

6D Assembly Pose Estimation by Point Cloud Registration for Robotic Manipulation

Kulunu Samarawickrama¹, Gaurang Sharma¹, Alexandre Angleraud¹ and Roel Pieters¹

Abstract—The demands on robotic manipulation skills to perform challenging tasks have drastically increased in recent times. To perform these tasks with dexterity, robots require perception tools to understand the scene and extract useful information that transforms to robot control inputs. To this end, recent research has introduced various object pose estimation and grasp pose detection methods that yield precise results. Assembly pose estimation is a secondary yet highly desirable skill in robotic assembling as it requires more detailed information on object placement as compared to bin picking and pick-and-place tasks. However, it has been often overlooked in research due to the complexity of integration in an agile framework. To address this issue, we propose an assembly pose estimation method with RGB-D input and 3D CAD models of the associated objects. The framework consists of semantic segmentation of the scene and registering point clouds of local surfaces against target point clouds derived from CAD models to estimate 6D poses. We show that our method can deliver sufficient accuracy for assembling object assemblies using evaluation metrics and demonstrations. The source code and dataset for the work can be found at: <https://github.com/KulunuOS/6DAPose>

I. INTRODUCTION

The ability to grasp an object successfully is a highly desired skill of robotic manipulators in applications such as industrial manufacturing, mining, space explorations, etc. The advancement of modern robot manipulators with multiple degrees of freedom, precise sensing and control capabilities highly encourages the development of the robot's cognition skills in order to execute dexterous tasks in above fields either autonomously or with human collaboration. Several research publications have demonstrated that vision-based deep learning and reinforcement learning can yield successful results in endowing cognitive skills such as object grasping [1], [2]. In that regard, the state of the art on robotic manipulation is heavily focused on object pose estimation and grasp pose detection [3], [4]. Pose estimation via point cloud based deep learning methods such as PVN3D [5] and FFB6D [6] have shown great accuracy in 6D pose estimation with fast inference times, which are ideal characteristics for robotic grasping applications. However, object pose alone is not sufficient to perform a successful grasp. The robot end-effector should reach a pose that produces a fail-proof grasping of the object, which is a different definition compared to the object pose. Grasp pose for an object can be sampled either analytically or using human expertise. Some analytical methods [7], [8], [9] have proposed deriving grasp pose by analyzing object model and depth information acquired from a sensor. Although these methods can produce feasible

¹Automation Technology and Mechanical Engineering, Tampere University, 33720, Tampere, Finland, firstname.surname@tuni.fi

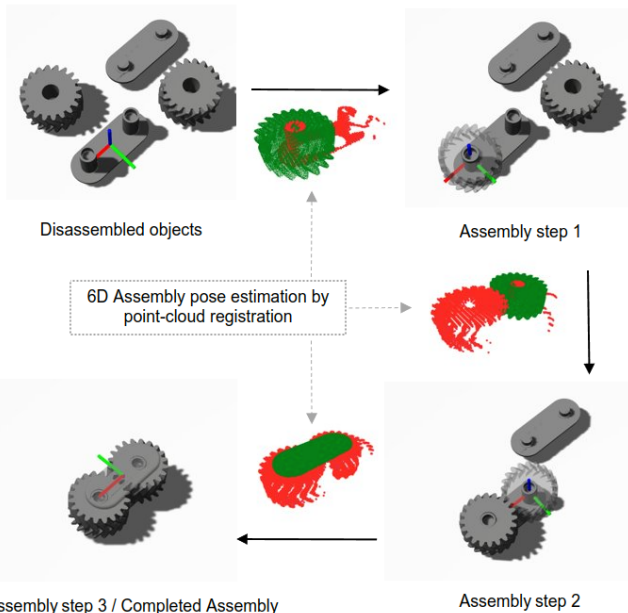


Fig. 1: Overview of assembly pose estimation pipeline depicting for each step the base assembly point cloud in red and the estimated assembly point cloud (object pose) in green.

grasp poses for a single object, they fail when objects are in clutter. GraspNet [10] introduced a model-agnostic method to directly produce grasp candidates on any observed unknown partial point cloud. This method produces multiple grasp candidates and prunes unfeasible candidates in an end-to-end pipeline which makes it highly suitable for bin picking, sorting and inspection tasks. High computational resource consumption of these methods are, however, an undesirable feature for robotic applications.

However, there exists a disparity in research for precise robotic assembly, as the majority of the existing studies on robotic assembly focus only on initial object pose estimation. Robotic assembly requires the additional estimation of object placement pose (see Fig. 1). We use the synonymous term 'assembly pose' henceforth in this paper. Assembly pose can be described as the pose of the end-effector that results in the successful placement of a grasped object that satisfies assembly constraints. The primary constraint that defines accuracy of assembly is the relative pose between objects of the assembly. Other assembly constraints vary depending on the nature of the task and the task environment, such as specific placement direction or motion, but is not considered in this work.

To address the issue of assembly pose estimation, we propose an approach that can be integrated into a robotic assembly framework without compromising the agility. As the final step of a robotic manipulation framework, an efficient method should well utilize the inferred information from object pose estimation and grasp pose detection steps, offer high accuracy, and not compromise the computational load and inference times. First, the proposed method includes a semantic segmentation module that estimates a per-pixel object label. Second, the segmentation results and RGB-D images are utilized to perform point cloud registration of a pair of source and target point clouds. As an object assembly may consist of several assembly steps, assembly poses are estimated as a sequential process, but separately for each individual assembly step. Pose estimation does therefore not rely on previous steps, besides the assumption of a correct previous assembly step, represented in the CAD model. The proposed method can be integrated with a framework consisting of existing pose estimation and grasp detection methods without additional model training.

The contributions of this work are as follow:

- 1) A novel approach for extending state of the art pose estimation datasets for assembly pose estimation
- 2) An effective method to generate source point clouds using CAD models
- 3) An iterative pipeline to estimate assembly poses for object assemblies with multiple objects
- 4) An evaluation of efficiency of point cloud registration as a method of assembly pose estimation

The remainder of this paper is organized as follows. In Section II we provide an overview of related work. Section III presents our proposed method, which is evaluated in Section IV and discussed in Section V. Section VI concludes the work.

II. RELATED WORK

An early attempt at combining image segmentation and point cloud registration for pose estimation and robotic grasping featured in the Amazon bin picking challenge in 2016 [11]. Zeng et al. takes an RGB-D input of the bin and outputs 6D object poses. The estimated object poses are converted to grasp poses to be grasped by a parallel gripper. They implemented a Fully connected Neural Network with VGG architecture for object segmentation in their method with a large dataset of around 130,000 images. Despite the fact that segmentation networks have vastly improved over time since 2016, their results show that the heavy occlusive and cluttered nature of bin picking tasks negatively affected point cloud registration, thereby increasing the pose estimation error. Besides, they constrain the source point cloud by orienting it towards the optical axis of the camera in the pose initializing step. The resulting ambiguity of rotation around the axis could result in an initial error that propagates through ICP refinement. Some later research proposed a solution to the problem by introducing model libraries containing samples of partial point clouds as seen from different view points for each unique object category.

SegICP [12] proposed another per-pixel instance segmentation module and an ICP registration method for the same purpose. They adopt Segnet [13], a convolutional neural network, for the semantic segmentation step in their pipeline. In the point cloud registration step they align a source point cloud against a target point cloud retrieved from a model library. Ultimately, they perform an exhaustive search for the best pose estimation by comparing the ICP result against all samples in the model library for each object category.

Some further studies have been done using SegICP as a base concept. Xu et al. [14] proposed a pose estimation pipeline with a custom segmentation network (FCN-artous-2s) and modified ICP algorithm for a grasp manipulation task. The modified ICP algorithm addresses lack of one-to-one correspondence points in partial point clouds of the target and source. Wang et al. [15] proposed a similar pose estimation method in their pipeline for a robotic spray painting application. They implement an additional network to estimate the initial orientation, which facilitates retrieval of the most accurate point cloud sample from the model library. However, the additional iterations and inference steps in these methods results in prolonged total inference times.

In contrast to above work, we perform real-time rendering of the target point cloud in the partial observable state without sampling views and generating model libraries. This eliminates naive assumptions in both pose initialization and exhaustive, repetitive ICP calculations. The evaluation metric used to measure registration accuracy directly affects the pose estimation accuracy. Zeng et al. [11] do not evaluate the registration result and SegICP uses a unique point correspondence matching criteria for evaluation of the registration result. We use two different evaluation metrics, namely inlier RMSE (Root Mean Square error) and a fitness score to analyze registration accuracy and reflect its propagation towards pose estimation accuracy in our work.

III. PROPOSED METHOD

We approach the object assembling task as an iterative process of several sub-assembling steps as described in Fig. 1. For the proposed method we assume that the order of assembling and pose of each object with respect to the base object are known. The base object can be defined as the completed assembly of objects at each step as described in Fig. 2. More complex and larger object assemblies can be divided into several sub-assemblies and corresponding base objects. Therefore it can be assumed that all objects assemblies can be assembled in above described manner.

An assembly pose estimation procedure that can be integrated to a generic vision based robotic assembling framework is presented in Fig. 2. The first step of the framework is to understand the scene and locate the objects. The second step is to produce a fail-proof grasp for each object. These two tasks can be achieved by object pose estimation and grasp pose estimation algorithms. Our proposed method applies in the third step where a successfully grasped object is assembled to satisfy assembly constrains. The assembly pose estimation algorithm is described in Alg. 1.

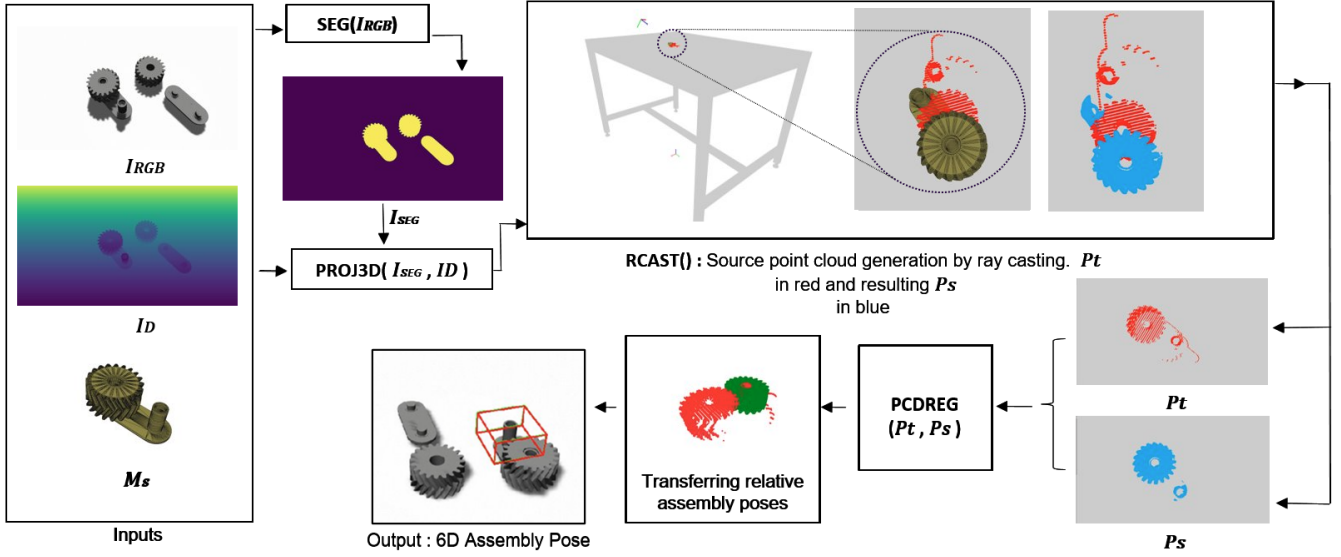


Fig. 2: Proposed framework for 6D assembly pose estimation

Algorithm 1: Assembly pose estimation

Parameters:

Input : $I_{RGB}, I_D, M_{s=1,2,3..n}$

Output : T_w^a

Functions : $SEG()$; Instance Semantic Segmentation
 $PROJ3D()$; 3D projection of I_D
 $TFORM()$; $SE(3)$ Transformation
 $RCAST()$; Raycasting in simulation
 $PCDREG()$; Pointcloud Registration

foreach *Assembly step* **do**

$I_{SEG} \leftarrow SEG(I_{RGB})$

$P_t \leftarrow PROJ3D(I_{SEG}, I_D)$

$P_s \leftarrow RCAST(TFORM(M_s, P_t))$

$T_b^w \leftarrow PCDREG(P_t, P_s)$

$T_w^a = T_b^a T_b^w$

end

the assembly object frame with respect to the robot base frame. An assembly pose must satisfy certain assembly constraints to create a precise assembly which varies depending on the task. Often in industrial object assemblies, the assembly pose is constrained by the relative pose of assembly object with respect to a base object of an assembly or an assembly surface.

In Alg. 1, each base object is represented by a source mesh M_s . A prior knowledge T_b^a defines the relative pose from base object b to assembly object a by the homogeneous transformation matrix T_b^a . The output of the algorithm is the homogeneous transformation matrix T_w^a which estimates the relative assembly pose of assembly object to world frame w .

Instance semantic segmentation: The first function in algorithm $SEG()$, an instance semantic segmentation module trained on RGB images I_{RGB} estimates per-pixel assembly object label for a given scene. A segmentation mask of the base object I_{SEG} in the assembly scene can be extracted from the segmentation result.

Target point cloud projection: In the second function $PROJ3D()$, we project a target point cloud of base object P_t using I_{SEG} and depth image I_D of the scene.

Source point cloud projection: A source point cloud P_s with enough correspondences and an initial transformation closer to the P_t is an essential for successful point cloud registration. To generate a P_s with above qualities, we transform the relevant M_s to the center of P_t in $TFORM()$ and implement a ray casting function $RCAST()$ in simulation. The $RCAST()$ function simulates a pinhole camera looking at center of M_s with the intrinsic and extrinsic parameters of the camera used to obtain I_{RGB} and I_D in real-time. This function generates a P_s that has a similar partial view of the target and sufficient correspondences to P_t . This procedure eliminates the requirement to have a separate database of point clouds for registration compared to other related work [12], [13]. we record the initial transformation of P_s as T_w^s .

A. Object and Grasp Pose Detection

Object pose locates the object in robot coordinate frame. Grasp pose defines the pose of robot's end effector while producing a fail-proof grasp of the object. A successful grasp is a prerequisite for a successful assembly and depends on both above factors. However, object grasping is not the main focus of our work. The pipeline provides the opportunity to utilize a grasp pose detection method of preference. A module that first estimates the object pose and then derives grasp poses relative to object pose is more suitable for the task. Alternatively, point cloud registration result can be used to detect object pose as well when relative grasp pose information is available as in our demonstration.

B. Assembly Pose Estimation

For a robotic assembling task, assembly pose estimation can be interpreted as the estimation of rotation and translation of

Point cloud registration: In the fourth step, we implement a point cloud registration pipeline to align P_s against P_t . We estimate an initial transformation using a global registration based on Random Sample Consensus (RANSAC). RANSAC detects corresponding points in P_s and P_t using a nearest neighbour algorithm with Fast Point Feature Histograms (FPFH) [16] descriptors as geometrical feature inputs. Furthermore, the algorithm runs for 100,000 iterations while pruning correspondences based on edge length E between any pair of corresponding points in P_s and P_t :

$$\begin{aligned} \|E_s\| &> 0.09\|E_t\|, \\ \|E_t\| &> 0.09\|E_s\|, \end{aligned}$$

and a point cloud distance threshold of 0.036. These values were figured out experimentally for each assembly.

A local registration based on point-to-plane ICP [17] further refines the global registration result. The refinement is based on the convergence of an objective function $L(T)$:

$$L(T) = \sum_{p \in P_s, q \in P_t} ((p - Tq) \cdot (n_p))^2, \quad (1)$$

where p and q are points in P_s and P_t respectively, n_p is an estimate of normal of point p . The function converges when the point clouds are aligned. The resulting 6D transformation from P_s to P_t in our context is equal to T_s^b the estimated 6D transformation of base object relative to P_s .

Local Pose transformation: Knowing the transformation of P_s as T_w^s , the estimated transformation of base object with respect to world frame T_w^b can be derived. In the final step, local assembly poses T_b^a can be transferred to estimate the 6D assembly pose of each object with respect to robot base T_w^a , as the output of algorithm:

$$T_w^b = T_s^b T_w^s, \quad (2)$$

$$T_w^a = T_b^a T_w^b. \quad (3)$$

IV. EVALUATION

A. Implementation

It is common to demonstrate the performance of pose estimation algorithms using existing standard datasets. However, there are no available standard datasets specifically designed for the purpose of assembly pose estimation. We therefore evaluate our method on two different object assemblies using the 3D mesh files of the assembly objects obtained from Thingiverse^{1,2}. We generate two synthetic datasets in a format specific for assembly pose estimation in gazebo simulation environment. Table I and II describe the helical and planetary gear assembly sets, with three assembly steps four assembly steps, respectively. For ease of integration the proposed format is designed as an extension to the existing BOP format [18] for object detection and 6D pose estimation. The data generation algorithm is presented in Alg. 2 and the source code and links to download the datasets can be found in the git repository: <https://github.com/KulunuuOS/6DAPose>.

¹<https://www.thingiverse.com/thing:3936460>

²<https://www.thingiverse.com/thing:8460>

Algorithm 2: Assembly dataset generation

Parameters:

ϕ : yaw angle of the camera

θ : pitch angle of the camera

s : scale of the camera

Input : CAD models of object assembly

Output : I_{RGB}, I_D ; color and depth images

I_{SEG} ; segmentation maps,

P_{obj} ; ground truth object poses,

P_{cam} ; ground truth camera pose,

K_{cam} ; ground truth camera parameters

Define and record assembly constrains

foreach *Assembly step* **do**

foreach ϕ, θ, s **do**

 | Record $\{I_{RGB}, I_D, I_s, P_{obj}, P_{cam}, K_{cam}\}$

end

end

For our implementation we only consider primary assembly constraints, defined by the relative transformation between the base object and the assembly object T_a^b . This has to be individually analyzed with human expertise for distinct assemblies. The helical gear and planetary gear assemblies are constrained by three and four sets of relative poses, respectively, corresponding to their assembly steps (see Fig. 4 and 6). In the process, assembling order of objects, the base object for each assembling step and object information such as mesh model diameter and corners have to be analyzed. Data is recorded for each assembly step starting from the disassembled configuration. For each assembly step, a simulated RGB-D camera captures the data by following hemisphere sampling procedure [19] parameterised by yaw angle ϕ , pitch θ and scale s . For the implementation, we simulate a Intel Realsense camera model with default camera parameters and capture 431 instances for each assembly step in both object assemblies. The major difference between our dataset and standard object pose estimation datasets is the inclusion of assembly steps as individual sub datasets. Furthermore, its important to note that contrasting to pose estimation datasets, the ground truth assembly pose for i^{th} assembly step is obtained from $i + 1^{th}$ assembly step. We share tools from Open3D [20] library to implement point cloud processing functions in our work.

B. Metrics

The pose estimation accuracy is directly affected by the point cloud alignment produced by the ICP registration process. Therefore for each of 431 instances in each assembly step we calculate the *Fitness* and root mean square error of inliers; l_{rmse} produced by point cloud registration. *Fitness* refers to the ratio between number of total inliers l and total points N in P_t . l_{rmse} is a function that calculates the error between inliers in source l_s and target l_t . For an ideal point cloud alignment *Fitness* should be closer to 1 and l_{rmse} must be closer to 0:

$$Fitness = \frac{l}{N}, \quad (4)$$

TABLE I: Helical gear assembly dataset, with four parts and three assembly steps.

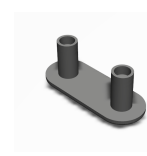







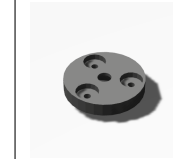
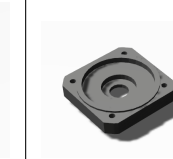
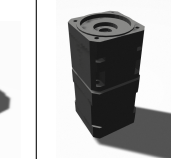
Mesh model					
Mesh name	Bottom casing	Left gear	Right gear	Top casing	Complete assembly
Diameter [cm]	3.81	2.78	2.78	3.82	

TABLE II: Planetary gear assembly dataset, with five major parts and four assembly steps.

Mesh model						
Mesh name	Nema17 Motor	Sun gear	Housing	Carrier	Cover	Complete assembly
Diameter [cm]	7.67	2.70	6.17	3.56	5.38	

$$l_{rmse} = \sqrt{\frac{1}{l} \sum_{j=1}^l \|l_{s_j} - l_{t_j}\|^2}. \quad (5)$$

The symmetric nature of objects deprecates rotation and translation error as an evaluation metric for pose accuracy. Hence, we use Maximum Symmetry-aware Surface Distance *MSSD* [18] and Average Distance of model points for objects with Indistinguishable views *ADI* [21] as pose error functions to evaluate the accuracy:

$$MSSD = \min_{y \in Y} \max_{x \in X} \|Tx - \hat{T}yx\|_2, \quad (6)$$

$$ADI = \frac{1}{K} \sum_{k=1}^K \min_{m=1}^M \|U_k - V_m\|. \quad (7)$$

ADI calculates average of distances to the nearest neighbors from vertices in the ground-truth pose to vertices in the estimated pose as described in Eqn. 7. U_k and V_m are 3D vectors describing the respective vertices where K and M are total number of sampled vertices. In contrast *MSSD* is less dependent on sampling vertices and calculates maximum distance between surfaces as in Eqn. 7 where Y is a set of global symmetry transformations, X is a set of mesh vertices, T and \hat{T} are ground truth and estimated 6D poses respectively. Hence *MSSD* is more suitable metric for robotic manipulation tasks. The errors are calculated individually for each assembly step and does not represent a cumulative error propagation. The mean values and standard deviation for each metric is calculated for each assembly step in the whole dataset. The calculations were run in a standard computing machine with an AMD Ryzen 7 4800h CPU with cores without multi-threading. The mean time per iteration is calculated in seconds. The evaluation results of the method for the two assembly datasets are summarized in the Tables III and IV.

C. Results

We present evaluation results for two simulated gear assembly datasets illustrated in Tables III and IV. The step-wise propagation of point cloud registration of base objects between target (red) and source (blue) point clouds are illustrated in Fig. 3 and 5, demonstrating well-matching overlap in all cases. The estimated (green) and ground truth (red) 6D assembly poses for each assembly steps are displayed using bounding box representation in Fig. 4 and 6. A statistical analysis of all estimated assembly 6D poses are summarized in Tables III and IV. The analysis shows that when it is possible to achieve good *Fitness* values closer to 1 with l_{rmse} closer to 0, the assembly poses can be estimated with a high accuracy according to the pose metrics. Values closer to 0 in *MSSD* explains that surfaces align well when an assembly mesh is rendered for an estimated pose, as compared to the ground truth pose. Similarly in *ADI*, values closer to 0 suggests that distances between vertices of an assembly mesh in estimated pose are closer to that of the ground truth result.

One exception is the assembly step 4 for the planetary gear dataset, which has a comparatively bigger pose error even with consistent *Fitness* values. The reason for this is the occlusions forced on assembly step 4, which is located inside assembly object 3. An absence of important assembly surfaces introduces a pose estimation error even when base object point clouds fit well.

A second observation is that the time for pose estimation increases with the increase of the assembly steps. This is due to an increase in the number of points in the source point cloud, hence increasing the computational load of the point cloud registration approach. In our cases, we sample 30,000 points for each assembly object in the dataset.

To observe the sim-to-real gap, we also evaluated our method on an industrial use case, for the estimation of eight assembly poses of rocker arms on a Diesel engine (see Fig. 7). The base object for this application is a region of interest (ROI) on the assembly surface. Henceforth we use the term

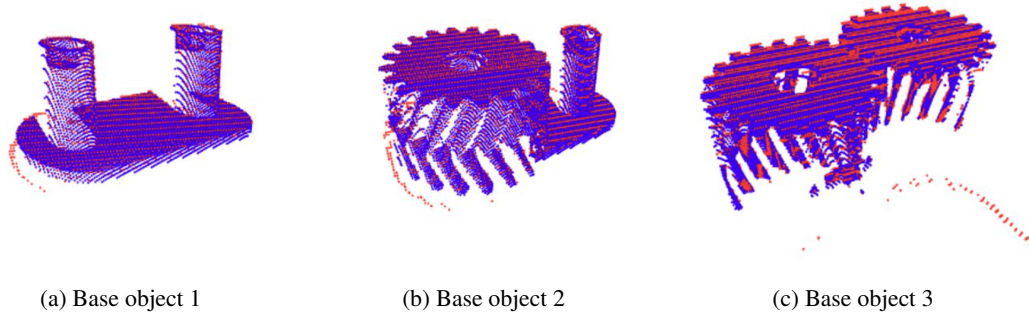


Fig. 3: Point cloud registration of base objects for each assembly step: bottom casing (a), bottom casing and left gear (b), bottom casing, left and right gear (c). Target (red) and source (blue) point clouds are obtained from camera and CAD, respectively.

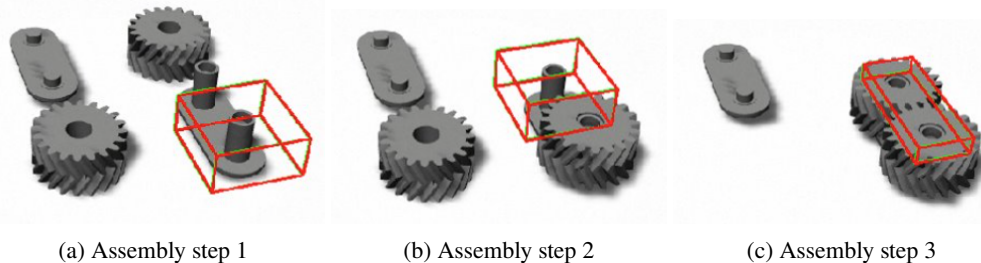


Fig. 4: Estimated (green bounding box) and ground truth (red bounding box) 6D assembly poses for assembly steps 1-3 in helical gear assembly: left gear to bottom casing (a), right gear to bottom casing (b), top casing to complete assembly (c).

TABLE III: Evaluation metrics for 6D assembly pose estimation on the helical gear dataset

Assembly step	<i>Fitness</i>		<i>lrmse</i>		<i>ADI</i>		<i>MSSD</i>		time [sec]
	mean	stdv	mean	stdv	mean	stdv	mean	stdv	
1	0.995184	0.062205	0.000429	0.000097	0.000604	0.004411	0.001075	0.005385	0.505338
2	0.999543	0.000571	0.000436	0.000073	0.000374	0.000036	0.000741	0.000156	0.562895
3	0.999333	0.000693	0.000444	0.000085	0.000510	0.000057	0.000778	0.000425	0.646653

base ROI to describe it. A semantic segmentation module was trained on Detectron2 [22] Mask-RCNN architecture to detect base ROIs as shown in Fig. 7b. The module was trained on 40 captured images and manually labelled annotations. The estimated base ROI and bounding box for estimated pose are illustrated in similar colors. The results emphasize that the method can produce feasible assembly poses for different regions of the surface while having less consistency compared to simulated data due to occlusions from the engine surface. Unlike in a simulated environment, ground truth information is unavailable for real life applications. Therefore, the accuracy of the estimated assembly pose has to be visually inspected. Due to absence of precise CAD models of the engine assembly we use an approximated CAD model to generate base ROI as explained in Fig. 7a. This adversely affects the point cloud registration by reducing point correspondences. However, the method generates feasible assembly poses when there are sufficient correspondences. The number of correspondences can be improved by changing the angle of the camera to capture as many points on the ROI as possible. This can be enabled by changing the viewing angle of the camera as

placed on the end-effector of the robot. As visible in Fig. 7b, the bounding boxes align well on ROI surfaces with more correspondences (red, blue, yellow) and weaker alignments on incomplete ROI segments with less correspondences (cyan, purple). The observations suggest that surface regions located at the center and corners of the field of view tend to capture less ROI surfaces compared to other regions. The estimated poses are sufficient for assembling, considering the magnetic force between rocker arm and surface in the application.

V. DISCUSSION

The proposed framework is capable of estimating a 6D assembly pose for an object assembly without initializing source point clouds at a predefined constraint, unlike Zeng et al. [11] or other work that require creating a model library [12]. Furthermore, calculating two separate metrics for point cloud registration step and pose estimation provides a feedback on the quality of captured depth information, pose initialization and point cloud registration parameters. Unlike work that only estimates object poses [14], [15], which is more suited for bin picking and sorting tasks, the proposed framework can put together an object assembly while maintaining quality control.

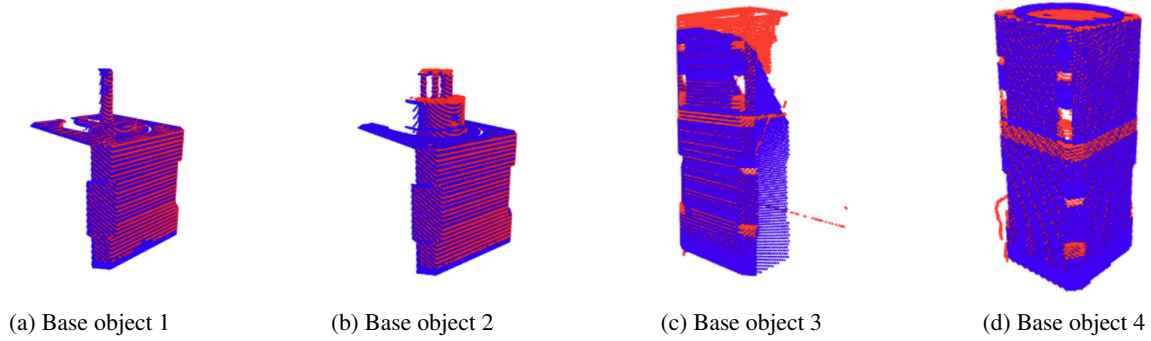


Fig. 5: Point cloud registration of base objects for each assembly step: motor (a), motor and sun gear (b), motor, sun gear and housing (c), motor, sun gear, housing and cover (d). Target (red) and source (blue) point clouds are obtained from camera and CAD, respectively.

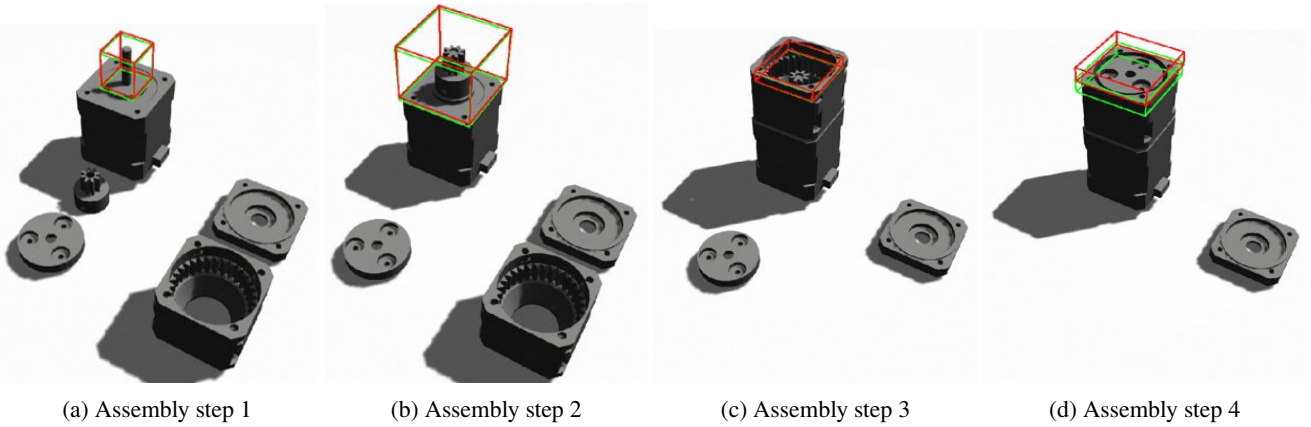


Fig. 6: Estimated (green bounding box) and ground truth (red bounding box) 6D assembly poses for assembly steps 1-4 in planetary gear assembly: sun gear to motor (a), housing to motor (b), carrier to housing (c), cover to complete assembly (d).

TABLE IV: Evaluation metrics for 6D assembly pose estimation on the planetary gear dataset

Assembly step	<i>Fitness</i>		<i>lrmse</i>		<i>ADI</i>		<i>MSSD</i>		time [sec]
	mean	stdv	mean	stdv	mean	stdv	mean	stdv	
1	0.999880	0.000417	0.000548	0.000109	0.000528	0.000045	0.001425	0.000087	1.329737
2	0.990592	0.096101	0.000618	0.000122	0.002384	0.019576	0.003604	0.023681	1.359196
3	0.999851	0.000489	0.000534	0.000146	0.000427	0.000619	0.000796	0.000964	2.369656
4	0.997401	0.048216	0.001335	0.000199	0.003576	0.008271	0.006678	0.009545	2.396376

Although the proposed method estimates assembly poses accurately for object assemblies with few objects, the computational load increases for complex assemblies with multiple objects, due to large number of points accumulated in the registration step. Similarly, the absence of precise CAD models decreases the number of correspondences in derived source point clouds which adversely affects point cloud registration. Furthermore, as a purely geometrical approach, the method is not robust to occlusions and obstructions in the scene. Occlusions and obstructions eliminate points, which represent important features and affect correct point cloud alignments. An example is when assembling or placing an object internally on a place surrounded by walls or surfaces such as plumbing, box packaging etc. In such cases, although a point cloud registration achieves a good fitness score, the resulting assembly pose may be less accurate.

As a future improvement of this work it would be appropriate to have a learning-based pose estimation module before the point cloud registration step. A key point-based approach would reduce the computational load by ruling out the necessity to deal with a complete point cloud with large number of points. It will make it possible to use the framework on complex object assemblies with a large number of objects without compromising the inference time. Estimated assembly poses can also be used to prune unfeasible object grasp candidates in grasp detection networks. Moreover, complex robot manipulation tasks that involve additional constraints will be considered, as the next step of assembly pose estimation. This includes specific placement direction or motions, as found in insertion tasks, such as peg-in-hole or clamping.

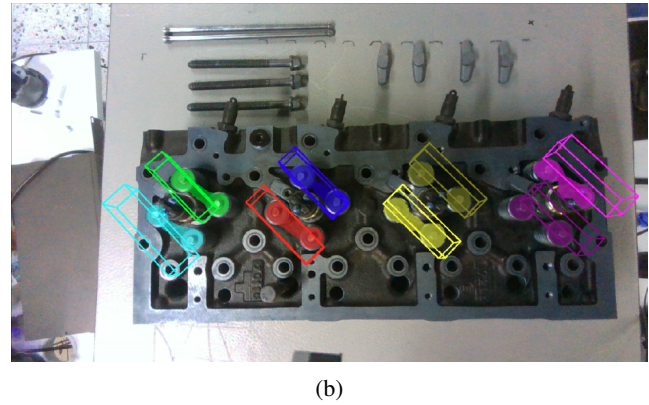
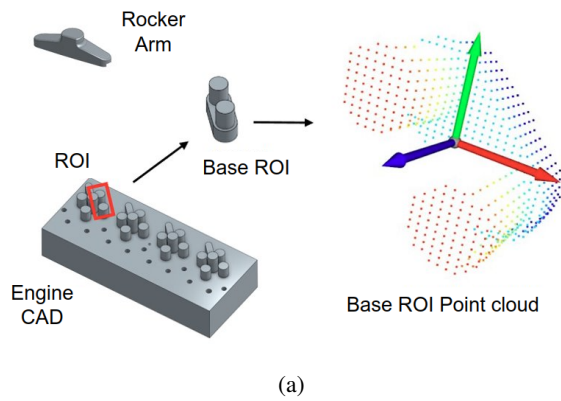


Fig. 7: Diesel engine assembly pose estimation results for eight rocker arms. Point cloud registration utilizes base regions of interest (ROI) on the assembly surface (a), obtained from semantic segmentation (b). The eight estimated assembly poses are highlighted with 3D bounding boxes (b).

VI. CONCLUSION

This work proposed an assembly pose estimation method for robot manipulation and assembly tasks, utilizing semantic segmentation and point cloud registration between target and source point clouds. The work also presented a synthetic data generation pipeline as an improvement to existing object pose estimation datasets. Our approach is evaluated with suitable metrics on two simulated gear assembly datasets, which indicates that point cloud registration is well capable of estimating 6D assembly poses for object assemblies. In addition, we demonstrate the approach on a real industrial assembly task, i.e., Diesel engine assembly, which verifies that feasible assembly poses can be estimated for real applications.

ACKNOWLEDGEMENTS

Project funding was received from Helsinki Institute of Physics' Technology Programme (project; ROBOT) and European Union's Horizon 2020 research and innovation programme, grant agreement no. 871252 (METRICS).

REFERENCES

- [1] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robotics Reports*, pp. 1–11, 2020.
- [2] X. Liang and H. Cheng, "RGB-D camera based 3D object pose estimation and grasping," in *IEEE Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2019, pp. 1279–1284.
- [3] M. Q. Mohammed, K. L. Chung, and C. S. Chyi, "Review of deep reinforcement learning-based object grasping: Techniques, open challenges, and recommendations," *IEEE Access*, vol. 8, pp. 178 450–178 481, 2020.
- [4] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review," *Artificial Intelligence Review*, vol. 54, pp. 1677–1734, 3 2021.
- [5] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "FFB6D: A full flow bidirectional fusion network for 6D pose estimation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [6] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3D keypoints voting network for 6DoF pose estimation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [7] K. Kleeberger, F. Roth, R. Bormann, and M. F. Huber, "Automatic grasp pose generation for parallel jaw grippers," in *International Conference on Intelligent Autonomous Systems*, 2021, pp. 594–607.
- [8] H. Xu, Y. Sun, Q. Sun, M. Yang, J. Chen, B. Qiang, and J. Wang, "3D grasp pose generation from 2D anchors and local surface," in *ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, 2023, pp. 1–7.
- [9] B. S. Zapata-Impata, P. Gil, J. Pomares, and F. Torres, "Fast geometry-based computation of grasping points on three-dimensional point clouds," *International Journal of Advanced Robotic Systems*, vol. 16, no. 1, 2019.
- [10] A. Mousavian, C. Eppner, and D. Fox, "6-DOF graspnet: Variational grasp generation for object manipulation," in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2901–2910.
- [11] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge," in *IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 1386–1383.
- [12] J. Wong, V. Kee, T. Le, S. Wagner, G.-L. Mariottini, A. Schneider, L. Hamilton, R. Chipalkatty, M. Hebert, D. Johnson, J. Wu, B. Zhou, and A. Torralba, "SegICP: Integrated deep semantic segmentation and pose estimation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5784–5789.
- [13] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [14] H. Xu, G. Chen, Z. Wang, L. Sun, and F. Su, "RGB-D-based pose estimation of workpieces with semantic segmentation and point cloud registration," *Sensors*, vol. 19, no. 8, p. 1873, 2019.
- [15] Z. Wang, J. Fan, F. Jing, Z. Liu, and M. Tan, "A pose estimation system based on deep neural network and ICP registration for robotic spray painting application," *The International Journal of Advanced Manufacturing Technology*, vol. 104, pp. 285–299, 2019.
- [16] R. B. Rusu, "Semantic 3D object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, pp. 345–348, 2010.
- [17] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and vision computing*, vol. 10, no. 3, pp. 145–155, 1992.
- [18] T. Hodaň et al., "BOP: Benchmark for 6D object pose estimation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 19–34.
- [19] S. Ahmad, K. Samarawickrama, E. Rahtu, and R. Pieters, "Automatic dataset generation from cad for vision-based grasping," in *20th International Conference on Advanced Robotics (ICAR)*, 2021, pp. 715–721.
- [20] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.
- [21] S. Hinterstoisser et al., "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Asian Conference on Computer Vision (ACCV)*, 2012, pp. 548–562.
- [22] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," <https://github.com/facebookresearch/detectron2>, 2019.