Modeling, Identification and Control, Vol. 46, No. 3, 2025, pp. 111–123, ISSN 1890–1328

Pose Refinement for Reflective Workpieces using Deep Iterative Matching

O. Kaya [1] K. Thieu [1] C. Holden [1] O. Egeland [1]

[1] *Department of Mechanical and Industrial Engineering, Norwegian University of Science and Technology, N-7330 Trondheim, Norway. E-mail:* {ozan.kaya,Kevinthi,christian.holden,olav.egeland}@ntnu.no

Abstract

Determination of the pose of workpieces is important for robotic applications in manufacturing, including handling, assembly, machining and welding. Established methods based on 3D sensors may fail for workpieces with highly reflective materials. In this paper, we take advantage of recent development in machine learning to determine the pose of reflective workpieces without the use of depth data. Our proposed method is based on deep iterative matching of image data of the workpiece with a computer-aided design model. Starting with an initial estimate of the workpiece pose, the method iteratively aligns the computer-aided design model projections with an image of the actual workpiece, adjusting the pose until computer-aided design model matches the image of the workpiece. The deep learning-based approach optimizes this alignment by updating the pose estimate at each iteration, achieving high precision even for geometrically complex or reflective surfaces. This refinement process enhances accuracy in robotic applications where precise workpiece positioning is critical, such as in automated welding and assembly tasks. We use photorealistic rendering to create two datasets for pretraining the network, which reduces both training time and the need for real labeled data. After the network is trained on synthetic data, it is fine-tuned and tested on real images of reflective aluminium workpieces. We show that the proposed deep iterative matching method outperforms established methods based on iterative closest point with two 3D scanners due to large errors in the scans caused by reflections.

Keywords: Pose estimation, Machine learning, Reflective metals, Rendering, Eye-to-hand

1 Introduction

The manufacturing industry is increasingly moving toward automating repetitive tasks to reduce costs and enhance safety within the framework of Industry 4.0. Many production lines rely on specially designed fixtures to accurately set the pose of parts and use preconfigured robot trajectories to perform tasks. However, eliminating the need for part-specific fixtures enhances adaptability to new parts and tasks, significantly reducing costs for manufacturing lines with low production volumes (Kakish et al., 2000). Without part-specific fixtures, uncertainty is introduced regarding the pose of the part. A rough pose estimate may still be obtained through modular fixture systems or predefined orientations and locations of parts. To eliminate pose uncertainty, robotic systems must be equipped with sensor systems capable of detecting and accurately estimating the pose of parts (Litvak et al., 2019). Pose estimation systems are crucial in various robotic manufacturing tasks, such as handling, machining, assembly, and welding (Njaastad and Egeland, 2016; Jiang et al., 2022).

A pose estimation system consists of a sensor, an estimation method, and sensor placement. The sensor can be mounted either on the robot end-effector, referred to as eye-in-hand, or on external structures, referred to as eye-to-hand (Flandin et al., 2000). Pose

doi:10.4173/mic.2025.3.3

estimation methods can be classified according to the sensor measurement principle. Sensors are broadly divided into contact sensors, such as touch probes (Ren et al., 2020) or compliant grippers (Kaya et al., 2021), and non-contact sensors, such as computer vision (Fan and Zhao, 2015). Contact methods may need to estimate the external force (Yigit et al., 2021). Non-contact methods are generally preferred because of their higher data acquisition rates. Furthermore, vision-based methods can be based on depth data from a 3D sensor or from stereo vision methods using one or more cameras (Klingenberg et al., 2024). Technologies for integrated depth sensors that produce 3D point clouds include Time-of-Flight, laser triangulation, structured light, and stereo vision (Blais, 2004). The combination of depth scanners with iterative closest point (ICP) algorithms (Rusinkiewicz and Levoy, 2001) is a widely used approach in several industrial applications. In contrast, monocular vision is limited to workpieces containing easily distinguishable features or QR codes (Schleth et al., 2018).

When measuring reflective metals, 3D sensors often fail to produce reliable results. Time-of-Flight, laser triangulation, and structured light methods are all based on projecting light onto the measured surface. These methods fundamentally rely on the projected light being directly reflected back to the sensor, which is not the case for highly reflective metals (Blais, 2004). Common artifacts in depth sensor scans of reflective parts include missing detections and false geometries caused by clustered surface outliers (Shen et al., 2009). Stereo vision also has challenges when triangulating by matching pixels from two views, as the appearance of the same point on a reflective surface can change when observed from different perspectives (Bhat and Nayar, 1998). Research has been conducted to address reflections in laser scanning (Alstad and Egeland, 2022; Marco-Rider et al., 2022), and (Njaastad and Egeland, 2016) employs a structured light scanner that effectively mitigates reflections to a significant extent. However, the limited range of these scanners makes them impractical for eye-to-hand configurations, where greater measuring distances are required. For larger parts, a combination of eye-in-hand and eye-to-hand methods can be advantageous. The eye-in-hand configuration provides an accurate partial view of the object, while the eye-to-hand setup offers a less accurate but broader global perspective (Flandin et al., 2000).

Early monocular vision methods for pose estimation led to challenges such as handling texture-less objects and partial occlusions (Li et al., 2018). However, advances in processing power and the rise of machine learning have significantly improved monocular vision-based pose estimation. The BOP benchmark includes several datasets that compare a wide range of pose estimation methods (Hodaň et al., 2020). Among these bench-marked datasets, we find the texture-less dataset T-LESS (Hodan et al., 2017) to be the most representative of metal workpieces, as both share the challenge of uniform textures, which complicates pose estimation (Park et al., 2019).

We consider an eye-to-hand setup for determining the pose of reflective aluminum workpieces in this paper. We apply CosyPose which is RGB-based pose estimation method with a standard monocular camera, and compare it to ICP with scans from two 3D sensors using different range sensing technology. We create a pipeline for training CosyPose on synthetic data, and show that relatively few labeled examples are required to fine-tune the network for real data.

This paper is structured as follows: Section 2 provides an overview of related works. Section 3 details the methodology for pose estimation of reflective workpieces, including the network input, network structure, loss function, synthetic datasets, training of DeepIM, and evaluation metrics. Section 4 describes the experiments, which include simulated data, a real dataset, depth scanners with ICP, and their comparison. Section 5 presents the results and discussion, followed by Section 6, which concludes the paper by summarizing the key findings and suggesting avenues for future research.

2 Overview of Related Work

Monocular object pose estimation has traditionally relied on feature matching techniques such as SIFT and SURF to identify correspondences between a 3D model and an observed image (Rothganger et al., 2006; Lowe, 1999). Once 2D-3D correspondences are established, the Perspective-N-Point (PNP) algorithm, combined with RANSAC, is used to estimate the object's pose while filtering out incorrect matches (Fischler and Bolles, 1981). However, these methods are less effective for textureless objects because of the lack of distinctive features. Another approach is template matching, where predefined templates, such as rendered images, are compared against the observed image to determine the best-matching viewpoint (Hinterstoisser et al., 2011; Jurie and Dhome, 2001). This method evaluates multiple sampled viewpoints and selects the one with the highest similarity score. However, template matching is highly sensitive to occlusions, as increased obstruction reduces the matching score and often leads to inaccurate pose estimations.

Recent advances in pose estimation have increasingly taken advantage of machine learning techniques.

Early approaches employed neural networks to directly regress the pose of an object. For example, BB8 estimates the pose by predicting the 2D projections of a 3D bounding box around the object (Rad and Lepetit, 2017), while SSD-6D classifies the pose by selecting from a predefined set of discrete viewpoints (Kehl et al., 2017). More recent methods have shifted toward predicting 2D-3D correspondences using machine learning, followed by solving the pose with PnP-RANSAC. This strategy has been shown to improve accuracy compared to direct pose regression (Park et al., 2019; Hodan et al., 2020; Li et al., 2019). The main differences among these approaches lies in how they establish correspondences. Pix2Pose (Park et al., 2019) utilizes an encoder-decoder network to infer object surface coordinates from the observed image. CDPN (Li et al., 2019) estimates rotation through correspondence prediction and PnP while handling translation separately via a dedicated network head. EPOS (Hodan et al., 2020), on the other hand, learns a dense mapping between image pixels and object surface fragments.

DeepIM (Li et al., 2018) introduced a render-and-compare approach for pose refinement, where a neural network iteratively predicts pose updates starting from an initial estimate. The method takes as input both the observed image and a rendered image generated from the current pose estimate. The network then outputs a pose update, which is applied to refine the estimate. This process is repeated iteratively, with each refined pose used to render a new image for the next iteration. A key innovation in DeepIM was making the pose update prediction independent of the 3D model's coordinate system. Instead of directly estimating the model's pose, the network predicts the camera's pose update. This transformation ensures that rotation and translation predictions remain decoupled. Specifically, DeepIM represents rotation using a unit quaternion and introduces a translation representation that mitigates the scale-distance ambiguity inherent in monocular vision. This scale-invariant translation formulation allows the network to generalize across varying depth distributions and object scales, even those different from its training data.

CosyPose (Labbé et al., 2020) builds on DeepIM by incorporating a more advanced backbone network and a continuous rotation representation. One notable modification is the adjustment of the translation representation: CosyPose replaces DeepIM's logarithmic depth scaling with a linear scaling approach. Additionally, the method extends to multi-camera setups and explicitly accounts for object symmetries, improving robustness. Due to these enhancements, CosyPose was recognized as the top-performing RGB-based pose estimation method in the 2020 BoP Challenge (Hodaň et al., 2020).

3 Method

This section presents the pose estimation method designed for reflective surfaces. The proposed approach builds on the deep iterative matching framework introduced by DeepIM (Li et al., 2018), aiming to predict the transformation between an object frame and a camera frame. The method assumes a coarse initial pose estimate, obtained either through modular fixtures or predefined object placement within the workspace. The system refines this initial estimate by rendering the CAD model and comparing it with the observed image using a neural network. To enhance performance, we integrate improvements from CosyPose, utilizing EfficientNet-B3 (Tan and Le, 2019) as the backbone network, along with linear depth scaling, a disentangled loss function, and a 6D rotation representation (Labbé et al., 2020). The model is first trained on simulated data and then fine-tuned on a limited real-world dataset. Finally, we evaluate the deep iterative matching approach against robust ICP, using point cloud data from two depth scanners.

3.1 Network Input

A calibrated camera with a known camera matrix $\boldsymbol{K}$ captures an image of a workpiece, assuming an initial pose estimate $\boldsymbol{T}$ relative to the camera. The network processes two inputs: a cropped section from an undistorted real image and a rendered image of the workpiece based on the initial pose estimate. To extract the crop, the rendered image is used to define a square bounding box around the projected model, as illustrated in Fig. 1.

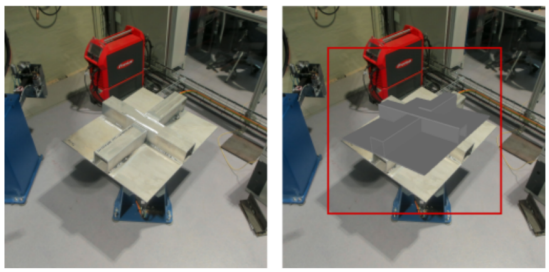

Figure 1: Left: Camera image of workpiece. Right: A crop from the real image created around the rendering with the initial pose estimate.

The width and height of the bounding box is set to w_{bb}, and p pixels of padding is applied with

$$p = w_{bb} s \tag{1}$$

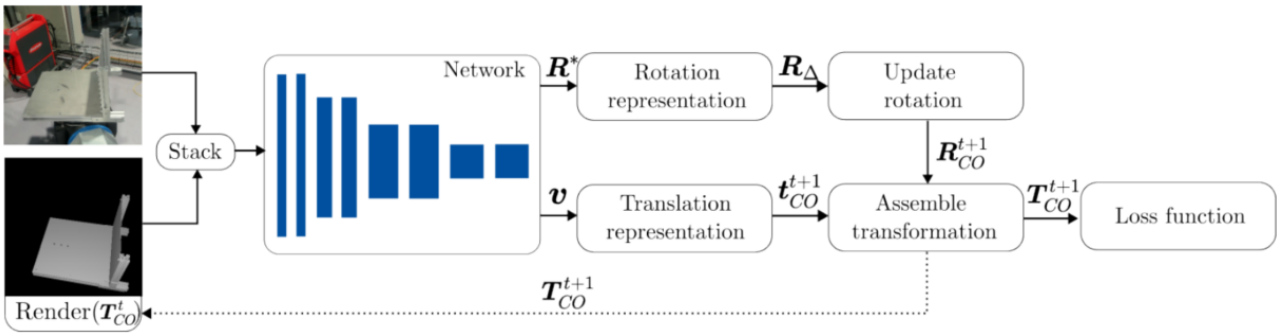

Figure 2: Deep iterative matching architecture. An image from a camera and a render with the current pose estimate is stacked and used as input to a neural network. The network outputs a rotation and translation representation which is converted to a transformation matrix. During training, the new transformation matrix is used to calculate the loss. The predicted transformation matrix can be used to render a new input, creating an iterative process.

where s is the percentage of padding that is applied. The image crops are then resized to a predefined width and height since the network requires the same image size for the input.

To generate the rendered image, it must be rendered using the initial transformation matrix $\boldsymbol{T}$ along with the adjusted camera matrix corresponding to the cropped and scaled image. The camera matrix $\boldsymbol{K}$ of the original image is given as:

$$\boldsymbol{K} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where f_x and f_y are the effective focal lengths expressed in pixel units along the image x and y axes, respectively.The principal point $(u_{0,c}, v_{0,c})$ of the cropped camera matrix is then shifted according to

$$u_{0,c} = u_0 - x_0 \tag{3}$$

$$v_{0,c} = v_0 - y_0 \tag{4}$$

where x_0 and y_0 are the pixel location of the top left corner of the crop relative to the original image. The camera matrix of the cropped and scaled image is given as

$$\boldsymbol{K}' = \begin{bmatrix} \frac{w_2}{w_1} f_x & 0 & \frac{w_2}{w_1} u_{0,c} \\ 0 & \frac{w_2}{w_1} f_y & \frac{w_2}{w_1} v_{0,c} \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

where w_1 and w_2 are the widths of the unscaled and scaled crop respectively.

3.2 Network Architecture

An illustration of the deep iterative matching architecture is presented in Fig. 2, where the network predicts the transformation $\boldsymbol{T}_{CO}$ between the object frame O and the camera frame C. The network inputs the target camera image and a rendered image based on the current pose estimate as input, and outputs a 6D rotation representation $\boldsymbol{R}^*$ along with a 3D translation vector $\boldsymbol{v}$ in pixel space. In Zhou et al. (2019), a continuous 6D rotation representation was proposed, where the network predicts two axes of the rotation matrix. The first three elements of $\boldsymbol{R}^*$ are denoted as $\boldsymbol{r}_1^*$, and the remaining three outputs are denoted as $\boldsymbol{r}_2^*$. The full rotation matrix is recovered through orthogonalization, as follows:

$$\boldsymbol{r}_1 = \frac{\boldsymbol{r}_1^*}{\|\boldsymbol{r}_1^*\|} \tag{6}$$

$$\boldsymbol{r}_3 = \frac{\boldsymbol{r}_1 \times \boldsymbol{r}_2^*}{\|\boldsymbol{r}_1 \times \boldsymbol{r}_2^*\|} \tag{7}$$

$$\boldsymbol{r}_2 = \boldsymbol{r}_3 \times \boldsymbol{r}_1 \tag{8}$$

where $\times$ is the cross product. The rotation matrix is then assembled with the orthonormal vectors as

$$\boldsymbol{R}_\Delta = \begin{bmatrix} | & | & | \\ \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 \\ | & | & | \end{bmatrix} \tag{9}$$

The current rotation estimate from the camera to the object, $\boldsymbol{R}_{CO}$ is updated from iteration t to $t+1$ with

$$\boldsymbol{R}_{CO}^{t+1} = \boldsymbol{R}_\Delta \boldsymbol{R}_{CO}^t \tag{10}$$

To create a translation representation that is independent of the depth-scale ambiguity in monocular vision, the network predicts a translation in pixel space $[v_x, v_y]$ and a depth scaling v_z (Li et al., 2018). The translation output from the network is converted from pixel space and depth scaling, to a translation vector in Euclidean space with

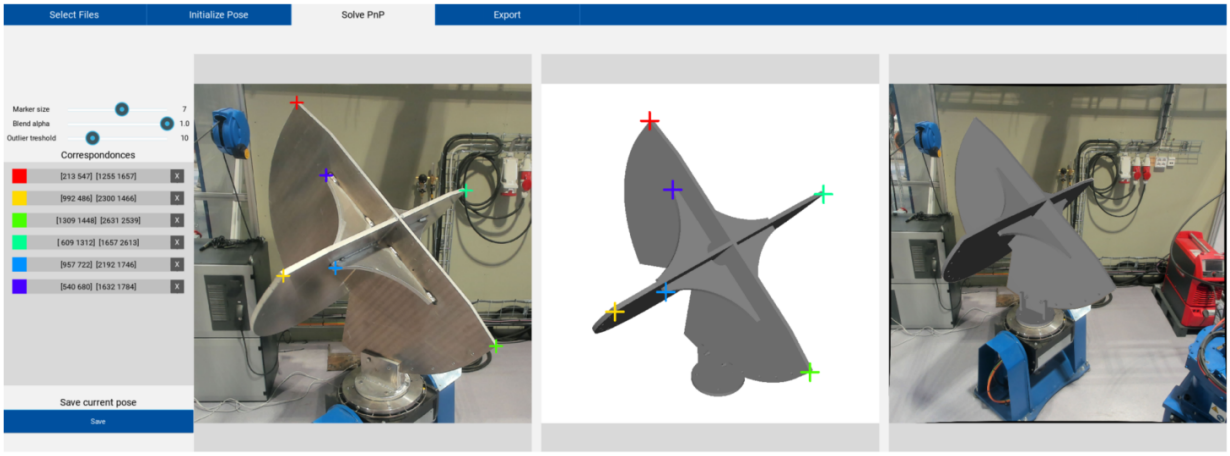

Figure 3: Pose annotation program for manual ground truth labeling. Given a camera image, the corresponding CAD model is rotated to the approximate orientation of the model in the image. At least five correspondences are then manually selected and the pose is solved with PnP.

In order to eliminate the depth-scale ambiguity inherent in monocular vision, the network predicts the translation components in pixel space $[v_x, v_y]$ and a depth scaling factor v_z (Li et al., 2018). The predicted translation is then transformed from pixel space and depth scaling into a translation vector in Euclidean space using the following approach:

$$z^{t+1} = v_z z_t \tag{11}$$

$$x^{t+1} = \left(\frac{v_x}{f'_x} + \frac{x^t}{z^t}\right) z^{t+1} \tag{12}$$

$$y^{t+1} = \left(\frac{v_y}{f'_y} + \frac{y^t}{z^t}\right) z^{t+1} \tag{13}$$

where f'_x and f'_y are the focal length from the camera matrix given in Equation 5. The predicted translation vector is then given by

$$\boldsymbol{t}_{CO}^{t+1} = \begin{bmatrix} x^{t+1} & y^{t+1} & z^{t+1} \end{bmatrix}^T \tag{14}$$

The updated pose estimate, denoted as $\boldsymbol{T}_{CO}^{t+1}$, is obtained by combining the predicted rotation $\boldsymbol{R}^{t+1}$ with the corresponding translation. This updated pose is then used to generate a new input for the network, which drives the iterative refinement process. Each time the network updates the pose, it constitutes one iteration of prediction. In order to adjust the network weights, a loss function is required to compare the updated pose estimate $\boldsymbol{T}_{CO}^{t+1}$ with the ground truth pose $\boldsymbol{T}_{GT}$. The equations should be implemented within an automatic differentiation framework, allowing the gradients of the loss function with respect to the network to be computed.

3.3 Loss Function

The network predicts a transformation matrix between the frame of the camera and the object. Based on the predicted transformation matrix $\boldsymbol{T}$ and ground truth transformation matrix $\boldsymbol{T}_{GT}$, we want to express a loss based on the difference between the two transformations. Since $SE(3)$ has no bi-invariant metric, it requires a separate scaling for the angular and translation distance. Choosing the rotation and translation scaling causes ambiguity regarding how to balance the losses. A better option is to use the average distance ADD between the transformed points on the model, with the estimated pose and ground truth pose (Kendall and Cipolla, 2017). The ADD loss with the l_1 distance metric is given by

$$\text{ADD}(\boldsymbol{T}, \boldsymbol{T}_{GT}) = \frac{1}{N} \sum_{n=1}^{N} |\boldsymbol{T}\boldsymbol{x}_i - \boldsymbol{T}_{GT}\boldsymbol{x}_i| \tag{15}$$

where $\boldsymbol{x}_i$ are points sampled from the surface of the 3D model. The function calculates the average absolute value of the distance between points transformed by $\boldsymbol{T}$ and $\boldsymbol{T}_{GT}$. Building upon this loss function, CosyPose uses a concept from (Simonelli et al., 2019), by disentangling the rotation, image plane translation, and depth prediction. Denoting the construction of a transformation matrix with

$$\boldsymbol{T}(\boldsymbol{R}, x, y, z) = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \tag{16}$$

where $\boldsymbol{t} = [x, y, z]^T$, the disentangled rotation is given by

$$\boldsymbol{T}_R = \boldsymbol{T}(\boldsymbol{R}, x_{GT}, y_{GT}, z_{GT}) \tag{17}$$

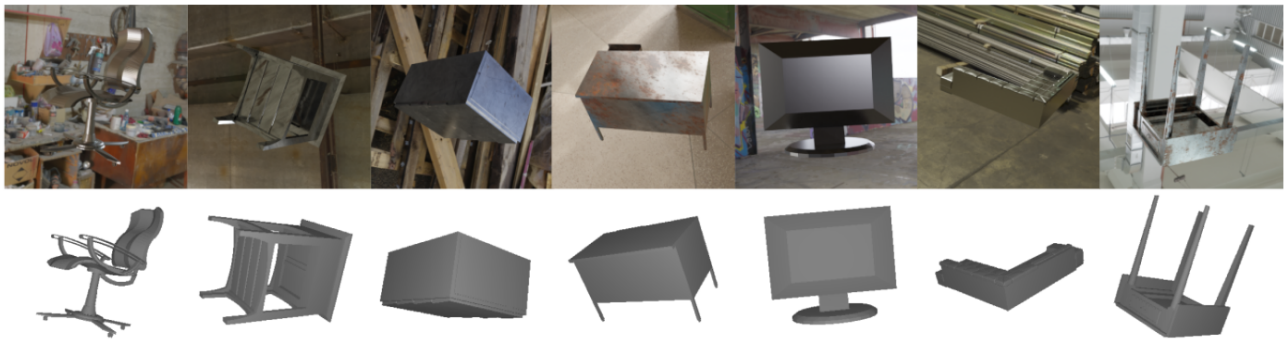

Figure 4: Cropped training examples from dataset rendered with ModelNet10. The top row shows images rendered in Blender with reflective metal textures and industrial backgrounds. The bottom row shows the corresponding 3D model rendered in Pyrender.

the disentangled $[x, y]$ translation by

$$\boldsymbol{T}_{[x,y]} = \boldsymbol{T}(\boldsymbol{R}, x, y, z_{GT}) \tag{18}$$

and the disentangled depth by

$$\boldsymbol{T}_z = \boldsymbol{T}(\boldsymbol{R}_{GT}, x_{GT}, y_{GT}, z) \tag{19}$$

The disentangled ADD loss, which we denote as ADD_D, is then given by

$$\begin{aligned} \text{ADD}_D = \quad & \text{ADD}(\boldsymbol{T}_R, \boldsymbol{T}_{GT}) + \\ & \text{ADD}\left(\boldsymbol{T}_{[x,y]}, \boldsymbol{T}_{GT}\right) + \\ & \text{ADD}\left(\boldsymbol{T}_z, \boldsymbol{T}_{GT}\right) \end{aligned} \tag{20}$$

3.4 Datasets

CosyPose requires a large amount of training time and data. To reduce the need for labeled realworld data, we render synthetic data to pretrain the network. Blender (Community, 2018) is used to render photorealistic training examples, where 3D models are given reflective metal materials. We apply domain randomization by using a mix of procedurally generated materials within Blender and Physically Based Rendering (PBR) textures, to better adapt from simulation to real data (Tobin et al., 2017). Industrial high dynamic range images are applied in Blender to create realistic lighting and background for the rendered scenes. The camera position is randomized and pointed toward the scene origin.

Two simulated datasets are created. The first dataset is created with large a variety of 3D models from ModelNet10 as given in Fig. 4. The trained model from this dataset serves as a baseline for training on specific 3D models, such that training time is reduced. The second smaller dataset uses the specific 3D models that we will use with the real data.

Real pose estimation datasets are convenient to generate with the use of 3D scanners to label a vast amount of data with labeled ground truth (Hodaň et al., 2020; Qin et al., 2023). Since no 3D scanner to our knowledge works well on reflective surfaces for longer measurement distances, we create a program for manually labeling the ground truth poses. We calibrate a camera, capture images from unique viewpoints of each workpiece and undistort the images such that image projections are given by a linear map with the camera matrix (Duda and Frese, 2018). Given an image of a workpiece, the corresponding 3D model is rotated in the program such that it roughly matches the orientation in the camera image. At least 5 corresponding pixels from the image and the render of the 3D model are selected. The selected pixels from the render of the CAD model are projected to 3D points using the depth pass of the render, such that 2D-3D correspondences between the model and real image are established. The pose is then solved using the PnP-RANSAC implementation from OpenCV (Bradski et al., 2000).

3.5 Training

DeepIM (Li et al., 2018) found that when the network was trained to regress the pose update in a single step, the predictions during testing did not improve over several iterations. Better results were found by using the predicted pose from the network to generate new examples during training for a set amount of iterations. In this paper, we start training without any prediction iterations to get the first prediction stable and gradually increase the prediction iterations throughout the training process.

3.6 Evaluation Metrics

The methods are evaluated on ADD L2 distance, rotation and translation error, and the $(k^o, k \text{ cm})$ threshold. First, we use the ADD distance metric as given

in Equation 15, but with the Euclidean L2 distance instead of the L1 distance between the sampled points as proposed in (Hinterstoisser et al., 2013). For the rotation error the angle between the predicted rotation matrix $\boldsymbol{R}$ and ground truth $\boldsymbol{R}_{GT}$ is calculated with

$$\theta = \arccos \frac{tr(\boldsymbol{R}^T \boldsymbol{R}_{GT}) - 1}{2} \tag{21}$$

The translation error is calculated as the L2 distance between $\boldsymbol{t}$ and $\boldsymbol{t}_{GT}$. The $(k^o, k$ cm) metric proposed in (Shotton et al., 2013) evaluates a pose estimate as correct if both the translation and rotation are within k cm and k degrees with respect to the ground truth. The methods are evaluated for several values of k being 1, 2, and 5.

4 Experiments

In this section, we describe the experimental setup and compare deep iterative matching with two depth scanners. Two scanners with different range sensing technologies are chosen, the Intel Realsense D435i and Intel Realsense L515. The 3D point clouds from the scanners are used with a robust ICP method to refine the pose. The methods are compared on aluminium workpieces with corresponding CAD models shown in Fig. 5.

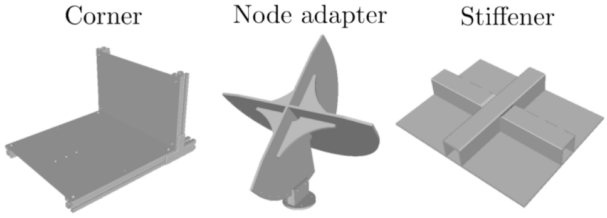

Figure 5: 3D models of aluminum workpieces used for testing.

4.1 Simulated data

Using 3D models from ModelNet10 (Wu et al., 2015), we render 25000 training examples and 160 validation examples. The 3D models are normalized to fit within a unit cube to ensure that the whole model is seen from the camera. Half of the models are given a randomized reflective appearance with the BSDF material in Blender, while the other half is given a random industrial PBR texture from (Polyhaven, 2024). The scene is rendered to a square image size with 720 pixels in both width and height. A crop is created around the initial pose estimate of the workpiece as described in Section 3.1, with 20% padding on the bounding box according to Equation 1.

The geometric parameters of the scene and initial pose estimate are given in Table 1, where the camera is denoted with subscript C, the scene world origin W, the ground truth pose GT and the initial pose of the object I. The rotations R are sampled uniformly around a random axis and the translations t are sampled uniformly in a random direction. Examples from the training set are shown in Fig. 4. The network is trained with a batch size of 8 for 600 000 batches. We use the Adam (Kingma, 2014) optimizer with a learning rate of $3 \cdot 10^{-4}$ and the disentangled loss function from Equation 20. The training starts without the use of prediction iterations but is incremented by one for every 100 000 batches after batch 200 000. For the workpiece dataset, we use the same method and create a dataset with 10 000 training examples and 160 validation examples. The network is then trained for 50 000 batches using five prediction iterations. The weights of the network are saved every time a new lowest validation loss is achieved.

Table 1: Geometric parameters for Blender scene.

Scene parameter	Min	Max	Unit
$\lVert \boldsymbol{R}_{CW} \rVert$	0	360	deg
$\lVert \boldsymbol{t}_{CW} \rVert$	2.6	3.0	m
$\lVert \boldsymbol{R}_{GT,I} \rVert$	0	30	deg
$\lVert \boldsymbol{t}_{GT,I} \rVert$	0	0.3	m

4.2 Real dataset

To generate a real dataset we capture images of three aluminum workpieces with a Logitech Brio camera from a distance of 1.8 m to 2.2 m. Using the manual pose annotation program we create 210 annotated examples for fine-tuning, 30 for validation, and 90 for testing. The images are cropped and resized to an image size of 320 by 320 pixels around the initial pose estimate. The initial pose is given a uniformly random perturbation of up to 30° around a random unit axis and a 30 cm translation offset. We randomize the initial pose estimate for the training examples, making each crop from the real image in the training examples unique. The network is trained for 10 000 batches with five prediction iterations with the same parameters as with the simulated data.

4.3 Depth scanners and ICP

We compare our method with two 3D scanners using different range sensing technology. The Intel Realsense D435i uses stereo vision with infrared light projection and the Intel Realsense L515 uses time-of-flight (Keselman et al., 2017; Lourenço and Araujo, 2021). We

scan the workpieces from the same viewpoints distribution as the machine learning dataset, creating 90 test examples. The ground truth poses are annotated using the images from the RGB sensor with the pose annotation program described in Section 3.4. Using the scans from the sensors and the same distribution of initial pose estimates as the machine learning methods, the pose is refined with iterative closest point. We choose the pointto-plane method implemented in Open3D, with Tukey loss as the robust kernel (Zhou et al., 2018). The discriminator parameter k between outliers and inliers for the Tukey loss is set according to the depth accuracy of the scanners. For the D435i scanner, we set $k = 0.04$ and for the L515 scanner we set $k = 0.014$.

4.4 Comparison

To compare the methods we use the evaluation methods described in Section 3.6 across three different ranges of initial error of the pose estimate. The initial pose estimate deviates from the ground truth with the following ranges

- $0° - 10°$, $0.0\,\text{m} - 0.1\,\text{m}$
- $10° - 20°$, $0.1\,\text{m} - 0.2\,\text{m}$
- $20° - 30°$, $0.2\,\text{m} - 0.3\,\text{m}$

For each test example, three initial pose estimates in the given rotation and translation range are tested. The median results are reported for ADD, translation and angle deviation between the predicted pose and ground truth pose. The robustness of the methods is given through the ($k°$, k cm) metric, which states the percentage of examples within the angle and translation threshold. To compare the effect of pretraining on simulated data, we also train a network from scratch on real data. We evaluate the benefits of pretraining on simulated data in terms of validation loss.

5 Results and discussion

In this section, we show and discuss the results of the experiments. Results are reported with the ADD error metric in Equation 15, with the l_2 distance metric. The average numeric results for ADD, rotation, and translation are shown in Table 2 and the percentage of examples within the ($k°, k$ cm) metric is shown in Table 3.

Each prediction iteration takes an average of 0.0912 seconds with an RTX 3090 GPU and AMD 3900x CPU. The validation loss from the synthetically pretrained model and the model trained from scratch are shown in Fig. 6.

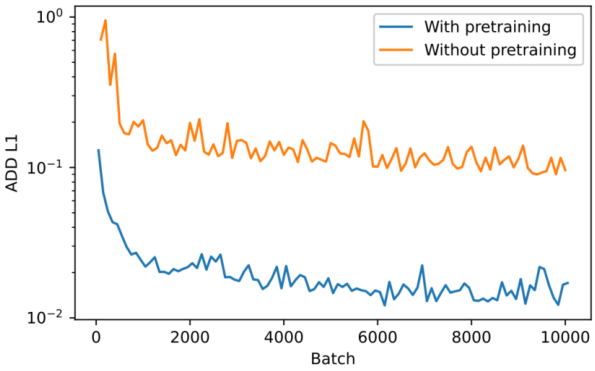

Figure 6: Validation loss compared for synthetically pretrained model and model trained from scratch on real dataset.

The plot shows that the synthetically pretrained model reached a validation loss that is an order of magnitude lower than the model trained from scratch. An example from the ModelNet10 and workpiece set is shown in Fig. 7 which shows that the trained model had a capability of pose estimation.

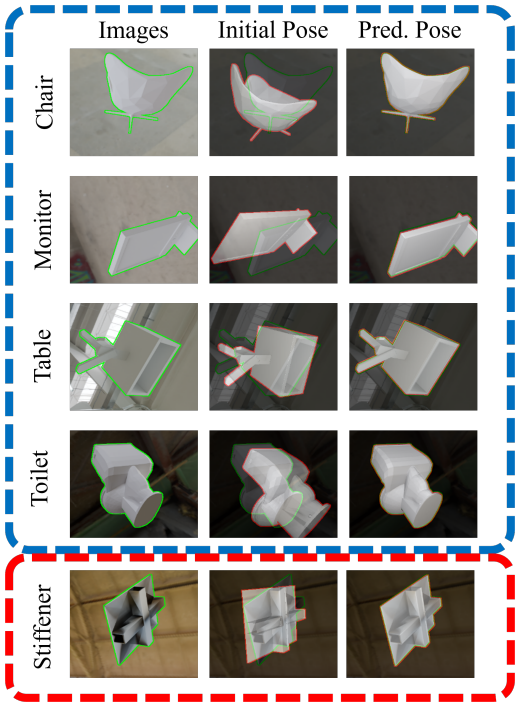

Figure 7: ModelNet10 (blue dash line) and workpiece (red dash line) set examples with CosyPose.

Table 2: Median numeric results for ADD L2 error, rotation error in degrees, and translation error in meters.

Part	**Node adapter**			**Corner**			**Stiffener**		
Metric	ADD	t.err	r.err	ADD	t.err	r.err	ADD	t.err	r.err
Initial error: $0° - 10°, 0.0\text{m} - 0.1\text{m}$									
D435i & ICP	0.0815	0.0787	3.676	0.0698	0.0659	4.488	0.0667	0.0644	5.448
L515 & ICP	0.0423	0.0365	3.579	0.0461	0.0406	2.967	0.0348	0.0331	2.346
CosyPose	0.0167	0.0154	0.7831	0.0080	0.0073	0.7848	0.0153	0.0146	0.6889
Initial error: $10° - 20°, 0.1\text{m} - 0.2\text{m}$									
D435i & ICP	0.1074	0.1019	6.025	0.0832	0.0804	5.703	0.1065	0.0946	8.23
L515 & ICP	0.1076	0.0917	7.504	0.0578	0.0498	5.604	0.0754	0.069	2.622
CosyPose	0.0154	0.014	0.7912	0.0079	0.0076	0.8407	0.0153	0.0179	0.7285
Initial error: $20° - 30°, 0.2\text{m} - 0.3\text{m}$									
D435i & ICP	0.1527	0.1494	10.35	0.1284	0.1201	10.16	0.1559	0.1365	13.94
L515 & ICP	0.1789	0.1469	12.09	0.1279	0.116	13.86	0.1495	0.1442	5.351
CosyPose	0.0156	0.0153	0.8593	0.0097	0.0102	0.966	0.0168	0.0175	0.7181

The images prepared synthetically are shown to the left, the initial pose estimate in the middle, and the final pose estimate after five prediction iterations to the right. An example from the test set of the two models is shown in Fig. 8 which shows that the pretrained model had a much better pose estimate.

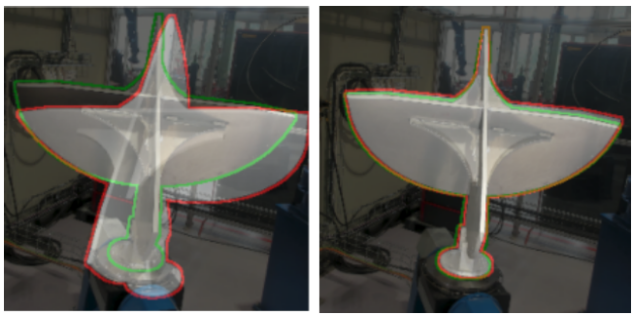

Figure 8: Test set example comparing model trained from scratch (left) and the synthetically pretrained model (right). The render with the ground truth pose is outlined in green and the pose estimate in red.

CosyPose clearly outperformed the ICP scanners by a large margin. CosyPose achieved similar accuracy for all the three initial pose deviations. The depth scanners faced significant problems when scanning the reflective aluminium workpieces. Scans were severely corrupted by missing points as shown in Fig. 9, and clustered surface outliers as shown in Fig. 10. Missing points from whole surfaces of the model, gave the ICP too few points to register the workpiece. Clustered surface outliers caused problems because the ICP registered false outliers as inliers along errors of the scan that resembled the real geometry.

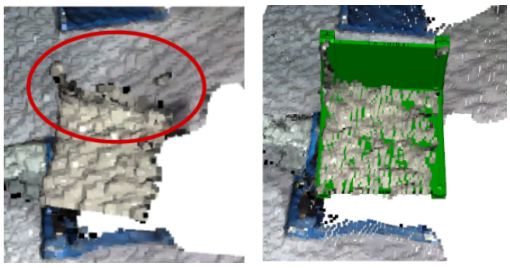

Figure 9: Missing points outlined in red from a scan of the corner workpiece. The raw scan is shown to the left and the same scan with the ground truth pose of the model to the right.

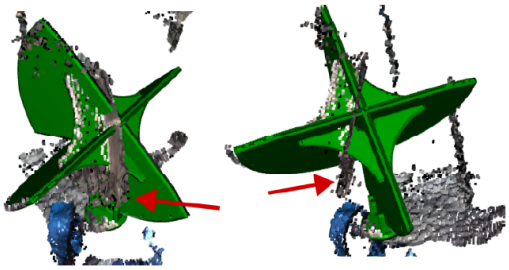

Figure 10: A scan shown from two viewpoints with the ground truth pose of the model in green. Red arrows point to clustered measurements that systematically deviate from the true object surface.

Table 3: Percentage of test examples within the $(k°, k\,\text{cm})$ metric for k equal 5, 2 and 1.

Part	**Node adapter**			**Corner**			**Stiffener**		
Metric	5cm,5°	2cm, 2°	1cm, 1°	5cm, 5°	2cm, 2°	1cm, 1°	5cm, 5°	2cm, 2°	1cm, 1°
Initial error: $0° - 10°, 0.0\text{m}-0.1\text{m}$									
D435i & ICP	5	0	0	17	6	0	10	0	0
L515 & ICP	66	27	0	65	20	0	94	26	0
CosyPose	100	74	23	100	97	53	100	78	20
Initial error: $10° - 20°, 0.1\text{m}-0.2\text{m}$									
D435i & ICP	3	0	0	16	6	0	1	0	0
L515 & ICP	17	4	0	35	13	0	46	14	0
CosyPose	100	71	23	100	98	48	100	68	21
Initial error: $20° - 30°, 0.2\text{m}-0.3\text{m}$									
D435i & ICP	1	0	0	8	3	0	1	0	0
L515 & ICP	5	1	0	10	4	0	18	7	0
CosyPose	100	57	12	100	88	25	100	63	17

More examples from the test set of the pretrained model are shown in Fig. 11.

Figure 11: Test set examples with CosyPose. The image from the camera is shown to the left, the initial pose estimate in the middle, and the final pose estimate after five prediction iterations to the right.

The deep iterative matching implementation is significantly more complex than ICP. An apparent drawback is that the network must be trained for specific models in contrast to a depth sensor and ICP. However, monocular vision does not suffer from sensor artifacts commonly found with depth sensors and reflective surfaces. Furthermore, the deep iterative matching method is independent of depth given that the crop of the model in the camera image is of sufficient resolution. Deep iterative matching is therefore viable for larger structures where the sensor placement is too far away for depth scanners to work. Cameras are also generally cheaper and more available than custom depth sensors.

The accuracy achieved with the eye-to-hand setup using deep iterative matching may be sufficient for tasks such as gripping and handling. For tasks such as machining or welding where high precision is required, an option is to combine deep iterative matching with a more accurate eye-in-hand sensor. The more accurate robot mounted sensor can then use the pose estimate to inspect local parts of the workpiece from shorter measuring distances. For further work, the method described in this paper can be extended to work without the assumption of an initial pose through predefined placement or modular fixtures. This can be achieved by combining deep iterative matching with a less accurate method that produces an initial pose estimate such as BB8 (Rad and Lepetit, 2017) or SSD-6D (Kehl et al., 2017). The accuracy and robustness of the deep iterative matching method may be further improved by using more cameras and training data.

6 Conclusion

In this paper, we have demonstrated the effectiveness of deep iterative matching for the determination of the

pose of reflective workpieces without the use of depth data. By using photorealistic rendering to create synthetic datasets for pretraining the network, we were able to reduce the need for real labeled data and train the network more efficiently. Through our experiments with reflective aluminum workpieces, we found that our deep iterative matching method outperformed ICP with 3D scanners, due to the large errors in the scans caused by reflections. Overall, our results have demonstrated that the proposed approach is a promising solution for estimating the pose of reflective parts in robotic manufacturing applications.

Acknowledgments

This work was funded by the Norwegian Research Council under the project Robotic Welding of Large Aluminium Hull Structures, Project Number 295138.

This paper has benefited from the work of MSc Ola Alstad, who worked on a preliminary version of the paper.

References

Alstad, O. and Egeland, O. Elimination of Reflections in Laser Scanning Systems with Convolutional Neural Networks. *Modeling, Identification and Control*, 2022. 43(1):9–20. doi:10.4173/mic.2022.1.2.

Bhat, D. N. and Nayar, S. K. Stereo and specular reflection. *International Journal of Computer Vision*, 1998. 26:91–106. doi:10.1023/A:1007940725322.

Blais, F. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 2004. 13(1):231–243. doi:10.1117/1.1631921.

Bradski, G., Kaehler, A., et al. Opencv. *Dr. Dobb's journal of software tools*, 2000. 3(2).

Community, B. O. *Blender - A 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL `http://www.blender.org`.

Duda, A. and Frese, U. Accurate detection and localization of checkerboard corners for calibration. In *BMVC*, volume 126. pages 1–11, 2018.

Fan, Y. and Zhao, B. Combined non-contact coordinate measurement system and calibration method. *Optics & Laser Technology*, 2015. 70:100–105. doi:10.1016/j.optlastec.2015.01.001.

Fischler, M. A. and Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 24(6):381–395. doi:10.1145/358669.358692.

Flandin, G., Chaumette, F., and Marchand, E. Eye-in-hand/eye-to-hand cooperation for visual servoing. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 3. IEEE, pages 2741–2746, 2000. doi:10.1109/ROBOT.2000.846442.

Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., and Lepetit, V. Gradient response maps for real-time detection of texture-less objects. *IEEE transactions on pattern analysis and machine intelligence*, 2011. 34(5):876–888. doi:10.1109/TPAMI.2011.206.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I 11*. Springer, pages 548–562, 2013. doi:10.1007/978-3-642-37331-2_42.

Hodan, T., Barath, D., and Matas, J. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pages 11703–11712, 2020. doi:10.48550/arXiv.2004.00605.

Hodan, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., and Zabulis, X. T-less: An RGB-D dataset for 6D pose estimation of texture-less objects. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pages 880–888, 2017. doi:10.1109/WACV.2017.103.

Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E., Michel, F., Rother, C., and Matas, J. Bop challenge 2020 on 6d object localization. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, pages 577–594, 2020. doi:10.1007/978-3-030-66096-3_39.

Jiang, Y., Huang, Z., Yang, B., and Yang, W. A review of robotic assembly strategies for the full operation procedure: planning, execution and evaluation. *Robotics and Computer-Integrated Manufacturing*, 2022. 78:102366. doi:10.1016/j.rcim.2022.102366.

Jurie, F. and Dhome, M. Real time 3d template matching. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1. IEEE, pages I–I, 2001. doi:10.1109/CVPR.2001.990559.

Kakish, J., Zhang, P.-L., and Zeid, I. Towards the design and development of a knowledge-based universal modular jigs and fixtures system. *Journal of Intelligent Manufacturing*, 2000. 11:381–401. doi:10.1023/A:1008978319436.

Kaya, O., Tağlıoğlu, G. B., and Ertuğrul, S. The series elastic gripper design, object detection, and recognition by touch. *Journal of Mechanisms and Robotics*, 2021. 14(1):014501. doi:10.1115/1.4051520.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., and Navab, N. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE international conference on computer vision*. pages 1521–1529, 2017. doi:10.1109/ICCV.2017.169.

Kendall, A. and Cipolla, R. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 5974–5983, 2017. doi:10.48550/arXiv.1704.00390.

Keselman, L., Iselin Woodfill, J., Grunnet-Jepsen, A., and Bhowmik, A. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. pages 1–10, 2017. doi:10.1109/CVPRW.2017.167.

Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. doi:10.48550/arXiv.1412.6980.

Klingenberg, K., Johannessen, K. V., Kaya, O., and Tingelstad, L. Industrial camera-aided trajectory planning for robotic welding on reflective surfaces. In *2024 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. pages 182–187, 2024. doi:10.1109/RCAR61438.2024.10671320.

Labbé, Y., Carpentier, J., Aubry, M., and Sivic, J. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, pages 574–591, 2020. doi:10.1007/978-3-030-58520-4_34.

Li, Y., Wang, G., Ji, X., Xiang, Y., and Fox, D. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. pages 683–698, 2018. doi:10.48550/arXiv.1804.00175.

Li, Z., Wang, G., and Ji, X. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*. pages 7678–7687, 2019. doi:10.1109/ICCV.2019.00777.

Litvak, Y., Biess, A., and Bar-Hillel, A. Learning pose estimation for high-precision robotic assembly using simulated depth images. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pages 3521–3527, 2019. doi:10.1109/ICRA.2019.8794226.

Lourenço, F. and Araujo, H. Intel realsense sr305, d415 and l515: Experimental evaluation and comparison of depth estimation. In *VISIGRAPP (4: VISAPP)*. pages 362–369, 2021. doi:10.5220/0010254203620369.

Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2. Ieee, pages 1150–1157, 1999. doi:10.1109/ICCV.1999.790410.

Marco-Rider, J., Cibicik, A., and Egeland, O. Polarization image laser line extraction methods for reflective metal surfaces. *IEEE Sensors Journal*, 2022. 22(18):18114–18129. doi:10.1109/JSEN.2022.3194258.

Njaastad, E. B. and Egeland, O. Automatic touch-up of welding paths using 3d vision. *IFAC-PapersOnLine*, 2016. 49(31):73–78. doi:10.1016/j.ifacol.2016.12.164.

Park, K., Patten, T., and Vincze, M. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*. pages 7668–7677, 2019. doi:10.1109/ICCV.2019.00776.

Polyhaven. Hdri: Forest trail. `https://polyhaven.com/`, 2024. Accessed: 2024-11-22.

Qin, W., Hu, Q., Zhuang, Z., Huang, H., Zhu, X., and Han, L. Ippe-pcr: a novel 6d pose estimation method based on point cloud repair for textureless and occluded industrial parts. *Journal of Intelligent Manufacturing*, 2023. 34(6):2797–2807. doi:10.1007/s10845-022-01965-6.

Rad, M. and Lepetit, V. Bb8: A scalable, accurate, robust to partial occlusion method for predicting

the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE international conference on computer vision*. pages 3828–3836, 2017. doi:10.1109/ICCV.2017.413.

Ren, G., Qu, X., and Chen, X. Performance evaluation and compensation method of trigger probes in measurement based on the abbé principle. *Sensors*, 2020. 20(8). doi:10.3390/s20082413.

Rothganger, F., Lazebnik, S., Schmid, C., and Ponce, J. 3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal of Computer Vision*, 2006. 66:231–259. doi:10.1007/s11263-005-3674-1.

Rusinkiewicz, S. and Levoy, M. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*. IEEE, pages 145–152, 2001. doi:10.1109/IM.2001.924423.

Schleth, G., Kuss, A., and Kraus, W. Workpiece localization methods for robotic welding-a review. In *ISR 2018; 50th International Symposium on Robotics*. VDE, pages 1–6, 2018.

Shen, J., Yoon, D., Shehu, D., and Chang, S.-Y. Spectral moving removal of non-isolated surface outlier clusters. *Computer-Aided Design*, 2009. 41(4):256–267. doi:10.1016/j.cad.2008.09.003.

Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., and Fitzgibbon, A. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 2930–2937, 2013. doi:10.1109/CVPR.2013.377.

Simonelli, A., Bulo, S. R., Porzi, L., López-Antequera, M., and Kontschieder, P. Disentangling monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pages 1991–1999, 2019. doi:10.48550/arXiv.1905.12365.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*. PMLR, pages 6105–6114, 2019. doi:10.48550/arXiv.1905.11946.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, pages 23–30, 2017. doi:10.1109/IROS.2017.8202133.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 1912–1920, 2015. doi:10.1109/CVPR.2015.7298801.

Yigit, C. B., Bayraktar, E., Kaya, O., and Boyraz, P. External force/torque estimation with only position sensors for antagonistic vsas. *IEEE Transactions on Robotics*, 2021. 37(2):675–682. doi:10.1109/TRO.2020.3031268.

Zhou, Q.-Y., Park, J., and Koltun, V. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. doi:10.48550/arXiv.1801.09847.

Zhou, Y., Barnes, C., Lu, J., Yang, J., and Li, H. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pages 5745–5753, 2019. doi:10.1109/CVPR.2019.00589.