. To the best of our knowledge, such a comprehensive validation has not been previously reported in the context of screwdriving applications.

2. Related work

The related work is categorized into two main segments. Section 2.1 reviews systems designed to address robotic screwdriving across diverse applications, while Section 2.2 examines vision-based methods employed to guide robotic screwdriving operations.

2.1. Robotic screwdriving approaches

The challenge of automated robotic screwdriving has long been a subject of research [14]. Researchers have approached the problem from diverse perspectives, proposing a range of different methodologies to address its complexities.

With advancements in robotic hands and their dexterous designs, several studies have been presented that explore the manipulation of hand-held screwdrivers for performing screwdriving operations [15–17]. The authors in [17] proposed a backward-chaining force control strategy for a robotic arm-hand system executing screwdriving. Their methodology involved sequential estimation of the screwdriver's pose and applied wrench using visual and kinematic feedback, then mapping these parameters through a grasp matrix to achieve precise closed-loop control. Another approach where manipulation of a manual screwdriver was proposed is [18], where a robotic arm mounts a rigidly attached screwdriver onto a pre-installed screw and it tightens it. The approach features a two-phase process: a sliding and rotating search of the screwdriver tip aided by impedance control to prevent slippage, followed by hybrid position/admittance control during screw turning. In contrast, automated assembly methods that rely on industrial screwdrivers have also been developed. The research in [19] presented a flexible assembly approach using a 6-DOF robotic arm and an intelligent automatic screwdriver, relying on calibration and image-based part detection for accurate positioning. Meanwhile, [20] focused on an unscrewing operation where the robot locates and aligns with screws along metal beams, enabling fully automated removal through a continuously adaptable learning model. Complementing these efforts, [21] integrate force sensing with a sensor-equipped screwdriver to achieve high success rates in automated screw removal from laptops, demonstrating that even under variable conditions, a coordinated sensor fusion strategy can deliver reliable performance.

Most of the previously mentioned approaches focus on controlled environments, where the uncertainty in the position of threaded holes is minimal due to fixed spatial relationships between the robot and the workstation. In such setups, both the robot and the parts to be manipulated remain static or follow predictable patterns. However, in more dynamic scenarios where screwdriving across multiple workstations is required, the use of mobile manipulators has gained increasing attention. Addressing this, authors in [22] proposed a position-based visual servoing architecture that synchronizes platform and arm motion to perform accurate screw-fastening on moving parts. In addition to control-focused approaches, [23] introduced a physics-informed learning framework that improves screwdriving robustness under hole pose uncertainties, emphasizing adaptability in less structured settings. In a more application-specific context, authors in [24] developed the Autonomous Prism Target Maintenance Robotic System (APTMRS), which leverages mobile manipulation for autonomous prism screwing and maintenance on large-scale scientific equipment, demonstrating the broader potential of mobile manipulators in screw-related tasks beyond traditional manufacturing.

While significant research has focused on specific screwdriving strategies, less attention has been given to comparing automation frameworks in terms of deployment cost and operational efficiency. Traditional automation frameworks, such as [15–21] are often built around stationary robotic cells, each operating on a dedicated workstation. While this setup can be highly efficient for large-batch production, it becomes cost-inefficient and rigid in low-volume manufacturing scenarios, where workstations may remain idle for long periods [25, 26]. In contrast, mobile manipulation frameworks, such as [22–24], enable a single robotic system to service multiple workstations, significantly improving equipment utilization and amortization of capital investment. Moreover, mobile manipulation frameworks can be rapidly reconfigured for new additional tasks, such as material transport, enhancing both flexibility and cost-effectiveness. These characteristics make mobile manipulation a particularly suitable approach for scalable automation in modern manufacturing settings with limited production volumes and diverse assembly requirements.

2.2. Vision guidance for screwdriving tasks

Recent advancements in robotic screwdriving increasingly leverage advanced computer vision algorithms to detect screw positions and estimate precise hole locations for successful insertion. These approaches generally fall into two categories: look-and-move, where the robot observes the scene, computes a target pose, and then moves without further visual feedback, and visual servoing, which uses continuous visual feedback to guide the robot in real time during the task.

For look-and-move methods, when the screwdriving operation is performed in planar setups, the problem of screw hole estimation becomes two-dimensional and is therefore easier to tackle. A few approaches have addressed this using standard computer vision techniques to detect and estimate hole positions. For instance, the authors in [21,27] employ vision-based strategies to localize fasteners for robotic manipulation. Each method utilizes edge detection and shape analysis – hexagon detection in one case and circular hole detection in the other – to extract precise target positions.

However, addressing the problem using look-and-move methods becomes more challenging in complex scenarios, where environments are dynamic and assembly objects do not maintain fixed poses. Such cases require the estimation of the full pose of screw holes or screws. To this end, most approaches leverage 3D point clouds in combination with registration techniques, such as Iterative Closest Point (ICP) [28]. In this context, the authors in [24] proposed a method that combines RGB and depth data to segment the target object and extract corner features. These features are mapped to 3D space for coarse registration using random sampling, followed by fine-tuning with ICP to achieve accurate and robust pose estimation. Similarly, in [29], the authors proposed a vision pipeline that estimates the 6D pose of bolts by combining deep learning-based segmentation of both real-world and pre-scanned point clouds with a coarse-to-fine registration process to align the full motor model to the observed scene. Bolt positions are extracted via Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering [30], which groups spatially dense clusters of points based on proximity and minimum neighbor criteria, while labeling sparse regions as noise. The orientations are inferred using normal alignment and prior knowledge of motor geometry to enable precise robotic disassembly. In contrast to point cloud-based approaches, the method in [31] uses a binocular vision system to estimate the 6D pose of threaded holes directly from image-space ellipses, leveraging a special chord of the projected circle to infer position and orientation without requiring 3D reconstruction or prior model fitting.

In the context of visual servoing for precise industrial automation, several notable approaches have been reported in the literature. In [22], the authors propose a position-based visual servoing architecture that coordinates both platform and arm motions to enable accurate screw fastening on dynamically moving parts. Complementing this, the work in [32] presents a visual servoing strategy for detecting and correcting misalignment during screw fastening with a 4-DOF robotic system. Similarly, the authors in [33] introduce an image-based visual servoing framework that incorporates screw detection and robotic assembly into a unified and structured workflow.

3. System overview

The robotic system employed in this research consists of several components designed to facilitate automated screwdriving and assembly tasks. Built on a mobile robotic platform, the system integrates various sensing and actuation elements to ensure precise operation. At the core of the mobile manipulator is the MiR 100, an autonomous mobile robot that serves as the primary base for the robotic manipulator. Mounted on this platform is the UR 5e, a six-degree-of-freedom (6-DOF) robotic arm used for precise manipulation tasks. The robot's end-effector, responsible for the screwdriving operation, consists of a Kolver PLUTO 6CA industrial screwdriver, paired with a Basler ace 2 A camera equipped with an industrial ring-shaped LED light (LRLW100, Wenglor GmbH) mounted around the lens to ensure uniform illumination during image acquisition. This vision system enables precise hole detection, facilitated by a custom computer vision pipeline, described in detail in Section 4.2. The mobile manipulator is equipped with an additional control computer featuring an Intel Core i7 processor, an NVIDIA RTX 3060 GPU, and 32 GB of RAM, ensuring sufficient computational power for real-time processing. The entire system operates within a dedicated network, to which the workstations involved in the manipulation process are connected.

At the workstation where the screwdriving operation takes place, a pneumatic gripper is used to grasp the assembly object, which is manually placed by the operator. As precise pose estimation of the object is critical, two additional Logitech C922 Pro cameras have been integrated into the workstation, supplemented with adjustable LED panel lights mounted near the cameras to maintain consistent and controlled illumination conditions. The synchronization and operation of the workstation are managed by an embedded processing unit, which is connected to the manipulator's network, ensuring seamless coordination between onboard and on-station components. To support this coordination, the system employs a layered communication architecture centered around a main router inside the mobile manipulator. The onboard processing computer and the UR 5e robotic arm are both connected via Ethernet to this router, enabling direct TCP/IP communication between them. The same router establishes a wireless TCP/IP connection to the embedded Raspberry Pi 5 controller on the workstation, which interfaces via USB with the on-station cameras and via Bluetooth with the LED lighting system. Additionally, the UR 5e and the MiR100 base exchange control signals over Ethernet through the MiR's internal router. The representation of the mobile manipulation system and the workstation is presented in Fig. 1.

To ensure real-world applicability, the system was designed with deployment efficiency and hardware modularity in mind. Once assembled, the pipeline supports rapid re-deployment thanks to pre-configured software modules. In practical terms, this means that, following hardware mounting and a short calibration process, the system can typically be re-deployed within a single working day. Hardware choices were informed by practical deployment needs, including mechanical compatibility, computational demands, and workspace accessibility. For instance, regular RGB cameras on the workstation and the industrial RGB camera with interchangeable optics were chosen to ensure consistent performance under variable lighting and reflective surfaces — where depth sensors underperform. Likewise, the use of an NVIDIA RTX 3060 was dictated by the need to support real-time image processing with minimal size constraints, making the system suitable for compact mobile platforms. The choice of the screwdriving tool was also influenced by geometric constraints, as the selected screwdriver needed to provide sufficient clearance to maneuver within the target assembly objects without colliding with nearby surfaces or obstructing the robot's motion.

4. Methodology

In this section, we present our overall algorithmic approach for vision-guided mobile robotic screwdriving. We first frame the problem and outline the pipeline's main stages — including navigation, object identification and segmentation, pose estimation, and screw-hole localization (Section 4.1). We then detail the perception algorithms for slope, height, and hole detection (Section 4.2), followed by the required coordinate transformations and calibration procedures to align camera and robot frames (Section 4.3).

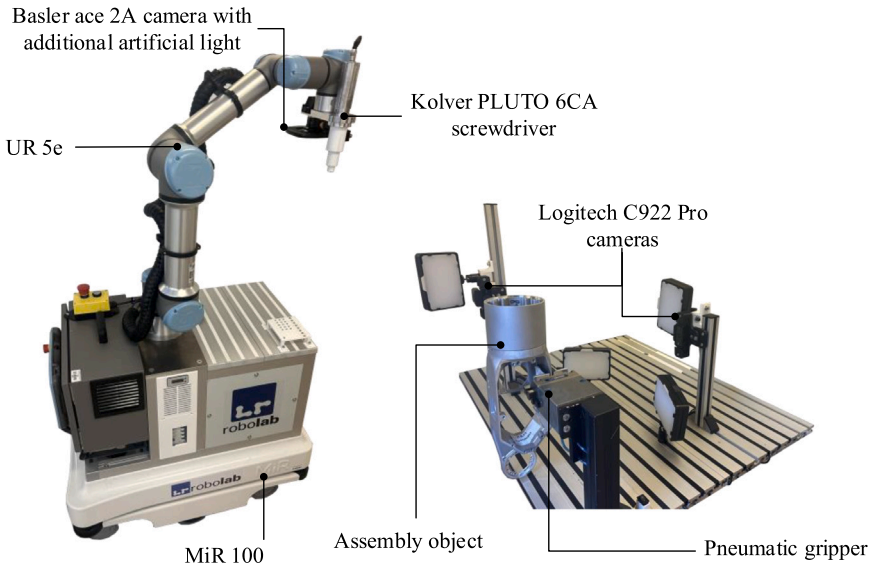

Fig. 1. Overview of the components of the proposed system used for addressing the screwdriving problem through mobile manipulation. On the left side of the figure is the mobile manipulation system, consisting of a MiR100 mobile platform upgraded with a collaborative UR 5e manipulator. The manipulator is equipped with a Kolver PLUTO 6CA industrial screwdriver, a Basler ace 2 A camera, and a Wenglor LRLW100 ring-shaped LED light. On the right side, the workstation components are presented, comprising two Logitech C922 Pro RGB cameras, LED panel lights, and a pneumatic gripper holding the assembly object.

4.1. Problem overview

To address the challenge of screwdriving across various workstations, we propose a comprehensive framework for vision-guided mobile manipulators. In the following section, we define the problem and outline the overall structure of our framework. An overview of this mobile manipulation pipeline is shown in Fig. 2. In the proposed deployment scenario, the interaction between the human operator and the mobile robot follows a sequential coexistence model, rather than physical human–robot collaboration. Specifically, the assembly object is manually placed into a workstation gripper by an operator prior to the robot's arrival. Once the object is positioned, the operator vacates the area, and the robot autonomously executes its task. Although no concurrent physical interaction occurs, robust safety measures are still necessary. For this reason, the MiR platform is equipped with 2D safety laser scanners, which are configured to immediately halt navigation or manipulation in case of unexpected human presence in the vicinity — whether during transit or operation.

The robot operates within a well-characterized environment that has been previously mapped as an occupancy grid $G \in \{0,1\}^{m \times n}$, where each cell indicates whether a location is occupied (1) or free (0). This grid is continuously updated using sensor data from newly observed scenes and serves as the primary basis for the robot's localization. The robot's pose at any time is defined relative to the map's global coordinate frame as $\mathbf{p}(t) = [x(t), y(t), \theta(t)]^\top$, where $x(t)$ and $y(t)$ represent the position coordinates and $\theta(t)$ represents the orientation. At the start of each operational cycle, the robot receives a goal pose $\mathbf{p}_{\text{goal}}$, corresponding to the workstation where the screwdriving task will be executed. Once the goal is set, the robot begins its autonomous navigation toward the designated target pose. As it approaches the vicinity of the workstation, the robot initiates a precise docking maneuver by leveraging an integrated physical marker mounted on the workstation.

Simultaneously, to improve cycle efficiency, two on-station cameras capture images to identify and segment the assembly object designated for screwdriving. Since the object is manually placed by an operator in the pneumatic gripper on the workstation, its pose is inherently uncertain and variable across cycles, meaning that even small estimation errors can compromise the precision required for the screwdriving process. To address this, the segmentation output S combined with the object identification data is used to refine the pose estimation. In particular, a tilt estimation algorithm is applied along two orthogonal axes (roll and pitch), yielding tilt angles α and β. Additionally, a height estimation is performed to accurately determine the object's vertical position. These measurements – tilt angles α and β and height h – are then communicated to the robot's control system, enabling precise alignment with the object's plane on which the screwdriving operation is being executed.

Once the refined pose data is received, the robot adjusts its initial pose to achieve precise alignment with the object's plane. With this alignment in place, the robot's end-effector camera acquires an image of the object. Image processing techniques are then applied to detect the screw holes intended for the screwdriving task. The metric locations of these holes are estimated by aligning the acquired image with the information from object's CAD model, which serves as a priori geometric information. Once the locations of the holes are provided to the robot, the robotic manipulator performs the screwdriving operation.

4.2. Perception algorithms

Our perception algorithms can be categorized into two main sections. The first focuses on utilizing images captured by stationary cameras to identify and estimate the precise pose of an object. The second involves the on-robot camera, which captures image of the object's plane where the robot performs the screwdriving operation. From this image, we detect and estimate the positions of the holes. In the following section, we will present the methodology behind these algorithms.

4.2.1. Object identification and pose estimation

The proposed algorithm estimates the slope (θ) and height (h) of an object from an image $I \in \mathbb{R}^{H \times W}$ using edge detection and least-squares fitting techniques. The method consists of five main steps: object detection, region-of-interest extraction, edge localization, slope estimation, and height computation, as presented in Fig. 3.

As outlined in Algorithm 1, we first apply a YOLOv8-based detector [34] to the input image I, which returns the identified object's class label, a bounding box B, and a binary mask M. The class label will later be utilized in the CAD-model registration step. Next, we use B to crop I and extract the region of interest I_{ROI}, effectively isolating the object for all subsequent analyses.

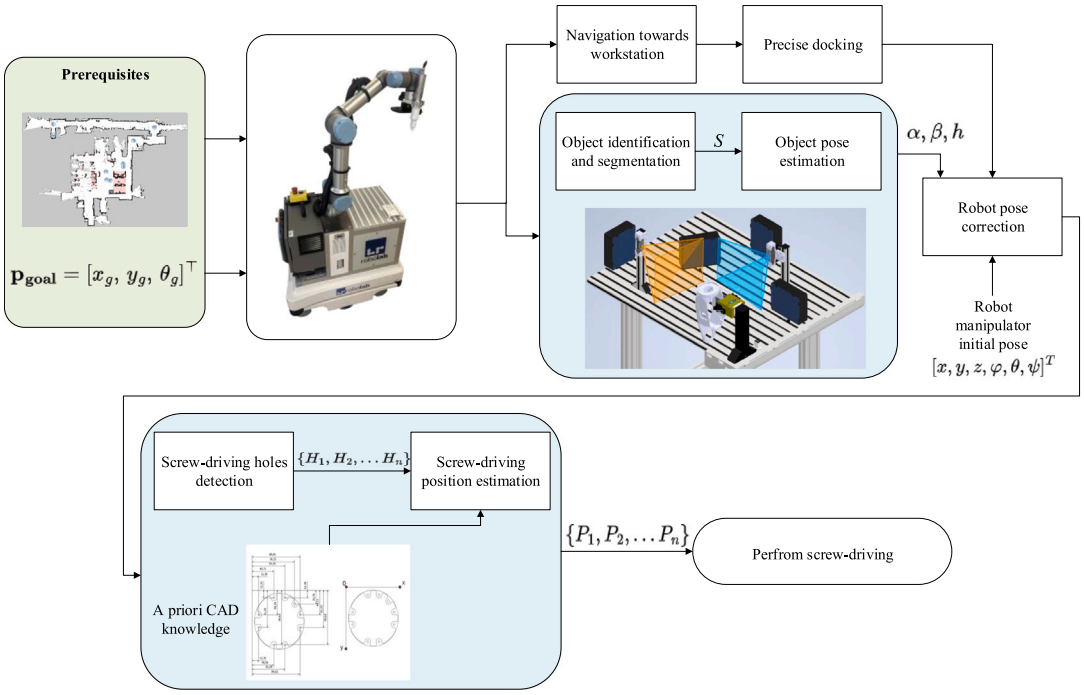

Fig. 2. Overview of the proposed mobile manipulation pipeline for robotic screwdriving tasks. The process begins with navigation of the mobile manipulator toward the workstation based on the goal position and grid map. Upon arrival, it performs precise docking. In parallel with navigation and docking procedures, object identification, segmentation, and pose estimation are performed using the on-station camera setup. The estimated object pose parameters are used for robot manipulator pose correction. The manipulator's on-arm camera then captures images of the assembly object for detecting screw holes. These detected hole positions are registered with the a priori CAD model of the object to obtain precise screwdriving locations. Finally, the robot executes screwdriving operations at the calculated positions.

Next, edge localization is performed to determine the object's boundaries. The algorithm processes scan lines within I_{ROI} (either horizontally or vertically) and evaluates the intensity function $I(p)$ along each scan line. The gradient $G(p)$ is computed to identify significant intensity changes, and the position of maximum gradient, denoted as p^*, is recorded as an edge coordinate (x_i, y_i). This process is repeated across multiple scan lines, producing a set of detected edge points $\{(x_i, y_i)\}_{i=1}^{N}$.

To estimate the slope of the object, a least-squares line fitting method is applied to the detected edge points (x_i, y_i). This fitting process determines the optimal slope (m^*) and intercept (c^*) of the best-fit line. The slope is converted into an angle (θ), which is further refined by applying a calibration offset δ_{cal} to obtain the final calibrated slope θ_{cal}.

For height estimation, the algorithm selects a vertical slice (S) near the center of the bounding box (B). For each column $x \in S$, the same gradient-based edge detection approach is used to determine the corresponding edge position (y_x^*). The average edge position, $\bar{y}$, is then computed and mapped to a physical height using predefined calibration parameters k_1 and k_2) and a reference coordinate (y_{ref}).

Finally, the algorithm outputs the estimated calibrated slope (θ_{cal}) and height (h), providing key geometric information about the detected object. Visualization of the algorithm is presented in Fig. 3. The process of obtaining the calibration offset is presented in Section 4.3.

To reduce the manual annotation burden during training of the segmentation model, we introduce a one-click label-propagation tool that enables rapid dataset generation. A detailed description of this tool is provided in Appendix.

4.2.2. Holes detection and position estimation

To address the detection and pose estimation of screwdriving holes, we first apply circle detection to the captured image, extracting 2D hole centers and radii. These detections are then registered to the

Algorithm 1 Slope and Height Estimation Algorithm

Input: Image $I \in \mathbb{R}^{H \times W}$, configuration parameters Θ (e.g., calibration offsets, edge direction)
Output: Estimated slope θ and height h
Step 1: Object Detection
  $L, B, M \leftarrow \text{Detector}(I)$ *(where L is the class label, B is the bounding box, M is a binary mask)*
Step 2: Region of Interest Extraction
  $I_{\text{ROI}} \leftarrow I[B]$
Step 3: Edge Localization via Gradient Analysis
for each scan line in I_{ROI} **do**
  Convert to grayscale if needed, then define intensity function $I(p)$
  Compute the gradient: $G(p) = |I(p+1) - I(p)|, \quad p \in$ scan line
  Find the max gradient position: $p^* = \arg\max_p G(p)$
  Record edge coordinates (x_i, y_i) at p^*
end for
  (Let $\{(x_i, y_i)\}_{i=1}^{N}$ be the collected edge points)
Step 4: Slope Estimation (Least-Squares Line Fitting)
Solve: $\min_{m,c} \sum_{i=1}^{N} (y_i - (mx_i + c))^2$
Compute $\theta = \arctan(m^*)$
Apply calibration: $\theta_{\text{cal}} = \theta + \delta_{\text{cal}}$
Step 5: Height Estimation
Define vertical slice S near the center of B
for each column $x \in S$ **do**
  Repeat gradient analysis to find y_x^*
end for
Compute average edge position: $\bar{y} = \frac{1}{|S|} \sum_{x \in S} y_x^*$
Convert to physical height: $h = k_1(y_{\text{ref}} - \bar{y}) + k_2$
return θ_{cal}, h

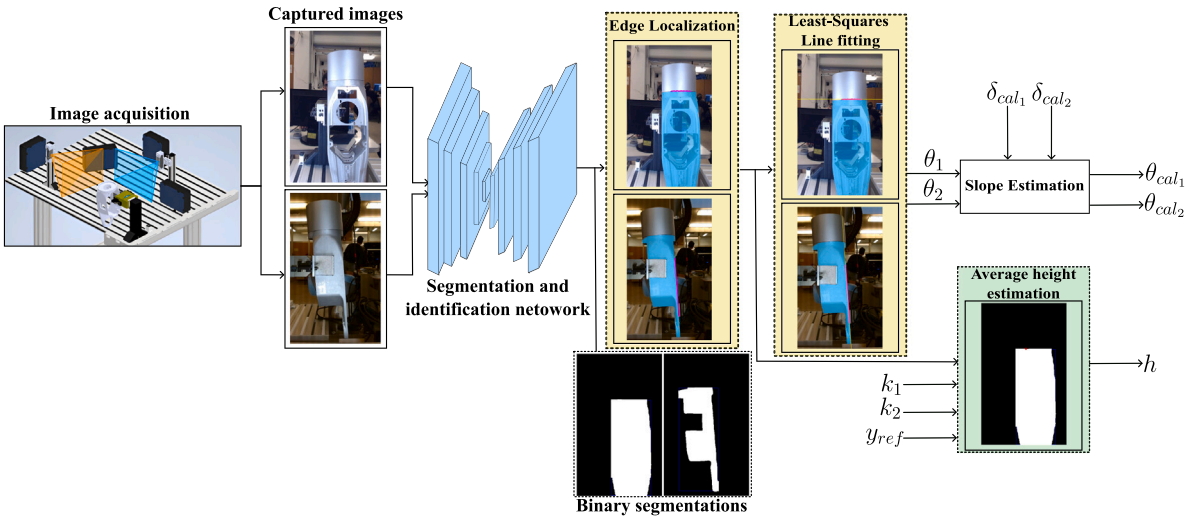

Fig. 3. Overview of the stationary-camera pose estimation pipeline. Starting from RGB images captured by the on-station cameras, a YOLOv8-based detector segments and identifies the object, then extracts a bounding box and mask. Within this region of interest, scan-line gradient analysis localizes edge points, which are then fitted with a least-squares line to compute the object's tilt angle θ. Finally, a vertical slice near the bounding-box center is processed via gradient detection to find the mean edge position, which is converted — using calibration parameters — into a height estimate h.

corresponding CAD model via a modified Iterative Closest Point (ICP) algorithm that integrates CAD-derived priors. The appropriate CAD reference is automatically selected based on the object's class label output by the YOLOv8 detector in the previous stage of the pipeline.

The input image is first converted to grayscale and binarized, and then a Gaussian blur is applied to reduce noise. The circular features (i.e., screwdriving holes) are detected using a Hough transform-based method [35]. The detected circles are represented by their center coordinates in the homogeneous form $\tilde{\mathbf{p}}_\mathbf{i} = [x_i, y_i, 1]^T$ and radii r_i. Corresponding to the object used in the ongoing cycle, a set of reference hole coordinates is extracted from a pre-stored CAD model, augmented to homogeneous coordinates, defined as $\tilde{\mathbf{q}}_\mathbf{i} = [x_i^{CAD}, y_i^{CAD}, 1]^T$.

The ICP alignment stage refines the initial correspondence between the detected hole centers and the CAD model points through iterative affine transformations. In the implementation, the moving (detected) points are first pre-transformed using a scaling factor and translation to approximately center them in the image domain. Specifically, a scaling and a translation corresponding to the image center are applied:

$$\tilde{\mathbf{p}}_i^{(0)} = \mathbf{T}_{\mathbf{init}} \cdot \tilde{\mathbf{p}}_i, \tag{1}$$

where $\mathbf{T}_{\mathbf{init}}$ denote the initial transformation used.

The iterative process then follows few steps. First, for each iteration, a bijective correspondence is established between the reference CAD points $\{\tilde{\mathbf{q}}_i\}$ and the current set of moving (detected) points $\{\tilde{\mathbf{p}}_i^{(k)}\}$ by minimizing the squared Euclidean distance,

$$j_i^{(k)} = \arg\min_j \left\| \tilde{\mathbf{q}}_i - \tilde{\mathbf{p}}_j^{(k)} \right\|^2, \tag{2}$$

where $j_i^{(k)}$ denotes the index of the moving (detected) point that is closest – in the least-squares sense – to the ith CAD reference point at iteration k.

Once a pair is matched, its distances are set to infinity to enforce a one-to-one mapping. Next, using the established correspondences, an affine transformation $\mathbf{T}^{(\mathbf{k})}$ is computed. This transformation comprises rotation, scaling, translation, and shear, and is estimated by computing the centroids

$$\mathbf{c}_q = \frac{1}{N}\sum_{i=1}^{N} \tilde{\mathbf{q}}_i, \quad \mathbf{c}_p^{(k)} = \frac{1}{N}\sum_{i=1}^{N} \tilde{\mathbf{p}}_i^{(k)},$$

estimating the rotation angle θ from the average angular difference between the vectors $\tilde{\mathbf{q}}_i - \mathbf{c}_q$ and $\tilde{\mathbf{p}}_i^{(k)} - \mathbf{c}_p^{(k)}$, determining a uniform scale factor s by comparing the mean distances from the centroids as:

$$\mathbf{s} = \frac{\frac{1}{N}\sum_{i=1}^{N} \left\| \tilde{\mathbf{p}}_i - \mathbf{c}_p \right\|}{\frac{1}{N}\sum_{i=1}^{N} \left\| \tilde{\mathbf{q}}_i - \mathbf{c}_q \right\|}, \tag{3}$$

and computing the translation as $\mathbf{c}_q - \mathbf{c}_p^{(k)}$. Thus, the estimated transformation matrix can be written as

$$\mathbf{T}^{(k)} = \mathbf{T}_{\text{trans}} \cdot \left(\mathbf{T}_{\text{back}} \cdot \mathbf{R}(\theta) \cdot \mathbf{T}_{\text{to-center}} \right) \cdot \mathbf{T}_{\text{scale}}, \tag{4}$$

where the individual matrices are constructed from the estimated parameters as follows:

- $\mathbf{T}_{\mathbf{scale}}$: the scaling matrix, which applies independent scaling factors along the x and y axes,

$$\mathbf{T}_{\text{scale}} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

  where s_x and s_y are estimated from the ratio of average distances of the moving and reference points to their respective centroids.
- $\mathbf{T}_{\mathbf{to-center}}$: the translation matrix that shifts the coordinate system so that the rotation is performed around the centroid (x_c, y_c),

$$\mathbf{T}_{\text{to-center}} = \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix}.$$

- $\mathbf{R}(\theta)$: the rotation matrix, which rotates the points by an angle θ (computed as the average angular difference between corresponding vectors),

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

- $\mathbf{T}_{\mathbf{back}}$: the inverse translation that repositions the points back to the original coordinate system after rotation,

$$\mathbf{T}_{\text{back}} = \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix}.$$

- $\mathbf{T_{trans}}$: the translation matrix that applies an overall translation (t_x, t_y) to the points,

$$\mathbf{T}_{\text{trans}} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

The moving points are then updated by applying the computed transformation:

$$\tilde{\mathbf{p}}_i^{(k+1)} = \mathbf{T^{(k)}} \cdot \tilde{\mathbf{p}}_i^{(k)},$$

and the alignment error is quantified as the root mean square (RMS) difference between the corresponding point pairs:

$$\mathrm{e}^{(k)} = \sqrt{\sum_{i=1}^{N} \left\| \tilde{\mathbf{q}}_i - \tilde{\mathbf{p}}_i^{(k)} \right\|^2}. \tag{5}$$

If the incremental change in the affine transformation (evaluated by comparing $\mathbf{T^{(k)}}$ with the identity transformation) is below a specified threshold ϵ yet the error remains high (e.g., $\mathrm{e}^{(k)} \geq 100$), a corrective rotation is applied around the current centroid using an angle of $\frac{2\pi}{N}$ to help overcome local minima. The iterative loop terminates when either the change in transformation parameters falls below ϵ or the maximum number of iterations is reached. All computed transformation matrices are compounded as

$$\mathbf{T_{final}} = \mathbf{T_{init}} \cdot \prod_{k=1}^{K} \mathbf{T^{(k)}}, \tag{6}$$

yielding the final affine transformation. Finally, a distance matrix is computed between the transformed reference points and the moving points. If any correspondence exhibits a sum of absolute differences that exceeds a threshold (based on the mean plus an offset), that point is considered an outlier and removed. The ICP process is then recursively re-applied on the reduced set of correspondences to further refine the transformation.

After convergence, the final transformation $\mathbf{T_{final}}$ is applied to the CAD points. The result is a set of transformed points in pixel coordinates:

$$\mathbf{q}_i^{\text{trans}} = \mathbf{T_{final}} \cdot \tilde{\mathbf{q}}_i. \tag{7}$$

A scale factor is computed by comparing a known inter-point distance in the CAD model with the corresponding distance in the transformed image domain. This factor converts the pixel measurements to real-world dimensions (in millimeters):

$$f = \frac{\mathrm{d}_{\text{CAD}}}{\mathrm{d}_{\text{pixels}}}, \tag{8}$$

so that the final pose in millimeters is given by

$$\mathbf{q}_i^{\text{mm}} = f\left(\mathbf{q}_i^{\text{trans}} - \mathbf{c}_{\text{img}}\right), \tag{9}$$

where $\mathbf{c}_{\text{img}}$ denotes the image center.

4.3. Transformations and calibration procedures

To formally describe the transformations between coordinate frames and the calibration procedures for specific system components, we provide Fig. 5, which defines the reference frames and the associated transformations. For the calibration process, two critical calibrations must be performed with high precision to ensure accurate system operation.

4.3.1. Calibration of the angular offset

The calibration procedure involves determining the angular offset required to transform angles from the camera coordinate frames to the workstation coordinate frame. When the side cameras capture an image and compute the angle using the methods described in Section 4.2.1, the resulting angle is expressed in the local frame of either CS_{C1} or CS_{C2}, depending on the camera. However, to be meaningful for the robot manipulator, this angle must be represented in the workstation coordinate frame CS_{WS}, which has been pre-taught to the robot.

To estimate the angular offset, we capture N images of the object, each with a distinct pose. For these N instances, we also measure the corresponding angles relative to CS_{WS} using a high-precision digital inclinometer with angular resolution of 0.1°. Let $\alpha_{C_j,i}$ be the angle measured in the jth camera's coordinate frame ($j \in \{1, 2\}$) and $\alpha_{WS,i}$ the angle measured in CS_{WS} for the ith image. The calibration offset for camera j, denoted δ_j, is then calculated as the average difference across all N measurements:

$$\delta_j = \frac{1}{N} \sum_{i=1}^{N} \left(\alpha_{WS,i} - \alpha_{C_j,i}\right) \quad \text{for} \quad j \in \{1, 2\}. \tag{10}$$

This procedure yields two distinct calibration offsets, one for each camera, ensuring that angles derived from either CS_{C1} or CS_{C2} can be accurately transformed into the workstation's coordinate frame CS_{WS}.

4.3.2. Calibration of the transformation between the camera and the screwdriver

Accurate knowledge of the transformation between the camera and the screwdriver is essential for precise operation. While initial estimates were obtained from dimensions in the CAD model of the tool interface, this method is inherently prone to error. In tasks such as screwdriving, even minor inaccuracies in the transformation can significantly degrade system performance. To address this issue and enhance precision, we propose a calibration method based on the optimization of static transformations.

We define ${}^{F}\mathbf{H}_{C}, {}^{F}\mathbf{H}_{TCP} \in SE(3)$ as shown in Fig. 5, representing the static transformations from the robot's flange frame CS_F to the camera frame CS_C and the tool center point CS_{TCP}, respectively. Since both the camera and the screwdriver are rigidly attached to the end-effector, these transformations remain constant during operation.

The transformation from the camera frame to the screwdriver frame, denoted ${}^{C}\mathbf{H}_{TCP}$, can be computed as:

$${}^{C}\mathbf{H}_{TCP} = {}^{F}\mathbf{H}_{C}^{-1} \cdot {}^{F}\mathbf{H}_{TCP}. \tag{11}$$

To calibrate this transformation, a set of target points is selected within the robot's workspace such that they are visible to the camera and reachable by the screwdriver. For each calibration instance i, two data points are recorded: $\mathbf{p}_c^{(i)} \in \mathbb{R}^3$, the point expressed in the camera frame, and $\mathbf{p}_{TCP}^{(i)} \in \mathbb{R}^3$, the corresponding point in the screwdriver's frame, obtained through touch-based positioning.

The objective is to optimize the transformations ${}^{F}\mathbf{H}_{C}$ and ${}^{F}\mathbf{H}_{TCP}$ such that the points observed in the camera frame, when transformed into the screwdriver's frame, align as closely as possible with their corresponding measured positions. To achieve this, we introduce a scalar cost J and write the estimation in a single optimization:

$$\begin{aligned} \left(\widehat{{}^{F}\mathbf{H}_{C}}, \widehat{{}^{F}\mathbf{H}_{TCP}}\right) &= \arg\min_{{}^{F}\mathbf{H}_{C}, {}^{F}\mathbf{H}_{TCP}} J\left({}^{F}\mathbf{H}_{C}, {}^{F}\mathbf{H}_{TCP}\right), \\ J\left({}^{F}\mathbf{H}_{C}, {}^{F}\mathbf{H}_{TCP}\right) &= \sum_{i=1}^{n} \left\| {}^{F}\mathbf{H}_{TCP}^{-1} \cdot {}^{F}\mathbf{H}_{C} \cdot \begin{bmatrix} \mathbf{p}_c^{(i)} \\ 1 \end{bmatrix} - \begin{bmatrix} \mathbf{p}_s^{(i)} \\ 1 \end{bmatrix} \right\|^2. \end{aligned} \tag{12}$$

This optimization minimizes the discrepancy between the predicted screwdriver positions and the ground-truth measurements. The transformation parameters are estimated using a nonlinear least-squares solver based on the Levenberg–Marquardt method.

5. Experiments

To evaluate the performance of the proposed pipeline, we categorized the experiments into several groups and carried them out on two test objects — Assembly Object 1 and Assembly Object 2 — both depicted in Fig. 6. While the number of evaluated objects is limited, Assembly Objects 1 and 2 were deliberately selected to reflect different geometric features and practical screwdriving constraints commonly

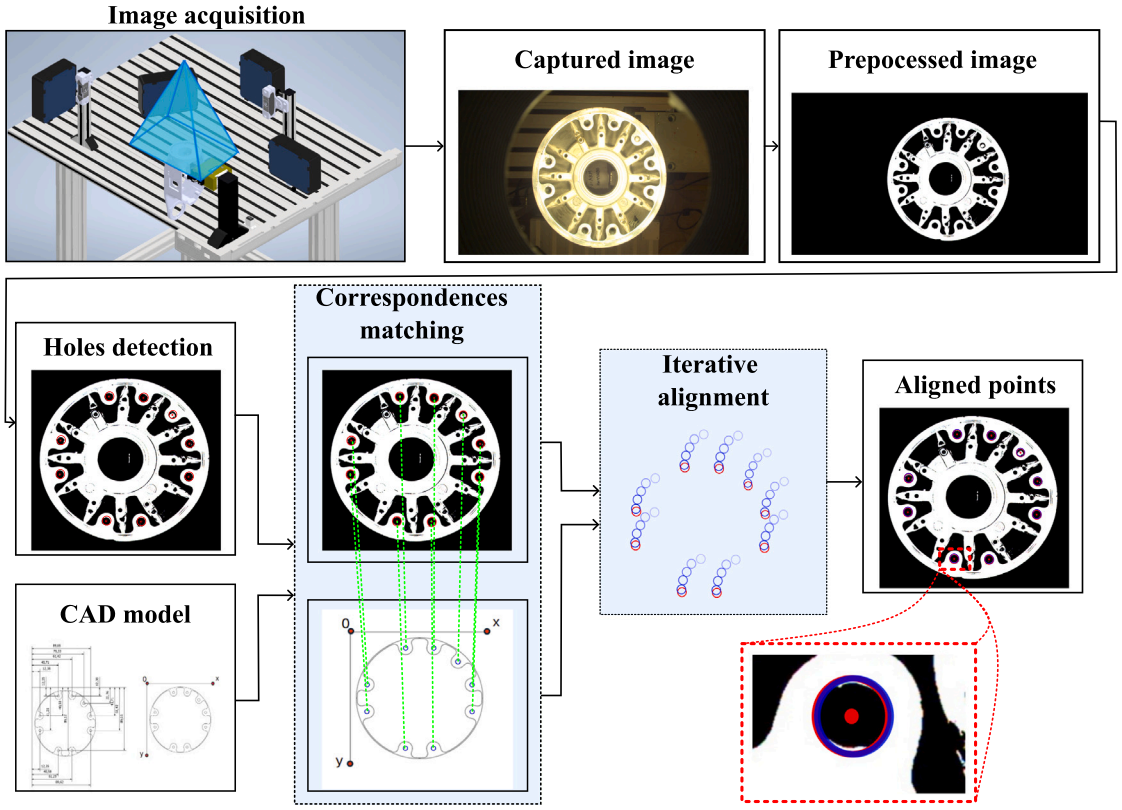

Fig. 4. Overview of the pipeline for screw-hole detection and CAD-based position estimation. The end-effector camera image is first preprocessed (grayscale, binarization, Gaussian blur) and then analyzed with a Hough-transform circle detector to yield 2D hole centers and radii. A corresponding set of reference hole coordinates is extracted from the part's CAD model. These detected and reference points undergo an iterative affine alignment – establishing one-to-one correspondences and solving for rotation, scaling, and translation at each iteration – until convergence. The final composite transformation maps CAD hole locations into image space, enabling conversion to real-world coordinates via a computed scale factor.

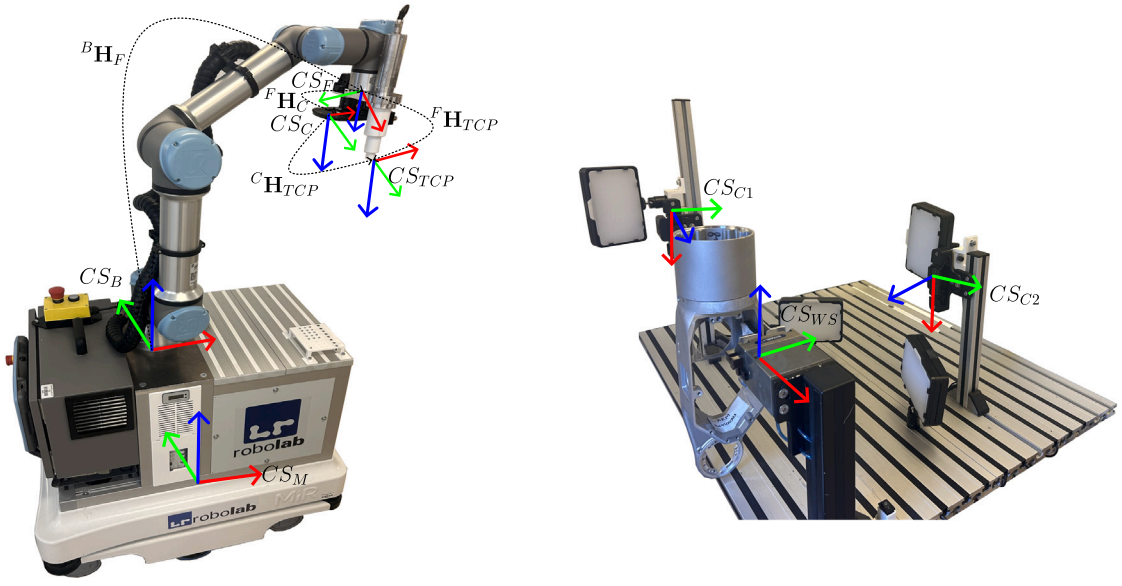

Fig. 5. Definition of all relevant coordinate systems (CS) and spatial transformations (**H**) used for system calibration and task execution. The coordinate systems are defined as follows: CS_B denotes the base frame of the robot manipulator, CS_M the base frame of the mobile platform, CS_F the robot's flange frame, CS_C the camera mounted on the robot, CS_{TCP} the tool center point (TCP) of the screwdriving tool, CS_{WS} the local frame of the workstation, and CS_{C1} and CS_{C2} the frames of the cameras mounted on the workstation. The associated transformations are defined as follows: ${}^{F}\mathbf{H}_{C}$ represents the transformation from the robot's flange to the camera frame, ${}^{F}\mathbf{H}_{TCP}$ the transformation from the flange to the tool's TCP, and ${}^{B}\mathbf{H}_{F}$ the transformation from the robot's base to its flange. Additional transformations are defined analogously where applicable.

encountered in high-mix, low-volume production. Both objects require screwdriving operations but differ significantly in terms of physical layout and hole placement. These variations allowed us to validate the generality of the proposed perception and calibration pipeline across distinct object geometries. As such, this evaluation serves as a representative principle toward broader applicability in diverse high-mix scenarios.

The first set of experiments focuses on assessing the segmentation capabilities of the perception algorithm. Since segmentation is the initial step in the perception pipeline, its accuracy is critical for the overall success of the system. Therefore, we evaluate the segmentation performance using the Intersection over Union (IoU) metric.

The second and third groups of experiments involve the use of the OptoTrak Certus motion capture system. The second group aims to assess the accuracy of angle estimation algorithm described in Section 4.2.1. The third group evaluates the overall performance of the complete pipeline by measuring the overall accuracy of the screwdriving task. Since this evaluation targets the end-to-end execution accuracy of the entire system, it inherently reflects the combined influence of various upstream factors (such as perception errors, calibration offsets, and uncertainties in the robot's kinematics) without performing a separate analysis of how each factor individually contributes to the overall error. Specifically, we assess how accurately and repeatably the screwdriver reaches the target locations (the holes to be screwed). The evaluation metrics used for this purpose are described in detail in Section 5.3. In addition to these quantitative accuracy and repeatability metrics, we conducted consecutive screwdriving experiments with randomized object poses to record the task success rate.

The remainder of this section is structured as follows: the experimental setup is presented in Section 5.1, followed by the translational evaluation metrics in Section 5.3, the orientation evaluation metric in Section 5.2, and finally, the evaluation protocol is detailed in Section 5.4.

5.1. Measurement setup and procedure

For the second and third sets of experiments, we employed the OptoTrak Certus motion capture system. In our system, we used two sets of linear CCD cameras positioned orthogonally to each other to track the 3D positions of infrared markers with submillimeter precision (±0.15mm 1D error at a distance of 2.25 m). However, it needs to be noted that the OptoTrak system exhibits reduced accuracy along its optical depth axis (denoted z_{opt}), particularly when the tracked markers are tilted relative to the camera plane. This characteristic must be considered when interpreting measurements in configurations where markers are viewed at non-perpendicular angles, as it can introduce anisotropic error distributions in the recorded 3D positions. In our case, the markers were attached to the robot, the workstation, and the assembly objects.

The second set of experiments focused on evaluating the accuracy of the angle estimation algorithm and was conducted using two different assembly objects equipped with markers, as shown in Fig. 6. Since the angles are estimated relative to the workstation's coordinate system (CS_{WS}, see Fig. 5), it was necessary to define this reference frame for evaluation. To achieve this, we mounted four infrared markers (M_1, M_2, M_3, M_4) on the pneumatic gripper. These markers enabled us to define coordinate frame CS_{WS}, thereby providing a reference for validating the estimated angles.

For the third set of experiments, which aimed to evaluate the overall accuracy and repeatability of the complete screwdriving pipeline, only Assembly Object 1 was utilized. Assembly Object 2 was excluded due to physical constraints, as the screwdriving surface was located within a recessed area, leaving less than 1 mm of clearance for the OptoTrak Certus touch probe. This made it nearly impossible to maneuver the probe without risking collisions with the object's vertical walls, thus preventing accurate and repeatable measurements. To assess the accuracy of the screwdriving process, the industrial screwdriver was replaced with an OptoTrak Certus touch probe. This probe estimates the position of the contact point using four integrated infrared markers (M_{15}, M_{16}, M_{17}, M_{18}), as shown in Fig. 6. Based on the four markers, calibration was performed in order to constantly and accurately track the pose of the contact point.

5.2. Orientation evaluation metrics

To evaluate the orientation accuracy of the angle estimation pipeline, we compute the angular deviation of planes defined by groups of markers affixed to both the object and the workstation. The evaluation procedure was carried out using both objects illustrated in Fig. 6. Orientation evaluation is based on estimating the normal vectors of these marker-defined planes and comparing them against a reference coordinate system derived from the known marker layout.

The setup involved markers M_1 to M_{14}, as shown in Fig. 6. Since data was recorded over an entire cycle and the affixed markers remained static during the experiments, the mean position $\overline{M_i}$ of each marker was computed to mitigate the impact of noise and ensure robust positional estimates:

$$\overline{M_i} = \left(\frac{1}{N}\sum_{k=1}^{N} x_i^{(k)}, \frac{1}{N}\sum_{k=1}^{N} y_i^{(k)}, \frac{1}{N}\sum_{k=1}^{N} z_i^{(k)} \right), \tag{13}$$

where N is the total number of recorded frames.

These averaged marker positions were then transformed into a local coordinate frame defined by markers M_1, M_2, and M_3, using the procedure described in Section 5.3. This results in a local reference frame with axes $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$.

The following explanation focuses on object 1, with object 2 being processed analogously using a different set of markers. Two planes are defined using triplets of markers: the first plane is formed by M_6, M_7, and M_8, and the second by M_5, M_9, and M_{10}. Although the definition of a single plane suffices for evaluation, two planes are used as a redundancy to account for possible noise or occlusion affecting some markers.

For each plane, we calculate two vectors from a central marker, and then compute the cross product to obtain the normal vector:

$$\mathbf{v}_{7\to8} = \overline{M}_8 - \overline{M}_7, \quad \mathbf{v}_{7\to6} = \overline{M}_6 - \overline{M}_7, \quad \mathbf{n}_{678} = \mathbf{v}_{7\to8} \times \mathbf{v}_{7\to6}. \tag{14}$$

An analogous procedure is applied to the second plane to compute $\mathbf{n}_{5910}$. To evaluate the orientation of these planes with respect to the local $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ axes, we project the normal vectors onto vectors encoding the reference directions. The projections are computed as:

$$\begin{aligned} proj_x &= \mathbf{n}_{678} \cdot \mathbf{d}_{\mathbf{x}}, \\ proj_y &= \mathbf{n}_{678} \cdot \mathbf{d}_{\mathbf{y}}, \end{aligned} \tag{15}$$

where $\mathbf{d}_{\mathbf{x}} = [1, 0, \hat{x}_z]$ and $\mathbf{d}_{\mathbf{y}} = [0, 1, \hat{y}_z]$ represent the local reference directions with respect to the x and y axes.

From these projections, the angles between the normal vector and the reference axes are calculated as:

$$\begin{aligned} \theta_{ref,x} &= \cos^{-1}\left(\frac{proj_x}{\|\mathbf{n}_{678}\|\|[1,0,\hat{x}_z]\|} \right), \\ \theta_{ref,y} &= \cos^{-1}\left(\frac{proj_y}{\|\mathbf{n}_{678}\|\|[0,1,\hat{y}_z]\|} \right). \end{aligned} \tag{16}$$

The resulting angles $\theta_{ref,x}$ and $\theta_{ref,y}$ represent the ground truth orientation with respect to the x and y axes, respectively. These are compared to the angles $\theta_{cal,x}$ and $\theta_{cal,y}$ obtained from the perception algorithm described in Section 4.2.1. The absolute angular errors are computed as:

$$\begin{aligned} \Delta\theta_x &= |\theta_{cal,x} - \theta_{ref,x}|, \\ \Delta\theta_y &= |\theta_{cal,y} - \theta_{ref,y}|. \end{aligned} \tag{17}$$

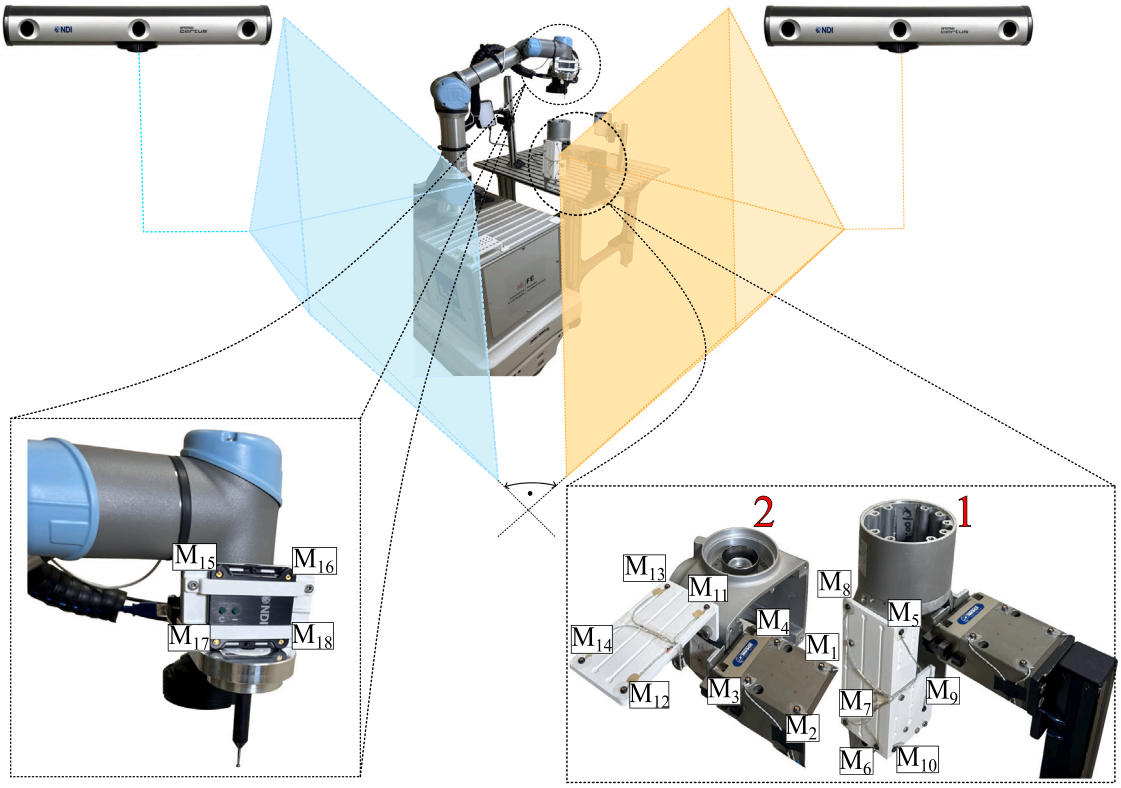

Fig. 6. Measurement setup using two OptoTrak Certus acquisition systems positioned orthogonally to capture the 3D positions of infrared markers with high precision. Marker placement was as follows: M_1–M_4 on the pneumatic gripper mounted on the workstation, M_5–M_{10} on Assembly Object 1, M_{11}–M_{14} on Assembly Object 2, and M_{15}–M_{18} on the OptoTrak Certus touch probe.

5.3. Translation evaluation metrics

To evaluate the overall success of the screwdriving pipeline we assess the accuracy and repeatability of the end-effector position between the reference position and the attained position of the end-effector in each experiment. The performance metrics were assessed according to ISO 9283 standard (Manipulating industrial robots — Performance criteria and related test methods) [36].

The position accuracy is defined as the deviation between the reference position and the mean value of the attained positions when the end-effector touches the screwdriving hole. The position accuracy is determined by the distance between the reference position and the barycenter of the cluster of attained positions. The position accuracy is then expressed by the following equation

$$\Delta L = \sqrt{(\overline{x} - x_c)^2 + (\overline{y} - y_c)^2 + (\overline{z} - z_c)^2}, \tag{18}$$

where $(\overline{x}, \overline{y}, \overline{z})$ are the coordinates of the cluster barycenter, obtained by averaging the N measurement points, assessed when repeating the motions into the same reference position O_c with coordinates (x_c, y_c, z_c).

For the same series of measurements, we calculate the position repeatability. The position repeatability expresses the closeness of the positions of the N attained positions when repeating the robots motion into the same reference position. The position repeatability is determined by the radius of the sphere r whose center is the cluster barycenter. The radius is defined as

$$r = \overline{D} + 3S_D, \tag{19}$$

where, $\overline{D}$ and S_D are defined as

$$\begin{aligned} \overline{D} &= \frac{1}{N}\sum_{j=1}^{N} D_j \\ D_j &= \sqrt{(x_j - \overline{x})^2 + (y_j - \overline{y})^2 + (z_j - \overline{z})^2} \\ S_D &= \sqrt{\frac{\sum_{j=1}^{N}(D_j - \overline{D})^2}{n-1}}. \end{aligned} \tag{20}$$

In the OptoTrak Certus system, each pose of the touch probe is expressed in a global (world) coordinate frame defined within the system's field of view. However, because the assembly object changes its position and orientation in each experiment, the probe's global pose will not match the original global reference pose. To evaluate the metrics – which require the robot's end-effector to consistently reach the same reference point in each experiment – it is necessary to transform both the newly acquired pose and the reference pose into a local coordinate frame that remains fixed to the moving assembly object. This transformation is essential, as it is the only way to ensure meaningful comparison and reliable assessment of the application's accuracy and repeatability.

In order to establish a local coordinate frame for evaluating the screwdriving precision, three markers affixed to the assembly object (markers M_6, M_7, M_8) were utilized. Each marker is represented by its global coordinate vector $\vec{M}_i = [x_i, y_i, z_i]^T$ for i = 6,7,8. The local $\hat{\mathbf{x}}$-axis is defined as

$$\hat{\mathbf{x}} = \frac{\vec{M}_6 - \vec{M}_7}{\|\vec{M}_6 - \vec{M}_7\|}. \tag{21}$$

With utilization of the obtained $\hat{\mathbf{x}}$ axis, we consequently define the local $\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ axis as follows

$$\hat{\mathbf{y}} = \frac{(\vec{M}_8 - \vec{M}_7) \times \hat{\mathbf{x}}}{\|(\vec{M}_8 - \vec{M}_7) \times \hat{\mathbf{x}}\|}, \hat{\mathbf{z}} = \hat{\mathbf{x}} \times \hat{\mathbf{y}}. \tag{22}$$

For the origin of the coordinate frame, we choose $\vec{t} = \vec{M}_7$. Thus, the homogeneous transformation matrix that converts coordinates from the local to the global frame is

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \vec{t} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} \hat{\mathbf{x}} & \hat{\mathbf{y}} & \hat{\mathbf{z}} \end{bmatrix}. \tag{23}$$

The established matrix $\mathbf{T}$ is then utilized for transforming the global points into the local coordinate system as

$$\mathbf{p}_{\text{local}} = \mathbf{T}^{-1}\mathbf{p}_{\text{global}}. \tag{24}$$

5.4. Evaluation protocol

The evaluation is structured into three sets of experiments, each targeting a specific aspect of the proposed pipeline.

The first set focuses on assessing the segmentation performance of the perception algorithm by computing the Intersection over Union (IoU) metric. A total of 60 experiments were conducted – 30 per object – where the segmentation masks generated by the algorithm were compared against manually annotated ground truth masks.

The second set of experiments evaluates the accuracy of the angle estimation algorithm, based on the orientation metrics described in Section 5.2. A total of 40 experiments were performed, with 15 randomized object poses evaluated for object number 1 and 25 for object number 2.

The third and final set of experiments assesses the overall performance of the complete pipeline by measuring the accuracy and repeatability of the screwdriving task, as defined in Section 5.3. This evaluation comprises 20 consecutive experiments. In each experiment, the entire pipeline is executed as outlined in Section 4.1, with the object placed in a randomized pose for every cycle. During the experiments, the payload was fixed at 0.850 kg, and the end-effector's point-to-point velocity between holes was maintained at 50 mm/s. This setup simulates realistic operator behavior in practical applications.

6. Results

In this section, we present the outcomes of the evaluation procedures described in Section 5. The results follow the structure of the experimental protocol and address the key components of the proposed screwdriving pipeline. Each subsection highlights the performance of individual system modules — ranging from perception to execution — and demonstrates the accuracy and consistency of the overall system in real-world scenarios.

6.1. Segmentation performance

The segmentation performance is summarized in Table 1, which reports the average Intersection over Union (IoU) along with the standard deviation for each camera–object pair. To complement the quantitative results, we also provide a figure (Fig. 7) illustrating representative segmentation examples for each pair. Each example includes the ground truth, the predicted segmentation, and the resulting difference. The selected samples are those whose IoU scores are closest to the mean values reported in Table 1, thereby offering a visual representation that best reflects the typical performance. The segmentation was performed using the same trained model for all camera–object combinations, without any additional tuning or adaptation.

Table 1
Average Intersection over Union (IoU) and standard deviation (SD) for each object–camera pair.

Object	Camera 1 (IoU ± SD)	Camera 2 (IoU ± SD)
1	0.973 ± 0.004	0.944 ± 0.008
2	0.969 ± 0.007	0.985 ± 0.002

Table 2
Average orientation error along with the corresponding standard deviation for rotations around the x and y axes ($\overline{\Delta\theta_x}$, $\overline{\Delta\theta_y}$), reported separately for each of the two tested objects.

Object	$\overline{\Delta\theta_x}$/°	$\overline{\Delta\theta_y}$/°
1	0.07 ± 0.06	0.13 ± 0.07
2	0.09 ± 0.07	0.20 ± 0.12

6.2. Orientation estimation accuracy

The results of the experimental pipeline for evaluating orientation estimation accuracy, as described in Section 5.2, are summarized in Table 2. The table presents the average orientation error along with the corresponding standard deviation for rotations around the x- and y-axes, reported separately for each of the two tested objects.

6.3. End-effector accuracy and repeatability

The results of the experiments described in Section 5.3, which evaluate the accuracy and repeatability of the end-effector's translational positioning, are presented in Table 3. Since each hole defines a distinct local reference pose used for metric estimation according to the standard, the results for both accuracy and repeatability are reported separately for each hole. To provide a clearer representation of the repeatability metric, an additional figure (Fig. 8) is included, illustrating the deviations in the x and y directions from the estimated barycenter. Only x and y deviations are shown, as they are more relevant to the task and exhibit greater variability, making them more significant to the overall analysis.

6.4. Overall task success rate

To evaluate the reliability of our complete screw-driving pipeline, we performed 50 independent experiments, each requiring insertion of screws into 8 target holes (totaling 400 insertion attempts). An experiment was deemed successful if the screw was fully inserted into the threaded hole without misalignment or any manual intervention. Across all 400 attempts, the system achieved a 100% success rate.

7. Discussion

The adoption of a mobile manipulator, rather than fully robotizing each workstation, introduces unique challenges such as localization uncertainty and dynamic workspace configurations. Our experimental results directly demonstrate that these challenges can be effectively mitigated through perception-driven compensation strategies. Specifically, the integration of stationary and onboard cameras enables sufficient pose correction to perform precision screwdriving, as reflected by the sub-millimeter accuracy and 100% success rate reported in Section 6.

Our second contribution centers on the design of a perception stack based entirely on monocular RGB cameras, in contrast to most mobile manipulation pipelines that rely on RGB-D or stereo vision systems. The approach supports operation using only a single RGB camera mounted on the robot, which sequentially acquires images from different viewpoints as needed. This configuration enables substantial cost savings and reduces system complexity when compared

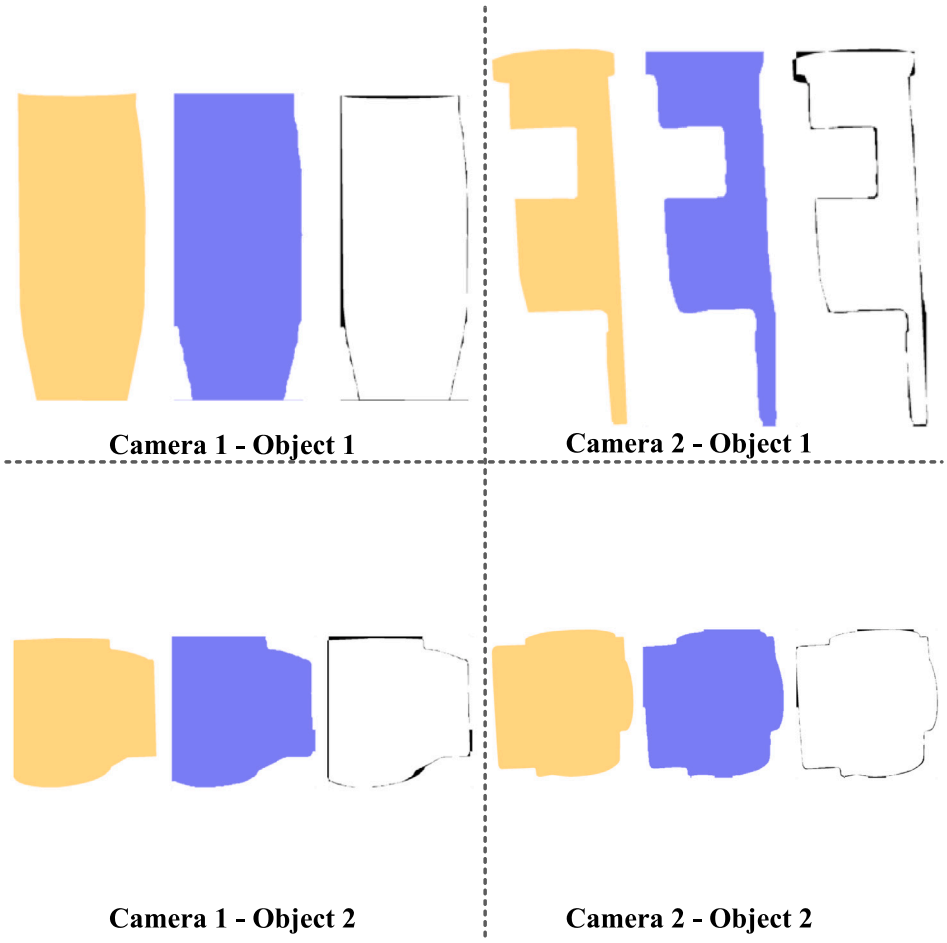

Fig. 7. Representative segmentation examples for each pair. Each example includes the ground truth mask (yellow), the predicted segmentation mask (purple), and the resulting difference (black).

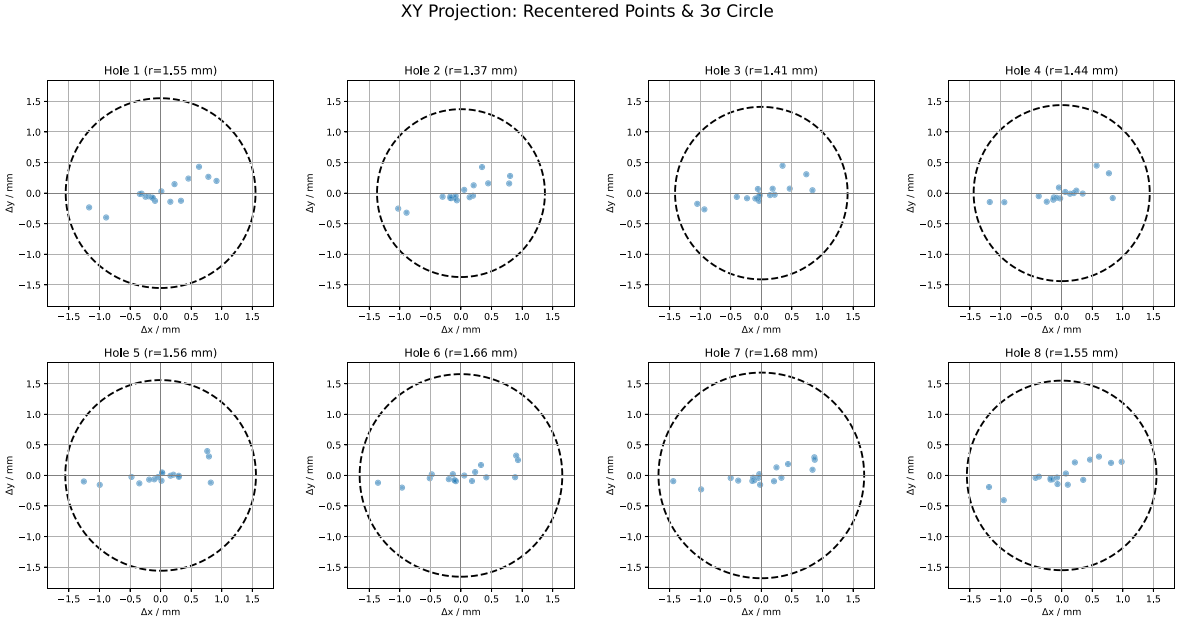

Fig. 8. Repeatability of end-effector translational positioning in the xy-plane for Holes 1–8. Each sub-figure shows individual deviations Δx and Δy from the estimated barycenter, and the dashed circle in each panel denotes the projection of the sphere with radius r (in mm) associated with that hole's local reference pose repeatability.

Table 3
Position accuracy (ΔL) and repeatability (r) for the eight screw-driving holes.

	Hole 1	Hole 2	Hole 3	Hole 4	Hole 5	Hole 6	Hole 7	Hole 8
Accuracy ΔL/mm	0.47	0.39	0.36	0.50	0.21	0.34	0.24	0.41
Repeatability r/mm	1.55	1.37	1.41	1.44	1.56	1.66	1.68	1.55

to setups that rely on industrial depth cameras, which are not only more expensive but also less reliable on reflective materials such as metal — our primary object type. In our implementation, we employ three RGB cameras to improve the system's execution time. Specifically, two additional stationary cameras are used to capture and preprocess visual information while the robot navigates toward the workstation. This parallelization ensures that perception data is already available by the time the robot arrives, thereby optimizing cycle time. The system remains fully compatible with a single-robot-mounted camera configuration, offering flexibility to balance between cost-efficiency and operational speed based on application requirements. While RGB-D and stereo vision systems are widely used in robotic perception, our choice of a monocular RGB-only pipeline was guided by practical requirements for precision, adaptability, and robustness. In our testing, cost-efficient depth sensors frequently produced incomplete or noisy depth maps when imaging reflective metallic surfaces, leading to unreliable geometry representation. Our observations are consistent with previous studies, which have shown that reflective materials such as metal interfere with depth sensing methods, causing specular reflections, invalid depth values, and surface voids [37]. These artifacts can significantly affect downstream processes such as 3D point cloud registration, where voxel downsampling can amplify inconsistencies and further degrade alignment accuracy. Similar issues were reported by [38], where metal objects often yielded point clouds with outliers and missing regions, directly impairing registration performance. In addition to structural artifacts, typical off-the-shelf depth sensors used in robotic applications provide limited depth accuracy, which poses a significant challenge for tasks requiring sub-millimeter and sub-degree precision, such as screwdriving. The accumulated noise and error in the depth channel can propagate through the perception pipeline and ultimately compromise pose estimation quality. Moreover, RGB-D cameras typically have fixed optics and lower spatial resolution compared to industrial RGB sensors with interchangeable lenses. This limits their adaptability for tasks involving fine object details, such as detecting small screw holes or adapting to varying object scales. These considerations make monocular RGB a more appropriate choice for the precision and flexibility demands of our application.

The effectiveness of the RGB-only setup is assessed through the performance of the key perception modules, beginning with object segmentation and pose estimation. In our approach, object orientation is estimated along two axes (roll and pitch) using a gradient-based scanline method on grayscale images, followed by least-squares line fitting. Experimental results demonstrate that this technique yields sub-degree estimation accuracy, with errors ranging from 0.07° to 0.20°, as reported in Table 2. Table 1 shows that our segmentation model performs strongly in terms of IoU, suggesting that the masks are highly consistent with ground truth. Given the high IoU scores, one might consider using the segmentation output directly for edge localization and orientation estimation. While the IoU scores are indeed high – and sufficient for general object detection tasks – a closer inspection (Fig. 7) reveals that the predicted masks tend to be imprecise along object boundaries. This is especially true in regions with low contrast or variable background, where slight deviations in mask contours are common. Because our application demands sub-degree angular precision, such edge inconsistencies become problematic. In earlier iterations of the pipeline, we evaluated orientation estimation using the segmentation mask alone, and found that angular errors frequently exceeded 1°, which was not acceptable for our task. Consequently, the segmentation model is used mainly for object identification and region-of-interest selection. The fine-grained geometric cues required for precise orientation estimation are then extracted via dedicated gradient-based edge localization and line fitting. This hybrid use of segmentation and classical image processing enables us to preserve both robustness and precision in the estimation process.

Nevertheless, the approach is not without its constraints. Specifically, for objects with highly curved or irregular outlines, our gradient-based edge detection followed by line fitting may fail to produce reliable orientation estimates. This is because the least-squares fitting method assumes that the detected edge points approximate a linear contour — which is not the case for curved geometries. In such scenarios, additional methods may be required. A potential extension of our current framework would be to apply the CAD-based ICP registration not only to the end-effector view, but also to the side images captured by the fixed cameras. Instead of relying on point-to-point registration, the algorithm could perform outline-to-outline registration to recover orientation from complex contours. However, this limitation did not affect our current application, as the experimental objects featured predominantly linear edges, which made them well-suited for the gradient-based line-fitting approach and enabled accurate orientation estimation using our current pipeline. From a practical deployment standpoint, the system exhibits robustness to moderate variations in ambient lighting, owing to the use of dedicated LED ring illumination and on-station LED panel lighting, which contribute to uniform and stable illumination conditions. The perception pipeline is also capable of reliably estimating object orientation even under significant angular deviations, making it resilient to imperfect placement. However, at the execution stage, limitations arise from the magnetic screwdriver, which does not mechanically secure the screw. As a result, excessive tilt can cause misalignment with the insertion axis due to gravity. This is a hardware-specific issue, not a limitation of the pose estimation pipeline. Future implementations with screw-stabilizing tools could address this and broaden tolerance to orientation errors caused by operator placement.

The final component of the RGB-only perception pipeline involves the detection and position estimation of screw holes via CAD-based registration. This module plays a critical role in the overall pipeline, serving as the final bridge between perception and execution. Its effectiveness is fundamentally enabled by the accurate calibration procedures described in Section 4.3, which ensure reliable spatial alignment between the robot's end-effector, the mounted camera, and the object's pose as observed from the on-station cameras. In practice, these calibrations remain valid over time and typically require no repetition unless mechanical changes occur — such as re-mounting the sensors, transport of the workstation, or replacing the screwdriver tip. Moreover, since the calibration methodology relies on standard static transformation estimation, it can be directly extended to other rigidly mounted tools, such as glue dispensers or welding nozzles, further enhancing the system's adaptability and generalization of the camera-to-tool calibration approach. The effectiveness of this module is directly assessed through the evaluation of standardized metrics. This leads us to the third contribution, as defined in the Introduction, concerning system accuracy and repeatability. Despite its relevance, such detailed validation based on standardized metrics is rarely reported in related screwdriving literature. Many existing works, such as [21] for example, tend to report success rates or similar simplified metrics, which, while informative, offer limited insight into the precision and consistency of the system's behavior. In contrast, our evaluation follows the ISO 9283 standard metrics, measuring both the accuracy and repeatability of the robot's end-effector positioning during screwdriving. This allows for more precise quantification of system performance, particularly

important in industrial applications where millimeter-level errors can lead to failure.

If we first consider the accuracy of the system, as presented in Table 3, we observe that the achieved positional accuracy ranges from 0.21 mm to 0.50 mm, based on validation using our optical measurement setup. To place these results into context, we evaluate them in light of the ASME B18.2.8 standard [39], which specifies recommended clearance hole sizes for bolts, screws, and studs ranging from M1.6 to M100. The standard defines three classes of clearance – close, normal, and loose fit – each with different tolerances suited to various assembly requirements.

Based on this standard, we can define the maximum allowable misalignment (Δ_{MAX}) that a robotic system can tolerate when aligning a screw with a pre-drilled hole, without causing jamming or interference during insertion. This tolerance can be calculated as:

$$\Delta_{\mathrm{MAX}} = \frac{D_h - D_s}{2}, \tag{25}$$

where D_h is the hole diameter and D_s is the nominal screw diameter. Considering the ASME B18.2.8 standard, our system's measured accuracy – ranging from 0.21 mm to 0.50 mm – is sufficient for handling M4 screws and larger under a loose fit classification. This fit type is commonly employed in general machinery applications due to its allowance for greater positional tolerance. For normal fit scenarios, our accuracy meets the requirements for M5 screws and above. In our experiments, M4 screws were used in conjunction with chamfered holes featuring an outer diameter of 5.1 mm, aligning with the ASME B18.2.8 specifications for a loose fit for M4 screws. This configuration provides a maximum allowable misalignment (Δ_{MAX}) of approximately 0.55 mm, which comfortably encompasses our system's measured positional errors. The presence of the chamfer likely facilitated smoother screw insertion, compensating for minor misalignments and contributing to the observed high success rate. The adequacy of our system's accuracy is further substantiated by a 100% success rate in screwdriving tasks performed with an industrial screwdriver.

When it comes to repeatability, the results in Table 3 show slightly higher values across all screw hole positions compared to absolute accuracy. A closer inspection of Fig. 8 reveals that the majority of the recorded end-effector positions fall within a 0.5 mm radius from the barycenter of each hole. Only a small number of measurements lie outside this range, which disproportionately affects the overall repeatability value. It is important to emphasize that these deviations are not necessarily indicative of poor system performance. In fact, the measured deviations along the y axis remain consistently within the ±0.5mm threshold across all screw holes. The slightly larger spread observed along the x axis – also visible in Fig. 8 – is not expected from the standpoint of the robotic implementation, where no axis-dependent limitations are present. Instead, these deviations may be partially attributed to limitations in the optical measurement setup used for validation. As explained in Section 5.3, the measurements are expressed in a local coordinate system defined on the object using markers M_6, M_7, and M_8, with the x axis aligned between markers M_6 and M_7. Throughout the screwdriving experiments, the normal of the plane defined by the touch-probe markers M_{15} to M_{18} was parallel to this local x axis — and coincident with the optical axis of the capturing unit. As mentioned in Section 5.1, the OptoTrak Certus system provides the highest accuracy in its $x_{\mathrm{opt}}y_{\mathrm{opt}}$ plane but exhibits reduced accuracy along the depth direction (z_{opt}), especially when markers are tilted. Our analysis indicates that the largest deviations occurred in experiments where the object was more tilted, increasing the angle between the touch probe and the optical camera system. This geometric configuration likely impaired measurement precision along the z_{opt} axis, manifesting as increased error along the local x axis in the recorded data.

Reflecting on potential limitations in screw hole registration, our method uses greedy one-to-one matching between detected hole centers and CAD references, which can be sensitive to local minima or overlapping detections. In our setup, this risk is mitigated by the Hough-based circle detector, which enforces a minimum separation between candidates, and by iterative refinement with outlier rejection (Section 4.2.2). As a result, no misalignment issues were observed in our experiments with well-structured hole configurations (Fig. 4). Nonetheless, for more cluttered or noisy scenes, incorporating robust alternatives such as RANSAC-based matching could enhance resilience and generalization. This presents a promising direction for future work, particularly in less controlled industrial environments.

8. Conclusion

In this work, we have demonstrated that a perception-driven mobile manipulator can achieve industrial-grade precision in high-mix, low-volume screwdriving tasks, successfully fulfilling each of the three contributions outlined in the Introduction.

First, we introduced a comprehensive mobile manipulation pipeline, integrating autonomous navigation, precise docking, and robust perception algorithms for object segmentation and accurate pose estimation of both assembly objects and their screw holes, thus effectively addressing localization uncertainty in dynamic assembly environments. Second, by relying solely on monocular RGB imagery, we demonstrated sub-millimeter translational and sub-degree angular precision, providing the accuracy necessary for reliable screwdriving, as evidenced by our 100% insertion success rate. Third, rigorous validation according to ISO 9283 standard's metrics confirmed the accuracy and repeatability of our approach, further underscoring its suitability for deployment in industrial settings.

Two key architectural properties of our system are generalizability and scalability. Built around standard and well-established perception modules (e.g., gradient-based edge detection, ICP-based registration), the pipeline is readily transferable to new objects and assembly configurations. The only part-specific component is the YOLOv8 segmentation model, which requires retraining upon the introduction of new objects. To minimize the associated overhead, we introduced an automatic, one-click label-propagation tool leveraging the static camera setup to efficiently propagate annotations (see Appendix). Additionally, scalability is enhanced by the parallelization of perception tasks with robotic navigation and execution, enabled by a shared network infrastructure, significantly facilitating deployment in larger and more complex multi-station environments. Importantly, system redeployment to new workstations or objects can be performed rapidly, as the robot does not require re-teaching and the core perception and control algorithms remain fully automated and object-agnostic, enabling fast adaptation with minimal software intervention.

Overall, our results highlight that combining mobility with high-precision, RGB-only perception is both feasible and advantageous for flexible manufacturing scenarios, making high-precision robotic manipulation more accessible and economically viable. While this work focused on screwdriving, the modular design of the pipeline, with its general-purpose perception and calibration components, enables adaptation to a variety of other robotic tasks. Examples include glue dispensing, inspection, or tool-based pick-and-place, where precise pose estimation and tool alignment are equally critical. In future work, we also plan to explore the transferability of these algorithms to such tasks, building on the generalization potential outlined above. Additionally, we aim to extend the pipeline to handle more complex geometries through advanced CAD-based registration methods, and validating the pipeline's implementation across multiple workstations to evaluate additional industry-relevant metrics such as cycle time, overall throughput, and robustness under varying operational conditions.

CRediT authorship contribution statement

Aleksandar Stefanov: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Miha Zorman:** Validation, Software, Methodology, Investigation, Formal analysis. **Sebastjan Šlajpah:** Validation, Formal analysis. **Janez Podobnik:** Validation, Investigation, Formal analysis. **Matjaž Mihelj:** Writing – review & editing, Validation, Supervision, Project administration, Methodology, Formal analysis, Conceptualization. **Marko Munih:** Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI models only in order to improve language and readability. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was funded by the DIGITOP project under Grant No. TN-06-0106 and from Slovenian Research and Innovation Agency (funding No. P2-0228).

Appendix. Automatic labeling: one-click label-propagation tool

To alleviate the burden of manual polygon annotation across large datasets, we adopt a one-click, class-wise auto-labeling methodology based on point-prompted segmentation. The core idea relies on the assumption that all instances of a given object class share similar visual and semantic characteristics, allowing the segmentation to be guided by a single click per class. Specifically, the method begins by parsing filenames to infer both the class label and the intended data split (training, validation, or testing), thereby enabling automatic partitioning without external metadata. For each unique class, one representative image is displayed to the annotator, who provides a single 2D click indicating the approximate location of the object. This click is used as an input prompt to a foundation segmentation model (Segment Anything Model 2, SAM2), which then propagates the segmentation across all images containing that class. The output mask with the highest confidence is selected and its external contour is extracted as a polygon. Each vertex is normalized with respect to the image dimensions to conform to standard annotation formats such as those used in YOLO-based detectors. The resulting labels, along with their corresponding images, are systematically organized into structured directories for downstream training. A complete description of this pipeline, including implementation details is provided in Algorithm 2.

Algorithm 2 One-Click Auto-Labeling Algorithm

Require: Image set $\mathcal{I} = \{I_k\}_{k=1}^{N}$, SAM predictor $\mathcal{P}$, scale factor s
Ensure: Normalized polygon annotations in YOLO format

Step 1: Class Discovery and Click Collection
Parse filenames to extract class labels $C = \{c_1, \ldots, c_M\}$
for each class $c \in C$ **do**
  Select a representative image I^c for class c
  Downscale I^c by factor s and display to annotator
  User clicks on object $\mathbf{p}_c = (x_c, y_c)$
  Rescale: $\mathbf{p}_c \leftarrow s \cdot \mathbf{p}_c$
end for
Step 2: Segmentation and Mask Extraction
for each image $I_k \in \mathcal{I}$ **do**
  Extract class c_k and split s_k from filename
  Set I_k as input to SAM predictor $\mathcal{P}$
  Predict masks $\mathcal{M} \leftarrow \mathcal{P}(I_k, \mathbf{p}_{c_k})$
  Select best mask $M^* \leftarrow \arg\max_{M \in \mathcal{M}} \text{score}(M)$
  Step 3: Polygon Extraction and Normalization
  Extract outer contour $\mathcal{V} = \{(x_i, y_i)\}$ from M^*
  Normalize: $(x_i', y_i') \leftarrow \left(\frac{x_i}{W}, \frac{y_i}{H}\right)$
  Step 4: YOLO Annotation and Packaging
  Write annotation file as:
    `<class_idx>` $x_1'\ y_1'\ x_2'\ y_2'\ \ldots$
  Move image and label to `images/`s_k`/`, `labels/`s_k`/`
end for
Generate `data.yaml` with class-to-index mapping

Data availability

The authors do not have permission to share data.

References

[1] Z. Jia, A. Bhatia, R.M. Aronson, D. Bourne, M.T. Mason, A survey of automated threaded fastening, IEEE Trans. Autom. Sci. Eng. 16 (2019) 298–310.
[2] S. Izumi, T. Yokoyama, A. Iwasaki, S. Sakai, Three-dimensional finite element analysis of tightening and loosening mechanism of threaded fastener, Eng. Fail. Anal. 12 (2005) 604–615.
[3] M. Soori, B. Arezoo, F. Karimi Ghaleh Jough, Intelligent robotic systems in industry 4.0, a review, J. Adv. Manuf. Sci. Technol. (2024) 2023.
[4] N. Ghodsian, K. Benfriha, A. Olabi, V. Gopinath, A. Arnou, Mobile manipulators in industry 4.0: A review of developments for industrial applications, Sensors 23 (2023) 8026.
[5] D. Pavlichenko, G. Martin Garcia, S.-Y. Koo, S. Behnke, KittingBot: A mobile manipulation robot for collaborative kitting in automotive logistics, Proc. 15th Int. Conf. IAS- 15 (2019) 849–864.
[6] T. Gašpar, M. Deniša, P. Radanovič, B. Ridge, T.R. Savarimuthu, A. Kramberger, M. Priggemeyer, J. Roßmann, F. Wörgötter, T. Ivanovska, S. Parizi, Ž. Gosar, I. Kovač, A. Ude, Smart hardware integration with advanced robot programming technologies for efficient reconfiguration of robot workcells, Robot. Comput.-Integr. Manuf. 66 (2020) 101979.
[7] S. Thakar, S. Srinivasan, S. Al-Hussaini, P. Bhatt, P. Rajendran, Y.J. Yoon, N. Dhanaraj, R. Malhan, M. Schmid, V. Krovi, S. Gupta, A survey of wheeled mobile manipulation: A decision making perspective, J. Mech. Robot. 15 (2022) 1–38.
[8] Y. Jiang, Z. Huang, B. Yang, W. Yang, A review of robotic assembly strategies for the full operation procedure: planning, execution and evaluation, Robot. Comput.-Integr. Manuf. 78 (2022) 102366.
[9] M.A.K. Niloy, A. Shama, R.K. Chakrabortty, M.J. Ryan, F.R. Badal, Z. Tasneem, M.H. Ahamed, S.I. Moyeen, S.K. Das, M.F. Ali, M.R. Islam, D.K. Saha, Critical design and control issues of indoor autonomous mobile robots: A review, IEEE Access 9 (2021) 35338–35370.
[10] A. Morgan, B. Wen, J. Liang, A. Boularias, A. Dollar, K. Bekris, Vision-driven compliant manipulation for reliable, high-precision assembly tasks, Robotics: Science and systems XVII, Robot.: Sci. Syst. Found. (2021).
[11] N. Correll, K.E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J.M. Romano, P.R. Wurman, Analysis and observations from the first amazon picking challenge, IEEE Trans. Autom. Sci. Eng. 15 (2018) 172–188.

[12] O. Kroemer, S. Niekum, G.D. Konidaris, A review of robot learning for manipulation: Challenges, representations, and algorithms, J. Mach. Learn. Res. (2019).
[13] Y. Sun, X. Wang, Q. Lin, J. Shan, S. Jia, W. Ye, A high-accuracy positioning method for mobile robotic grasping with monocular vision and long-distance deviation, Measurement 215 (2023) 112829.
[14] H.J. Warnecke, E. Abele, J. Walther, G. Fischer, Investigations of the screw driving process with sensor-controlled industrial robots, CIRP Ann 34 (1985) 41–44.
[15] G. Li, X. Liang, Y. Gao, T. Su, Z. Liu, Z.-G. Hou, A linkage-driven underactuated robotic hand for adaptive grasping and in-hand manipulation, IEEE Trans. Autom. Sci. Eng. 21 (2024) 3039–3051.
[16] N. Karnati, B.A. Kent, E.D. Engeberg, Bioinspired sinusoidal finger joint synergies for a dexterous robotic hand to screw and unscrew objects with different diameters, IEEE/ASME Trans. Mechatronics 18 (2013) 612–623.
[17] L. Tang, Y.-B. Jia, Y. Xue, Robotic manipulation of hand tools: The case of screwdriving, in: Proceedings of the 2024 IEEE International Conference on Robotics and Automation, ICRA, 2024, pp. 13883–13890.
[18] L. Tang, Y.-B. Jia, Robotic fastening with a manual screwdriver, in: Proceedings of the 2023 IEEE International Conference on Robotics and Automation, ICRA, 2023, pp. 5269–5275.
[19] W.-Y. Chang, M.L. Nadia Hartono, AI recognition based on multi-target images for assembling parts using robot with screwdriver and vision system, in: Proceedings of the 2023 IEEE 5th Eurasia Conference on IoT, Communication and Engineering, ECICE, 2023, pp. 920–924.
[20] A.C. Carvalho, A.I. Carvalho, R. Correia, Robot with vision guidance system for unscrewing operation, in: Proceedings of the 2024 International Conference on Decision Aid Sciences and Applications, DASA, 2024, pp. 1–5.
[21] N.M. DiFilippo, M.K. Jouaneh, A system combining force and vision sensing for automated screw removal on laptops, IEEE Trans. Autom. Sci. Eng. 15 (2018) 887–895.
[22] J. González Huarte, A. Ibarguren, Visual servoing architecture of mobile manipulators for precise industrial operations on moving objects, Robotics 13 (2024) 71.
[23] O.M. Manyar, S.V. Narayan, R. Lengade, S.K. Gupta, Physics-informed learning to enable robotic screw-driving under hole pose uncertainties, in: Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2023, pp. 2993–3000.
[24] R. Tao, F. Jing, J. Hou, S. Xing, Y. Fu, J. Fan, M. Tan, APTMRS: Autonomous prism target maintenance robotic system for FAST, IEEE Trans. Autom. Sci. Eng. 22 (2025) 4022–4038.
[25] S. Thakar, S. Srinivasan, S. Al-Hussaini, P.M. Bhatt, P. Rajendran, Y.J. Yoon, N. Dhanaraj, R.K. Malhan, M. Schmid, V.N. Krovi, S.K. Gupta, A survey of wheeled mobile manipulation: A decision-making perspective, J. Mech. Robot. 15 (2) (2022) 020801.
[26] S. Bøgh, M. Hvilshøj, M. Kristiansen, O. Madsen, Autonomous industrial mobile manipulation (AIMM): From research to industry, in: Proceedings of the 42nd International Symposium on Robotics, 2011.
[27] X. Duan, Y. Wang, M. Li, X. Kong, S.A. Aliy, A fastening bolt method based on image recognition, in: Proceedings of the 2013 IEEE International Conference on Robotics and Biomimetics, ROBIO, 2013, pp. 2709–2714.
[28] P.J. Besl, N.D. McKay, A method for registration of 3-D shapes, IEEE Trans. Pattern Anal. Mach. Intell. 14 (1992) 239–256.
[29] C. Wu, H. Fu, J.-P. Kaiser, E.T. Barczak, J. Pfrommer, G. Lanza, M. Heizmann, J. Beyerer, 6D pose estimation on point cloud data through prior knowledge integration: A case study in autonomous disassembly, Procedia CIRP 122 (2024) 193–198.
[30] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD, 1996, pp. 226–231.
[31] Y. Zhang, Z. Song, J. Yu, B. Cao, L. Wang, A novel pose estimation method for robot threaded assembly pre-alignment based on binocular vision, Robot. Comput.-Integr. Manuf. 93 (2025) 102939.
[32] S. Pitipong, P. Pornjit, W. Watcharin, An automated four-DOF robot screw fastening using visual servo, in: Proceedings of the 2010 IEEE/SICE International Symposium on System Integration, 2010, pp. 379–383.
[33] T. Hao, D. Xu, Robotic grasping and assembly of screws based on visual servoing using point features, Int. J. Adv. Manuf. Technol. 129 (2023) 1–13.
[34] G. Jocher, et al., YOLO by ultralytics, 2023, https://github.com/ultralytics/ultralytics.
[35] R.O. Duda, P.E. Hart, Use of the hough transformation to detect lines and curves in pictures, Commun. ACM 15 (1972) 11–15.
[36] International Organization for Standardization, Iso 9283:1998 – manipulating industrial robots – performance criteria and related test methods, 1998.
[37] W. Huang, Y. Liang, X. Zhang, Y. Song, Q. Dai, Polarization structured light 3D depth image sensor for scenes with reflective surfaces, Nat. Commun. 14 (1) (2023) 7200.
[38] K. Morita, et al., 3D point cloud registration and segmentation of reflective metal objects using go-ICP and improved RANSAC, in: Proceedings of the 2023 International Conference on Artificial Life and Robotics, ICAROB, ALife Robotics, 2023.
[39] The American Society of Mechanical Engineers, Asme b18.2.8:1999 (r2017) – clearance holes for bolts, screws, and studs, 1999.