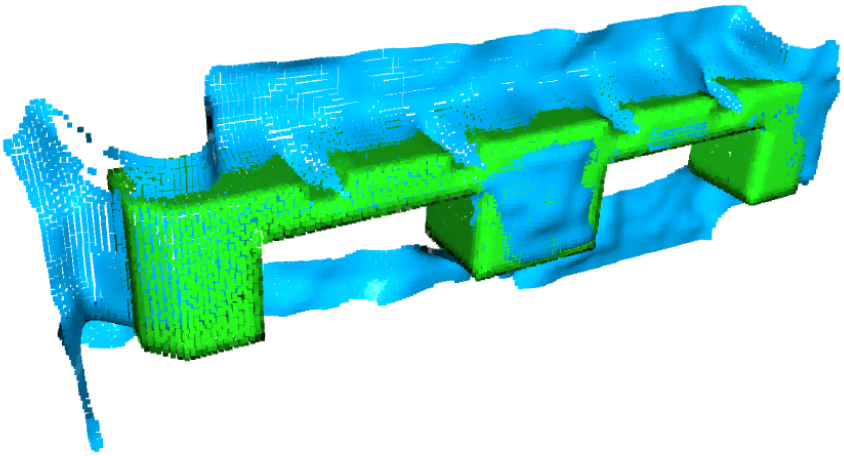


Exploring Load Localization for Automated Guided Vehicles

Pallet pose estimation using a depth sensing camera

Master's thesis in Systems, Control and Mechatronics

JONATAN NORDSTRÖM

DEPARTMENT Electrical engineering

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Exploring Load Localization for Automated Guided Vehicles

Pallet pose estimation using a depth sensing camera

Jonatan Nordström

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical engineering
Division of System and control
Chalmers University of Technology
Gothenburg, Sweden 2026

Exploring Load Localization for Automated Guided Vehicles
Pallet pose estimation using a depth sensing camera
JONATAN NORDSTRÖM

Supervisor: Petter Falkman, Depatment of Electrical Engineering
Examiner: Petter Falkman, Depatment of Electrical Engineering

Master's Thesis 2026
Department of Electrical Engineering
Division of Automation
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 734 161606

Cover: Pallet template point cloud (green) superimposed on depth cloud scene of a pallet in a rack (blue).

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Exploring Load Localization for Automated Guided Vehicles
Pallet pose estimation using a depth sensing camera
Jonatan Nordström
Department of Electrical engineering
Chalmers University of Technology

Abstract

Automated guided vehicles (AGVs) typically require pallets to be placed in precise locations to be able to pick them without advanced detection systems. This limits AGV systems in environments where pallets are also handled by human fork lift operators. Modern solutions to this problem often rely on machine learning, which requires powerful and costly computers inside the AGVs.

In this thesis a computationally efficient method for estimating the pose of a pallet using a depth sensing camera is proposed and evaluated. The method uses a novel approach to find the region of interest directly in the depth map by segmenting it into depth slices and detecting the fork pockets of the pallet with fast 2D image processing techniques. A rough pose estimate is calculated from the detected fork pockets, after which only a small region around the pallet is deprojected into a point cloud where the pose is refined using the iterative closest point (ICP) algorithm.

The method was evaluated on 2678 depth maps of EU-pallets captured on the floor and in a rack at angles up to $\pm 15°$. A pallet was detected in 90.14% of the frames and 73.61% of the frames resulted in an accepted pose estimation. The standard deviation of the estimated vertical position of a stationary pallet was 4.59 mm in the best case, indicating a high repeatability. The complete pipeline was 30.32% faster than performing ICP on the full point cloud, showing that the proposed approach reduces the computational load while maintaining an accurate localization.

Keywords: AGV, load, localization, pallet, point cloud, pose estimation, depth map.

Acknowledgements

I would like to express my gratitude to Nils Tornberg, who acted as supervisor on behalf of the case company and provided valuable guidance throughout the project. I would also like to thank Christian Skeppstedt for making it possible to carry out this thesis as part of my employment at the case company.

Finally, a special thank you goes to my partner Julia Brunke for her continuous support and encouragement during this work.

Jonatan Nordström, Gothenburg, June 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AGV	Automated Guided Vehicle
FOV	Field Of View
ICP	Iterative Closest Point
PCD	Point Cloud Data
ROI	Region of Interest
TOF	Time of Flight
DOF	Degrees of Freedom

Contents

List of Figures

List of Tables

1

Introduction

Automated guided vehicles (AGVs) are an increasingly common sight in modern warehouses, distribution centers and factories. The case company for this study has developed and installed complete AGV systems for 40 years. The current AGVs at the case company rely on the assumption that the pallets are both picked up and dropped off by the AGVs. This allows for very robust operations as the AGVs always place the pallets in a very precise location and can easily pick the pallets later without advanced detection systems. This limits the AGV system to either operating without human fork lift operators present, or requiring guiding structures for the pallets. With load localization the AGVs could pick pallets placed by human operators without guiding structures.

Pallet localization has been widely studied using varying sensing methods. 2D lasers, camera vision and most recently depth sensing cameras. More recent works use machine learning to either detect an approximate location or perform the full localization [1], [2]. These approaches typically use RGB images, point clouds or a combination of both, and can achieve a high accuracy even when occluded [3]. This comes at the cost of increased computational complexity driving a need for more powerful and costly computers inside the AGVs. With the rising cost of computational components, a key challenge to increase the computational efficiency has emerged in real life settings.

Several studies have explored strategies to reduce the computational load, such as Region Of Interest (ROI) filtering and simplifying feature extractions [1], [4]. Notably, the detection of the ROI itself is often performed with machine learning or by processing the full point cloud, meaning that a significant computational load remains in the first step of the pipeline.

In this paper, a novel approach to finding the ROI of the pallet directly in the depth map is proposed. Instead of directly converting the entire depth map to a point cloud, the proposed approach segments the depth map into slices and performs fast image space processing on these slices to find a rough estimate of the pallets position before converting to a point cloud. By reducing the amount of data being processed early the method aims to achieve faster execution while maintaining accurate localization.

1.1 Aim

The aim of this thesis is to develop and evaluate a method for estimating the pose of a pallet using a depth sensing camera, such that an AGV can pick pallets placed by human operators. The method should be computationally efficient enough to run without the need for GPUs or expensive hardware, while maintaining an accuracy sufficient for the AGV to safely pick the pallet.

1.2 Limitations

This thesis is limited to the localization of EU-pallets with the front face visible to the camera. The pitch and roll of the pallet are assumed to be zero as the pallet is placed on a flat surface, and the yaw is limited to $\pm 15^\circ$ relative to the camera. A single depth sensing camera is used, and the method is evaluated on recorded data rather than integrated on an AGV.

2
Theory

The raw data from most depth sensing cameras are a depth map. A depth map is an image where each pixel represent a depth at that pixels location. This data can be considered 2.5D (something between 2D and 3D) as the Z-axis is properly represented while the x- and y-axis are not. To get the full 3D representation of the data one can de-project the depth map into point cloud data (PCD) using the cameras intrinsic matrix K. Where f_x and f_y are the focal lengths of the camera in the x and y direction in pixels. And c_x and c_y are the coordinates for the principal point of the camera.

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

The point represented by a pixel u, v can be found by the equation below where $d(u, v)$ is the depth for that pixel as given by the depth map.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = d(u, v)\, K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2.2}$$

Working directly with the PCD has been done before by e.g. Molter & Fottner (2018) where they localize pallets in real-time using an overhead Time-of-Flight (TOF) depth camera on a forklift [5]. They segment the three vertical legs of a pallet and use the know geometry of a pallet estimate the pose. The method was found to be able to accurately detect a pallet at a 90° angle from the camera. This paper demonstrates that an accurate pose estimation can be made with PCD, however at a significant performance cost as the authors employ CUDA processing to achieve real time processing.

To reduce the computations needed a common first step is often to find a ROI. This is a region in the data where the desired object is expected to be found. Tsiogas et al. (2021) used a two stage pipeline to separate the detection of the ROI and the pose estimation [1]. They used a neural network to extract the pallet RoI in the camera images, and then used model based pattern matching on a point cloud extract of the ROI. This method achieves 95% accurate pose estimation for pallets with a wide difference in rotation relative to the sensor. As the method uses machine learning in the first step, a computer with a CUDA processor is required to compute this in real time.

In this paper a few limits are placed on the pose of the pallets that are to be detected. It is assumed that the pitch and roll are zero as the pallet is placed on a flat surface, and the yaw is limited to 15°. As such an approximate frontal view can be assumed and traditional 2D imaging techniques can be used. Suzuki & Abe (1985) used a border following algorithm on a binary image to detect contours [6]. A binary image is an image with only 1- and 0-pixels. This algorithm is both fast and efficient at detecting contours in an image. These contours can be used to find the ROI and a very rough pose estimate.

Shao et al. (2023) use a feature based method to find the rough estimate [7]. After which they use point cloud registration with a iterative closest point (ICP) method to align a template to the pallet. This step matches the features of a template point cloud of a pallet to the point cloud of the scene and give an accurate estimate of the 6DOF pose of the pallet. The ICP algorithm is a registration algorithm that was introduced by Besl and McKay [8] and Zhang [9] for different purposes. Besl and McKays intended use was to fit a shape to a scene where both are point clouds. This was done by minimizing the difference between the two point clouds. This application fits exactly with what this paper attempts to do. The ICP algorithm relies on a good rough estimate so the corresponding points between the clouds can be found. This rough estimate will be calculated in the depth map before converting to a point cloud.

More recent studies such as Kai et al. (2025) propose a solutions that uses RGB data instead of depth data [10]. In their paper they use multiple neural networks to detect the pose of the pallet. The networks are trained on the front face of pallets with the goal to learn the 3D structure of the pallet. The paper highlights and attempts to rectify one of the fundamental problems with neural networks in this field. The networks can only detect the type of pallet it is trained on. By teaching the network the general shape of the pallet, the paper attempts to mitigate this issue. The paper tests different load similarities with their network with good accuracy. However the same pallet is used with different loads further highlighting the core issue with neural networks.

3
Method

The work was carried out in an iterative manner. Relevant literature was reviewed and a part of the method was planned and implemented, after which it was tested and evaluated before the next iteration was planned. The method was implemented in Python using OpenCV [11] for the depth map processing and Open3D [12] for the point cloud processing. Testing during development was done both live in the case company's workshop and on saved frames with known pallets in them, which allowed changes to be evaluated against the same data between iterations. This chapter describes the selected camera hardware followed by each step of the proposed method.

3.1 Camera hardware

When selecting the camera several factors were considered. The main factors for the case company were availability and cost as these would affect future implementation in production. Another important factor was the Field of view (FOV) which needed to be enough to capture the full pallet, however a larger FOV would impact accuracy and performance. The type of technology used such as stereoscopic vision, time of flight (TOF) or structured light have different strengths and weaknesses that were considered. Resolution, depth accuracy, IP ratings and range were also collected for several cameras and with the previously mentioned factors discussed with the case company.

The model chosen was the Intel RealSense D415 due to its availability, cost and prevalent use in the industry. The specific model was chosen due to its lower FOV which gives a greater pixel density compared to other models in the same series as they all have approximately the same resolution. The RealSense cameras use stereoscopic depth vision which has the main downside of having a higher computational load than the other methods. However this computation is made in the RealSense camera and is therefore not an issue in this application. The main benefit of this type of camera is the low risk of interference due to reflections or outside light sources. This makes it suitable for many types of environments.

3.2 Proposed method

The core of this paper are depth maps and point cloud data (PCD). The raw data captured by the depth camera is in the form of a depth map. A depth map is

an image with depth instead of color values and can be considered 2.5D data with depth information but no concrete x, y information. This data can be converted to PCD with the use of the cameras intrinsic matrix as shown in equation 2.1. In this paper the depth map will be used to gain a rough estimate of the pallets pose. And the PCD will be used together with the rough estimate to gain an accurate estimate.

The benefits of a depth map compared to a point cloud is that the data is structured and compact, while a point cloud is unstructured and often requires computations over the complete point cloud for any computation. While a point cloud is a full 3 dimensional data structure allowing for shape fitting in all dimensions at the cost of increased computation. This allows for fast computation when looking at a smaller region in a depth map as desired data is easily accessible without iterating over the complete dataset.

This paper proposes a solution where an approximate location of the pallet is detected in the depth map using fast 2D imaging techniques, before deprojecting only a small region around the pallet to a point cloud. After which an accurate estimation of the pallet can be found using the smaller scale PCD using heavier algorithms. The goal of which is to lessen the computational load without losing accuracy. The proposed method will follow the steps shown in flowchart 3.1. The flowchart is separated into the depth map processes and the point cloud processes. The depth map process contain 2 main sub-processes that are used to detect the pallets fork pockets. The blocks in the flowchart represent operations that are required to extract the relevant data and/or information. Each operation is reviewed in more detail below.

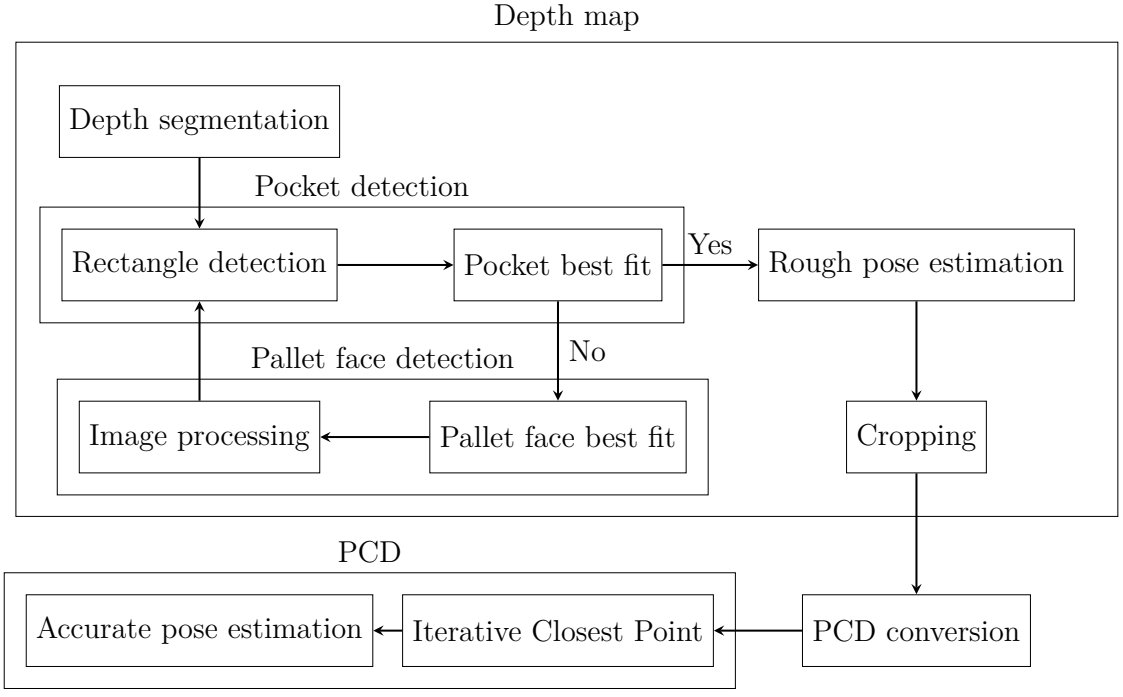


Figure 3.1: Flowchart for the pose estimation process

3.3 Depth map processing

3.3.1 Depth segmentation

The first step is to segment the depth map into smaller slices that are fit for techniques used for 2D imaging. Each segment only contains data that is within a certain span of depth d, emulating a binary bit map as can be seen in figure 3.2. An increase in depth allows for more leeway in the angle of the pallet but will reduce the desired effect of reproducing a 2D image with clear shapes as unwanted features make it into the segment. A decrease in depth might miss features and will reduce the maximum detectable angle of the pallet as parts of the front face will be outside the range of the segment. The step size s is how long it will be between each segment. Increasing the step size will increase performance as fewer segments are created. The depth and step size needs to be configures such that the overlap o is greater than the distance between the front edges of the pallet relative to the camera α. If this overlap is smaller the overlap the algorithm will have blind spots where only half the pallets front is visible in any frame. In figure 3.3 this is visually demonstrated.

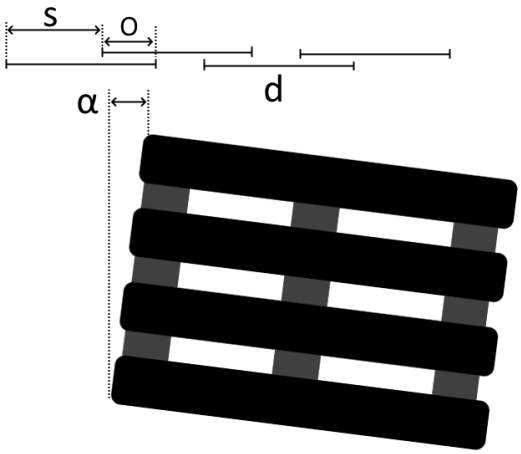
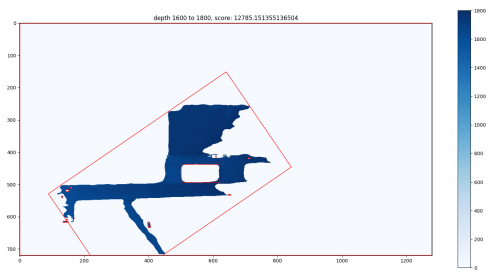


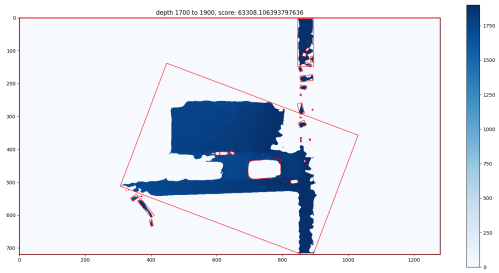
Figure 3.2: 2D view of depth segmentation. Segment depth d and step size s and the pallets "depth" due to it's relative angle to the camera α

A pallet with an angle of 15° compared to the cameras normal will have a α of 207 mm. Setting the step size to 50 mm and the segment depth to 300 mm will guarantee that each 250 mm slice will be fully examined, as the overlap between each segment is 250 mm. The desired overlap can be calculated by: $d - s = o$. The angle 15° was chosen by the case company as a reasonable angle for which a could still be picked by an AGV.

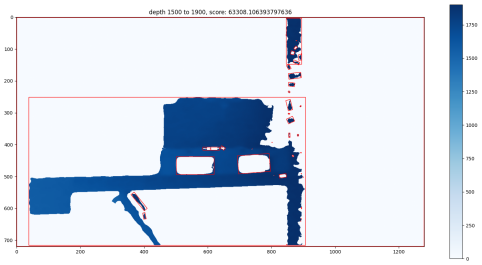
To find a set of values that both resulted in high performance while maintaining accuracy, a sample of 12 depth maps where tested. The depth maps where of pallets with varying angles from -15° to 15°. The depth map also included samples of pallets both on the floor and on a rack. The algorithm was then tested upon each of these



(a) First partial segment



(b) Second partial segment



(c) Full segment

Figure 3.3: (a) & (b) Depth set to 200 mm and step size set to 100 mm. (c) Slice depth changed to 400 mm and is able to capture the full pallet in one slice

depth maps with depth values ranging from 200 to 500 mm and step size ranging from 50 to 300 mm, both in 50 mm increments. The combination with the lowest depth while maintaining full accuracy was $s = 100$ mm and $d = 400$ mm and as such was the combination used for this paper.

3.3.2 Rectangle detection

In each segment, rectangle detection is used to find the pocket holes in the pallet. The algorithm to find these rectangles uses the binary border following algorithm implemented by OpenCV [6]. This converts the segment to a binary image with all non-zero values set to 1. The algorithm then finds all rectangles present in the image, both holes and solid objects. This set of rectangles is then passed on to the best pocket fit to detect which pair of rectangles correspond to the fork pockets. An example of a segment containing a pallet can be seen in figure 3.4a, with all rectangles found by the algorithm marked in figure 3.4b. As can be seen in the figure, far more rectangles are found than the two fork pockets, such as small holes in the surrounding structures.

(a) Segment containing a pallet

(b) All detected rectangles

(c) Best fit pair of fork pockets

Figure 3.4: (a) A segment containing a pallet and some surrounding structures. (b) All rectangles found by the rectangle detection marked in red. (c) The pair of rectangles selected by the pocket best fit.

3.3.3 Pocket best fit

A scoring algorithm finds the rectangles that most likely belong to the fork pockets. Each pair of found rectangles is scored based on how close their properties are to an EU-pallets pair of fork pockets. This allows for minor deviations in the dimensions caused by tilted, damaged pallets or noise in the data. Each corner point is deprojected so that the true height and width of the rectangles can be used.

The height difference y_diff between the two pockets is found by comparing the height of the bottom left corner of each rectangle to each other. This is measured as the height of both pockets should be the same. The width and height of both rectangles are compared to the height $h_{1,2}\_diff$ and width $w_{1,2}\_diff$ of the fork pockets of a EU-pallet. And the center to center distance of the pocket holes are compared to the true distance in an EU-pallet c_diff. Each value is calculated according to equation 3.1. The total score is then calculated with equation 3.2 and the rectangle pair with the lowest score is selected as a potential pair of fork pockets, as shown in figure 3.4c.

$$diff = |measured - EU\_pallet| \tag{3.1}$$

$$score = y\_diff + w_1\_diff + w_2\_diff + h_1\_diff + h_2\_diff + c\_diff \tag{3.2}$$

The winning pair will then be subjected to a few strict criteria before being allowed to pass forward to the rough pose estimation. The rectangles must be within ± 50 mm of the true width and height of a EU-pallets pocket dimensions. And y_{diff} must be less than 10 mm. This criteria prevents the algorithm from evaluating obviously incorrect rectangles in case no fork pockets are detected.

3.3.4 Pallet face detection

All four sides of a rectangle must exist in the image for a rectangle to be found. If the pallet is in a rack and has a overhang, the segment might not contain both the rack and the pallet. This will create two U shapes and the fork pockets will not be detected. If no pair of pockets are found in the segment a second best fit algorithm will be used to find the rectangle corresponding to the face of the pallet. This will look for a rectangle with the closest dimensions to a pallets front edge. If such a rectangle is found a floor will be added to the pallet which will complete the rectangle for the fork pockets. The floor is created by adding a 5 pixel tall line between the two bottom corners. The image will then be sent back to the original pocket detection to find the fork pocket rectangles, as can be seen in figure 3.5. If the fork pockets are not found after this stage the segment will be discarded and the next one will be examined.

(a) Pallet segment **(b)** With rectangles

Figure 3.5: (a) Pallet found in depthmap. (b) Bottom of pallet closed and pockets detected

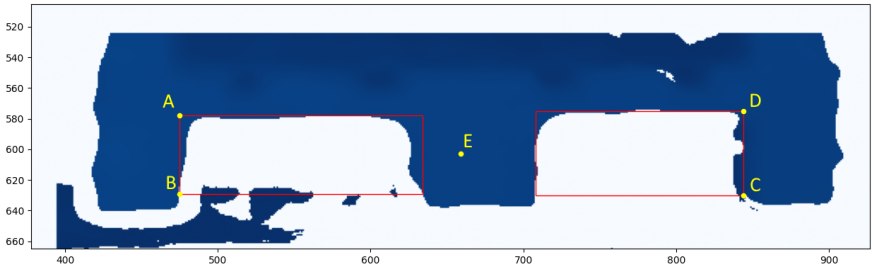


Figure 3.6: Points A, B, C, D, and E on the depthmap

3.3.5 Rough pose estimation and cropping

To get the rough transform from the depth map 4 points are chosen to define the normal of the pallets front face. The corners of the fork pockets are labeled A, B, C, and D as can be seen in figure 3.6. As only 3 points are needed to define the normal one point is discarded, the choice of point to discard is arbitrary unless one point is missing depth data. In the example in the figure, point C can be seen to possibly lack depth data, and as such points A, B and D would be chosen. Point E is the point in the center of the rectangle defined by points A, B, C, and D. Point E is therefore a rough estimate of the front face of the pallet and gives the translation of the rough pose estimation. The rotation is given by the previously mentioned normal. Only the rotation around the y-axis is applied as the x- and z- rotations are assumed to be 0 or near 0. This gives the rough transformation matrix that can be fed to the ICP algorithm as a rough pose estimation.

3.4 Point cloud processing

3.4.1 Cropping and Point Cloud conversion

The depth map is then cropped down to a rectangle that is a bit larger than the pallet to avoid processing unnecessary data in the depth map. The RoI starts of as a the area defined by minimum and maximum edges corners of the found rectangles. On both sides 80 pixels are added while 50 pixels are added above and below. These values where found experimentally. The width is somewhat larger to accommodate the increased size of a tilted pallet.

If a pure crop was made the intrinsic matrix would have to be adjusted to match the new image. Instead of this the values outside of the RoI is set to 0. The 0 values are not deprojected in the next step and will as such function the same as cropping the image without affecting the deprojection.

The depth map is then deprojected into a point cloud. Each non-zero pixel in the depth map can be deprojected into a point using the intrinsic matrix of the camera as shown in equation 2.2.

3.4.2 Iterative Closest Point

The ICP algorithm matches a source point cloud to a target point cloud. The source is an accurate point cloud of an EU-pallet and the target is the point cloud data extracted from the camera. The source is generated pre-runtime by projecting points at a 3D model of a EU-pallet. These points are projected only from a semi sphere originating from the front of the pallet after which only the points 200 mm from the front of the pallet is kept. This results in a point cloud representing the front 200 mm of the pallet without internal structures. It is not expected that the camera will be able to see much more than the front of the pallet and as such features at the back of the pallet would have nothing to match to, and might cause inaccuracies. The origin of the source point cloud is set to the center front of the pallet face respective to point E mentioned in the rough pose estimation figure 3.6.

The source, target and rough transformation matrix can now be fed to the ICP algorithm. The algorithm starts by applying the rough transformation to the source point cloud. Each point in the source is then matched to its closest point in the target, where pairs further apart than a maximum correspondence distance are discarded. A new transformation is then estimated that minimizes the sum of the squared distances between the remaining pairs, and is applied to the source. This process of matching and minimizing is repeated until the change in error between iterations falls below a set tolerance of 10^{-6} or a maximum number of iterations of 30 is reached [8]. In this paper the point to point variant of the algorithm as implemented in Open3D [12] is used.

The resulting transformation is the accurate pose estimation of the pallet which is

visualized in figure 3.7. In addition to the transformation, the algorithm returns the fraction of source points that found a corresponding point in the target (inlier ratio r) and the root mean square error of these correspondences (rmse ϵ). These two values are used to judge the quality of the estimation, as described in section 4.3.

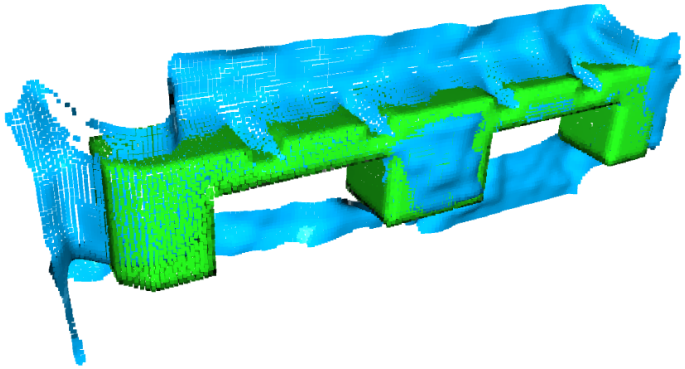


Figure 3.7: Estimated pose (green) superimposed on captured point could (blue)

4
Results

Since no ground truth pose is available, the method is evaluated across three complementary measures. The detection rate captures how reliably the full pipeline produces a result across a range of test conditions. The alignment quality, assessed by combining the ICP inlier ratio r and inlier RMSE ϵ, captures how well the template fits the observed point cloud for each successful detection. The pose consistency, measured as the standard deviation of the estimated vertical position across a video sequence, captures the repeatability of the pose estimate when the pallet is known to be stationary. Together these three measures characterize the reliability and precision of the method without requiring a calibrated reference system. All results presented in this chapter were produced on a computer equipped with a Intel Core Ultra 7 155H processor and 16 GB of RAM.

4.1 Test data

The test set consists of 2678 depth maps of EU-pallets captured at the case company's workshop. Pallets were captured both on the floor and in a rack, at yaw angles of approximately 0°, ±5°, ±10° and ±15° relative to the camera. The depth maps where captured in 14 separate video sections where the camera was placed on a moving table, one for each angle described above. This data does not include the 12 depth maps used to find the step size and depth values in chapter 3.3.1. The capture started at a distance of approximately 1500 mm from the pallet and was slowly moved back to a distance of 3000 mm in a straight line while keeping the angle as consistent as possible. The number of frames per test condition can be found in table 4.1. The pallet on the rack has a neighboring pallet right next to it which has been covered by a sheet. This is to ensure that the same pallet is detected each frame as the algorithm currently has no way to select which detected pallet to move forward to the next step. Since the cameras height above the floor was kept constant throughout, the vertical position of the pallet in the camera frame (y) is expected to remain constant across all frames, making it a suitable axis for a consistency evaluation (Section 4.4).

4.2 Detection rate

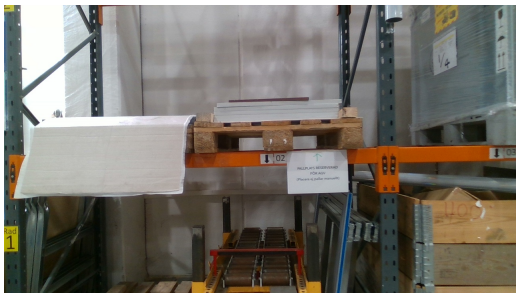
Table 4.1 shows the fraction of frames in which the algorithm detected and accepted a pallet, broken down by test condition. The detection rate is a test of the depth map pallet detections ability to detect a pallet. A detection is considered accepted

if the alignment quality criterion (Section 4.3) is satisfied. The accepted rate is a measurement of how many of the detected frames result in an accepted estimation and is a measurement of the entire algorithms performance.

Table 4.1: Detection rate by test condition

Position	Angle	Frames	Detected	Rate (%)	Accepted	Rate (%)
Floor	0°	144	139	96.53	121	87.05
Floor	±5°	329	315	95.74	250	79.37
Floor	±10°	273	266	97.44	231	86.84
Floor	±15°	331	299	90.33	216	72.24
Rack	0°	238	231	97.06	166	71.86
Rack	±5°	470	385	81.91	218	56.62
Rack	±10°	445	396	88.99	294	74.24
Rack	±15°	448	383	85.94	281	73.37
Total		2678	2414	90.14	1777	73.61

For the floor datasets the detection rate is sufficient at all angles with a drop at ±15°. This drop is expected as a larger portion of the pallets front face risks falling outside a single segment at larger angles, as described in the depth segmentation step of the method. The rack dataset has a satisfactory detection rate for the 0° case but is somewhat lacking at all other angles. This can be explained by the near proximity of the covered pallet on the left, seen in figure 4.1a. The sheet covering the neighbouring pallet creates a large surface at a similar depth as the pallet, which ends up in the same segments and interferes with the pallet face detection.



(a) Color image from rack dataset of angle 0

(b) Color image from floor dataset of angle 0

Figure 4.1: Images of pallets during data capture

The accepted rate follows a similar pattern with the floor datasets generally performing better than the rack datasets. Notably, the rack ±5° datasets have the lowest accepted rate of all conditions at 56.62%, which breaks the otherwise consistent pattern across the angles. This was found to be due to the gray pallet which can be seen to the right in figure 4.1a. In the specific case of a positive 5° angle during the segmentation step, the full gray pallet was detected first while the desired pallet only partially existed in the frame. Due to lower image quality in the corner

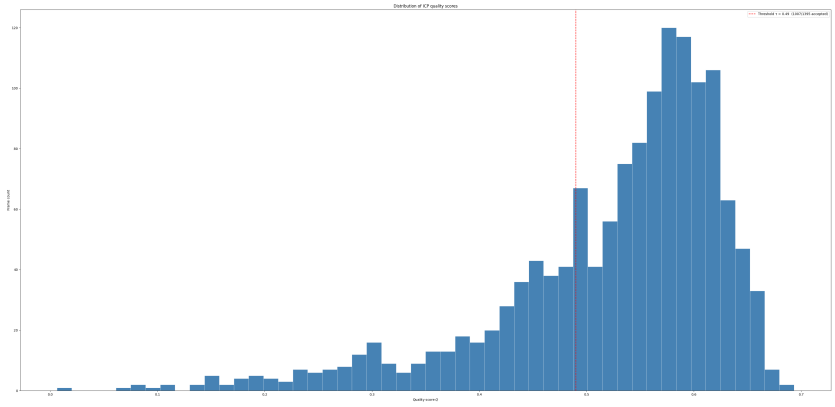


Figure 4.2: Histogram of quality scores with determined threshold

and the gray pallet not always being fully in frame, this cause that specific dataset to achieve a very low acceptance rate.

4.3 Alignment quality

The quality of each ICP alignment is assessed using the inlier ratio r (the fraction of source points with a corresponding inlier in the target, referred to as *fitness* in Open3D) and the inlier RMSE ϵ. Following Phillips, Liu and Tomasi, combining these two values into a single composite score rewards alignments that are simultaneously well-supported and geometrically tight [13]. A detection is accepted if

$$Q = 10 \cdot \frac{r}{\epsilon} > \tau \tag{4.1}$$

where τ is a threshold determined empirically. The threshold τ was determined by inspecting detections across a range of Q values. Detections below $Q = 0.5$ were consistently found to correspond to poor alignments on visual inspection. Figure 4.2 shows the distribution of Q across all frames, with a natural break visible at this value. All reported statistics in Table 4.2 are computed on accepted detections only. Table 4.2 summarizes the mean and standard deviation of r, ϵ and Q across all accepted detections.

Table 4.2: Alignment quality for accepted detections

Metric	Mean	Std. dev.
Inlier ratio r	0.63	0.04
Inlier RMSE ϵ (mm)	10.19	0.82
Quality score Q	0.63	0.08

4.4 Pose consistency

As the camera is moved, the lateral (x) and depth (z) components of the estimated pallet pose vary with the camera's position and cannot be used to assess repeatability without odometry data such as the cameras pose relative to the pallet. The vertical component (y) as mentioned in chapter 4.1, is expected to remain constant throughout the recording. The standard deviation of the estimated y translation across all confident detections in the video sequence is therefore used as a measure of the method's pose precision.
Table 4.3 shows the estimated y values across the video sequence for the rack and the single case of the rack with 0 deg angle.

Table 4.3: Estimated vertical pallet position across video frames

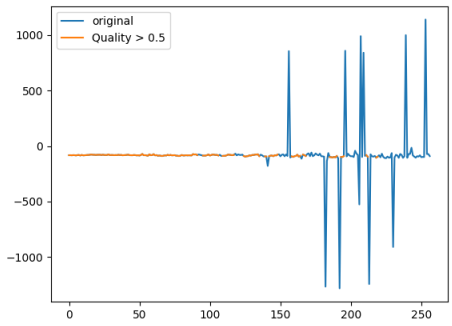
Metric	**All rack (mm)**	**Rack 0 deg (mm)**
Mean $\hat{y}$	-86.19	-85.63
Std. deviation σ_y	7.76	4.59
Min	-125.78	-102.58
Max	-57.00	-72.71
Frames evaluated	959	166

In figure 4.3 the estimated height of the pallet in two cases is plotted for each frame. The values marked in orange are the accepted estimations and the blue values are rejected estimates. The left most values represent nearby values, with the right most values representing distant values. In both figures it can be seen that large deviations in the estimation are filtered out with the quality threshold. It can also be seen that distant measurements are also filtered out. This is likely due to the decreased camera accuracy at longer distances causing larger noise in the captured image. This can more clearly be seen in figure 4.3b as the estimate noise increases with distance.

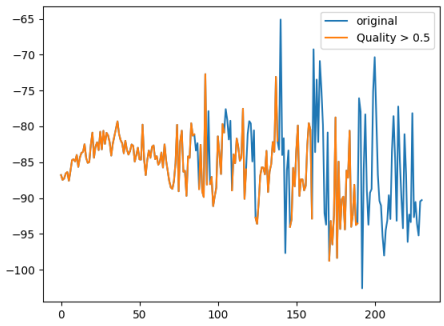
In figure 4.3b it can be seen that the y value slightly increases at first to then steadily decrease over time. This is most likely caused by the cameras movement relative to the pallet, such as uneven flooring or slight shifts in the cart the camera was placed upon.

4.5 Computational performance

A key motivation for the proposed method is reduced computational cost compared to full-scene point cloud processing. Table 4.4 shows the mean execution time for each stage of the pipeline, measured over 231 frames on the previously mentioned hardware. The result is compared to an ICP only setup where the point cloud was not cropped. However since no global registration was available the ICP algorithm was allowed to use the rough estimate from the proposed solution.



(a) y values from rack with angle +5



(b) y values from rack with angle 0

Figure 4.3: y values from 2 datasets, color coded based on acceptance or rejection

Table 4.4: Mean execution time per pipeline stage

Stage	**Mean time (ms)**
Rectangle detection	1.30
Pocket / face best fit	2.58
Rough pose estimation	0.65
Cropping	0.27
PCD conversion	22.21
ICP	274.69
Total	301.70
ICP Only	432.95

As can be seen from the table, the proposed solutions total time is 30.32% faster than using the entire point cloud. Showing a significant increase in performance even without considering global registration.

5
Discussion

In this chapter the proposed method, the evaluation and the results are discussed, followed by suggestions for future work.

5.1 The proposed method

The main idea of the proposed method is that the ROI and the rough pose estimate can be found directly in the depth map with cheap 2D image processing, leaving the heavier point cloud processing to a small region of the data. The depth segmentation makes this possible by reproducing binary images where the pallets features appear as clear shapes. The approach requires no training data and no specialized hardware, which was one of the motivations of this thesis. The downside is that the method is built around the known geometry of an EU-pallet, and a different load carrier would require new dimensions for the best fit scoring and a new template for the ICP step.

Another viable option would be template matching to find the pallet face. This is however sensitive to variations and occlusions. As the distance, and therefore scale of the pallet face in the 2D image is unknown this would be a less secure way to find the load. The perspective also change slightly with the angle, which has not been a major issue with this paper, but could be an issue for template matching. This was briefly explored by the author with poor results but could be a worthwhile investment for another paper.

The segment depth and step size were found by testing on 12 depth maps captured in the case company's workshop. While the values worked well in the evaluation, they were selected in the same environment as the evaluation was performed in. It is therefore possible that the values would need to be re-tuned for a different environment, such as a warehouse with different rack types or lighting conditions. A larger and more varied set of depth maps would give more confidence in the chosen values.

The segmentation also stops evaluating once an acceptable pair of fork pockets has been found. The issues with this was detected with the 5° rack case where the angle caused a pallet to the side to be detected first. An improvement to be made would be to evaluate all the segments in parallel. Parallelization would both increase performance and allow for all pallets in a frame to be detected and compared to each other.

Early in the project it was experimented with detecting the fork pockets with the height and width ratio to avoid deprojections that early in the algorithm. While this seemed to work reliably at first, the change to deproject the corner points proved to be an important improvement. The ratio based filtering was difficult to tune as small holes far away could be interpreted as fork pockets. By deprojecting only the four corner points the full 3D dimensions of the suspected pockets could be checked at a very low computational cost, keeping the spirit of the method while increasing the reliability.

5.2 The evaluation

The largest weakness of the evaluation is the lack of a ground truth pose. The three measures used, detection rate, alignment quality and pose consistency, together give a good picture of the reliability and repeatability of the method, but they cannot show how close the estimated pose is to the true pose. The pose consistency is also only evaluated in the vertical axis, as the camera was moved between frames and no odometry data was available. The accuracy in the lateral and depth directions, as well as the rotation, therefore remains unverified. A reference measurement system, or mounting the camera on a robot with known kinematics, would allow the absolute accuracy to be evaluated and is considered the most important next step.

The quality threshold τ was determined by visually inspecting detections across a range of quality values. While the histogram in figure 4.2 shows a natural break around the chosen value, the inspection was performed on the same data as the evaluation, and the threshold is to some degree subjective. Verifying the threshold on a separate dataset would strengthen the result.

The neighbouring pallet in the rack datasets had to be covered with a sheet as the algorithm currently has no way of selecting which detected pallet to move forward with. This is a limitation of the current implementation rather than the method itself, but it does mean that the evaluation does not show how the method behaves when several pallets are visible at once. This also affected the acceptance rate as was clearly seen in the 5° rack case. The 56.62% acceptance rate was due to other pallets being detected in areas where the accuracy was low, and therefore the accuracy fell sharply.

5.3 The results

The detection rates show that the depth map part of the method reliably finds the pallet, with a total detection rate of 90.14%. The floor datasets perform better than the rack datasets, where the covered neighbouring pallet interferes with the pallet face detection. The accepted rate of 73.61% means that roughly one in four frames does not result in an accepted pose. For the intended application this is less severe than it may appear, as the AGV will receive a continuous stream of frames while

approaching the pallet, and a rejected frame only means waiting for the next one. A false acceptance would be worse than a false rejection, as the AGV could act on an incorrect pose, which speaks for keeping the threshold conservative.

The pose consistency results show a standard deviation of the vertical position of 4.59 mm for the rack 0° dataset and 7.76 mm across all rack datasets. Together with the mean inlier RMSE of around 10 mm this indicates that the accepted estimations are both repeatable and geometrically tight. The variation is small in relation to the size of the fork pockets of an EU-pallet, and is therefore deemed to be within what the pick operation can tolerate.

The estimations were also seen to degrade with distance, where the increased noise of the camera at longer ranges caused larger spread and more rejections. For the intended application this is acceptable as the pose estimate matters the most during the final approach, where the camera is close to the pallet.

5.4 Computational performance

The proposed method was 30.32% faster than running ICP on the full point cloud. This comparison should however be read with some care. A true comparison should be done against a pipeline with a global registration providing the rough estimate, but the author was unable to successfully create a global registration within the time limit of the thesis. Instead the ICP only setup was allowed to use the rough estimate from the proposed solution. Since a global registration would add a significant computational cost of its own, the reported difference is likely an underestimation of the true advantage of the proposed method.

It is also worth noting that the depth map stages of the pipeline only account for about 4.8 ms of the total 301.70 ms. The cost of finding the ROI and the rough pose estimate is in other words almost negligible, and the remaining time is dominated by the PCD conversion and the ICP algorithm. Further performance gains would therefore come from the point cloud side, for example by downsampling the point clouds before registration.

With a total time of 301.70 ms the method produces around three pose estimations per second. Whether this is sufficient depends on the approach speed of the AGV and the requirements of the case company.

5.5 Future work

There are two main features that the author believes would have a significant impact on future work. The first is to investigate other accurate pose estimation tools. Most of the authors time in this paper was spent on finding the rough estimation. But the greatest computational time saver for future works is found in the ICP step. There are several variations of the ICP algorithm that could be combined with the rough pose estimation of this paper that could either improve accuracy or time.

The second would be to refine the scoring algorithm as the current implementation can find neighboring pallets from the one which is desired. Utilizing the AGVs odometry to find the cameras position would allow for a set range to be searched instead. Future work could also attempt to detect every pallet based on some threshold value for the score.

Other future work could also investigate the possibility to extend to other loads other than EU pallets. The depth map part of this paper is already somewhat usable for other loads with similar pocket dimensions. It could however be extended by adjusting which parameters are scored to include any object which could be picked by a fork lift type AGV. The ICP algorithm does not lean itself towards this as a template has to be used. However, a set of templates could be used if the ICP part could be improved. It is also possible that template matching could be used to detect which template to attempt ICP with.

It could also be investigated to record several estimations and average the values between them to possibly gain a higher degree of accuracy. Though a faster implementation of the accurate estimation would be recommended before that.

6
Conclusion

In this thesis a method for estimating the pose of a pallet using a depth sensing camera was developed and evaluated. The method uses a novel approach to find the ROI directly in the depth map by segmenting it into depth slices and detecting the fork pockets of the pallet with fast 2D image processing. A rough pose estimate is calculated from the detected pockets, after which only a small region around the pallet is deprojected into a point cloud where the pose is refined with the ICP algorithm. The method requires no machine learning and no specialized hardware.

The method was evaluated on 2678 depth maps of EU-pallets on the floor and in a rack at angles up to $\pm 15°$. A pallet was detected in 90.14% of the frames and 73.61% of the frames resulted in an accepted pose estimation. The standard deviation of the estimated vertical position of a stationary pallet was 4.59 mm in the best case, showing a high repeatability. The complete pipeline was 30.32% faster than running ICP on the full point cloud, even though the comparison favoured the full point cloud setup, showing that the computational load can be reduced without losing accuracy.

The results show that detecting the ROI directly in the depth map is a viable and lightweight alternative to machine learning based detection for the intended application. Before the method can be deployed on an AGV, the absolute accuracy needs to be verified against a ground truth and a selection logic for multiple detected pallets needs to be added, as discussed in the future work.

Bibliography

[1] E. Tsiogas et al., "Pallet detection and docking strategy for autonomous pallet truck agv operation," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3444–3451, Sep. 2021. DOI: 10.1109/iros51168.2021.9636270. Accessed: May 11, 2025.

[2] B. Wu, S. Wang, Y. Lu, Y. Yi, D. Jiang, and M. Qiao, "A new pallet-positioning method based on a lightweight component segmentation network for agv toward intelligent warehousing," *Sensors*, vol. 25, pp. 2333–2333, Apr. 2025. DOI: 10.3390/s25072333. [Online]. Available: https://www.mdpi.com/1424-8220/25/7/2333.

[3] V.-D. Vu et al., "Occlusion-robust pallet pose estimation for warehouse automation," *IEEE Access*, vol. 12, pp. 1927–1942, 2024. DOI: 10.1109/ACCESS.2023.3348781.

[4] Y. Li et al., "Pallet localization techniques of forklift robot: A review of recent progress," *Journal of Robotics and Mechanical Engineering*, vol. 1, no. 1, pp. 1–7, 2022.

[5] B. Molter and J. Fottner, "Real-time pallet localization with 3d camera technology for forklifts in logistic environments," *mediaTUM – the media and publications repository of the Technical University Munich (Technical University Munich)*, Jul. 2018. DOI: 10.1109/soli.2018.8476740. Accessed: Nov. 23, 2023.

[6] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, pp. 32–46, Apr. 1985. DOI: 10.1016/0734-189x(85)90016-7.

[7] Y. Shao, Z. Fan, B. Zhu, J. Lu, and Y. Lang, "A point cloud data-driven pallet pose estimation method using an active binocular vision sensor," *Sensors*, vol. 23, p. 1217, Jan. 2023. DOI: 10.3390/s23031217. Accessed: May 19, 2025. [Online]. Available: https://www.mdpi.com/1424-8220/23/3/1217.

[8] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992. DOI: 10.1109/34.121791.

[9] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, pp. 119–152, Oct. 1994. DOI: 10.1007/bf01427149. Accessed: Nov. 21, 2021.

[10] N. Kai, H. Yoshida, and T. Shibata, "Pallet pose estimation based on front face shot," *IEEE Access*, vol. 13, pp. 37 624–37 631, 2025. DOI: 10.1109/access.2025.3538045. Accessed: Oct. 22, 2025.

[11] G. Bradski, "The opencv library," *Dr. Dobb's Journal of Software Tools*, 2000.

[12] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv:1801.09847*, 2018.

[13] J. M. Phillips, R. Liu, and C. Tomasi, "Outlier robust icp for minimizing fractional rmsd," in *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM 2007)*, 2007, pp. 427–434. DOI: 10.1109/3DIM.2007.39.

A

Algorithm parameters

The table below summarizes the parameter values used by the algorithm during the evaluation.

Table A.1: Parameter values used in the evaluation

Parameter	**Value**
Segment depth	400 mm
Step size	100 mm
Crop margin, sides	80 pixels
Crop margin, above and below	50 pixels
Quality threshold τ	0.5
Maximum pallet yaw	$\pm 15°$
ICP maximum correspondence distance	[VALUE]
ICP maximum iterations	[VALUE]

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY