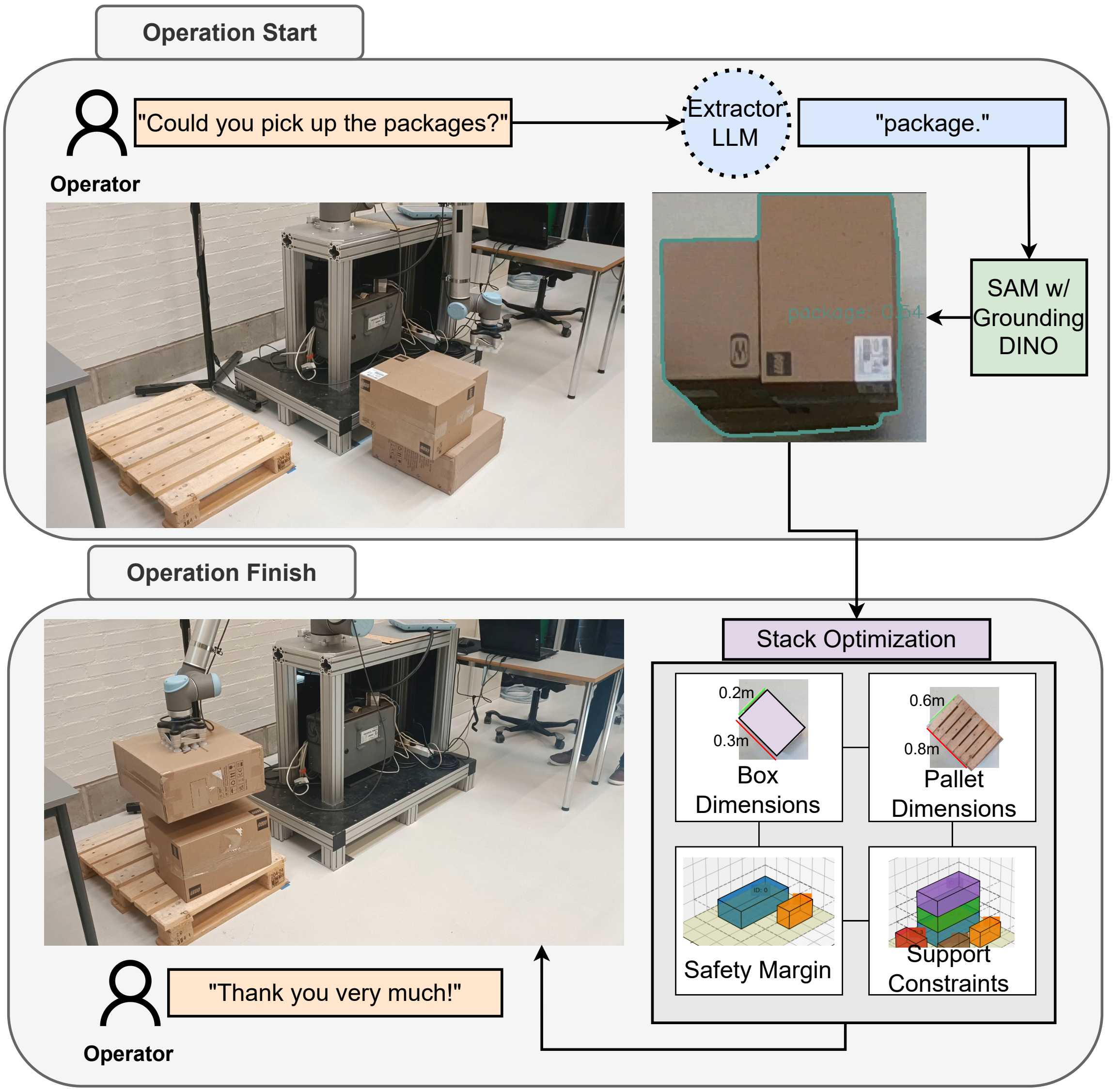
Smart Vision-Guided Robotic Depalletizing System

Project Report

ROB10 - 1076



Aalborg University
The Technical Faculty for IT and Design
Department of Electronic Systems

AALBORG UNIVERSITY
STUDENT REPORT

The Technical Faculty for IT and Design
Niels Jernes Vej 10, 9220 Aalborg Øst
http://www.aau.dk

Title:
Smart Vision-Guided Robotic Depalletizing System

Theme:
Theme

Project Period:
Spring Semester 2025

Project Group:
1076

Participant(s):
Christian Aleksander Hjorth 20203854

Supervisor:
Dimitris Chrysostomou

Copies: 1

Page Numbers: 78

Date of Completion:
June 3, 2025

Abstract:

This thesis presents the development of a flexible and vision-guided robotic system for depalletizing and palletizing tasks, designed to advance automation in dynamic logistics environments. Built within the ROS 2 framework and integrated using MoveIt2, the system combines RGB-D perception, large language models (LLMs), and state-of-the-art segmentation and detection models (GroundingDINO and SAM) to enable natural language interaction, object segmentation without prior training, and robust pose estimation.

A complete hardware-software stack was implemented using a UR10 robot, VG10 vacuum gripper, and an Intel RealSense D455 RBG-D camera. The solution features intelligent stacking through constraint-based optimisation, ensuring both stability and efficient space utilization. The system was successfully validated through real-world tests, demonstrating reliable perception, grasping, and intelligent object placement.

Key contributions include the integration of LLMs for natural language interpretation, promptable segmentation models for adaptable object identification, and a constrained optimisation approach for intelligent box stacking.

Despite the strong performance, areas for future improvement were identified, including enhanced segmentation through better camera configurations, more precise pose estimation for visually similar objects, refined system prompts. Overall, this thesis establishes a solid foundation for future human-robot collaborative systems that prioritize both flexibility, usability and efficiency.

Classification notes.

Preface

This master thesis is the materialization of the work done at Aalborg University in the first semester of 2025. The presented project is also a product of the continuos 5 years of studies in Robotics at Aalborg University. This is also a transitional moment in my life from being a Master's student into being a curious professional in this field.

I have been very fortunate to receive support in each of the steps that I have taken at the university. From the first day of my studies to the very last, I have felt that I have been valued and helped throughout my journey. I am very interested in Human-Robot Interaction with the recent advancements of Large Language Models and transformer-based models. The gap between humans and robots is closing and collaboration is becoming more effective and flexible. There has never been a more exciting time than now and I am grateful to have chosen Aalborg University to provide me with the education needed for it.

I would like to state my gratitude towards to Aalborg University as a whole and all of its staff who were always ready to help me when needed. Special thanks to Andreas, lab engineer, for helping me with practical hands-on challenges this semester, and also to Lasse and Søren my fellow students who shared constructive discussions about mine and their Master thesis. Lastly, however not least important, I would like to share my gratitude towards my supervisor Dimitris Chrysostomou, who has always helped me since my very first semester at Aalborg University to the very last in my Master studies. Thank you very much Dimitris for all the effort you have put into guiding me throughout my education, it is very much appreciated.

I would like to dedicate this project and my education to my lovely sister Veronica, whom unfortunately left us all too soon this April. You are the reason I am the person that I am today. Thank you. From your little brother, Chris.

Christian Aleksander Hjorth

Christian Aleksander Hjorth
chjort20@student.aau.dk

Contents

1 Introduction **1**

2 Related Works **4**
2.1 Depalletizing Frameworks 4
2.2 Object Detection and Segmentation 8
2.3 6D Object Pose Estimation 12
2.4 Human-Robot Interaction 16

3 System Design **18**
3.1 Use Case 18
3.2 Hardware 18
3.3 System Architecture 21

4 Implementation **28**
4.1 Intrinsic Camera Calibration 28
4.2 Extrinsic Camera Calibration 31
4.3 Setup 34

5 Testing **49**
5.1 Vision System Test 49
5.2 Pose Estimation 52
5.3 Extra Data on Image Segmentation and Pose Estimation 56
5.4 LLMs Object Extraction Test 60

6 Discussion and Conclusion **62**
6.1 Object Segmentation and Pose Estimation 62
6.2 LLMs Analysis 64
6.3 Thesis Achievements 66
6.4 Conclusion 68

Bibliography **70**

A Links **75**

B Appendix - Test Data **76**

1 Introduction

The recent advancements in automation logistics and manufacturing have enabled a shift in the processes performed in the industry. The progress in the technology used in automation and robotics has made it possible to have high-complex systems operating in the industry with high efficiency and accuracy while reducing costs of processes and enhancing worker safety. The use of automated systems can reduce error in high-paced manufacturing systems where stops in the production can result in high cost even if for short amount of time. [1], [2]

In the context of depalletizing/palletizing there are benefits involved in automating and integrating robotic systems in these operations. The use of vertical storage and the maximization of space used in warehouses and manufacturing floors allow for significant storage possibilities. The use of robots allow for better work conditions for the company's employees. Palletizing/depalletizing that previously were performed manually or with the aid of vehicles such as forklifts can now be done with robots and in automated environments, which create safer work conditions and also free the workers from dull and repetitive work. Efficiency is also another crucial factor that highlights a clear advantage in having automated robotic systems, throughput numbers can increase, since specialized systems can be made for the operation and the robots and machines can operate during the whole day. The automation also reduces the number of errors that can slow down production, since vision systems can ensure the correct placement of parcels and automated inspections can be implemented. Damage to goods are also mitigated with the introduction to systems that have consistent performance, the manual transport by humans in repetitive tasks can result in goods being damaged as well as humans and slowing down production and decreasing worker safety. [1], [3]

In the supply-chain depalletizing acts as a strong link between delivering and receiving products. As for example, when trucks with parcels are needed to unload both the delivery and the receiver must cooperate in order to transport the goods from one to the other. This highlights how impactful depalletizing can be since it is a crucial and time consuming operation that can be efficiently done with the help of automated systems. [4]

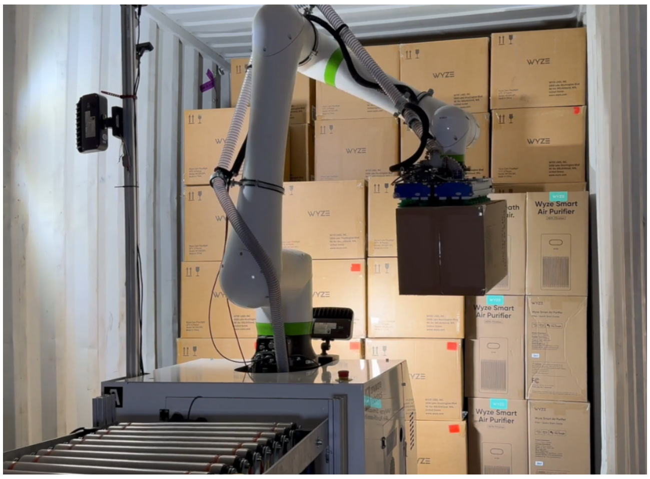


Figure 1.1: Truck being unloaded by a robot arm in a depalletizing operation. [5]

There are, therefore, benefits related to automating transport logistic, specially in the context of depalletizing/palletizing operations. These are crucial in the supply-chain and can enable efficient unloading/loading of goods. An example of depalletizing operation is seen in figure 1.1, where a robotic arm is taking parcels from a truck and putting them on a conveyor belt. There are though challenges when implementing automation systems like the one seen in figure 1.1. Many of the developed automated systems rely on boxes of known dimensions and can also require them to be placed in pre-defined positions. More intelligent systems have also been developed to allow for greater flexibility. RGB-D technology has been used and vision has been a crucial part of the newer systems. These, however, are highly dependent on lighting conditions for sufficient object detection/segmentation and also have problems due to parcels placements. Boxes closely placed to each other and with uneven shapes and rotated make the detection of them difficult. As it is difficult to locate box edges and there is low contrast in the image, these become difficult to detect. If boxes are not perfectly stacked detection of its pose and most predominant face can also occur.[6]

The literature has shown that most of the developed systems are not flexible and require preconditions such as: known box dimensions, predefined placement and orientation, and extensive training in deep learning approaches for more recent systems. There is a clear gap in what can be achieved in depalletizing/palletizing operations with the current technologies. More flexible systems without the extensive training time for detecting parcels and possibility of generalization for different parcel types can indeed now be achieved while also enabling human-robot interaction for more efficient changes in production systems. A system that can be easily adapted to different parcel types by simply an operator command.

In this thesis, a flexible approach to depalletizing/palletizing is presented, a vision-guided flexible robotic system that aims to address the gap seen in the literature and commercial solutions. Some of the contributions include:

- Use of **RGB-D vision** for environmental perception.
- Integration of **LLMs** for natural language command understanding and flexible object specification.
- Advanced and promptable segmentation models (e.g., **GroundingDINO and SAM)** for identifying diverse objects without prior training on specific instances.
- Robust **pose estimation** techniques
- An **optimisation** approach for **Intelligent stacking** of retrieved boxes.

The presented solution to depalletizing/palletizing challenge is composed of three main hardware components: a UR10 cobot from Universal Robots, VG10 vacuum gripper from OnRobot, and the Intel RealSense D455 RGB-D camera. The developed system used ROS2 for integrating all the different control aspects and used a MoveIt2 to control the robot while avoiding collisions.

The proposed solution is made based on RGB-D vision system for detection of parcels. A camera with a top view of the boxes is used to detect them in the environment and then use the depth information to find the pose of the object in the robot coordinate system. The boxes are located in the image with a combination of GroundingDINO and SAM (Segment Anything Model). The combination of the two makes it possible to prompt the object to be segmented. This is therefore the combination of a text string, for example ”box” or ”package” and the RGB image of the environment containing the boxes.

The text string given to SAM with GroundingDINO is extracted from a command given by the operator. The operator would write ”Please, pick up the packages.”, and then the dual LLM architecture would be used to extract the object requested ”package”. The LLM responsible for the object extraction should then extract from the text ”package”, and then the second LLM would verify if the extraction has been done correctly. If the extraction is valid, the operation continues and the extracted string ”package” is sent to SAM with GroundingDINO for image segmentation.

After the image is segmented with the combination of SAM and GroudingDINO, the package's pose needs to be estimated in order to proceed with the depalletizing operation. As the parcels are assumed to have cuboid forms, the pose estimation is done with RANSAC and the box's top dimensions are also estimated.

After the pose estimation is done the robot picks up the package while checking for sufficient pressure from the vacuum gripper to ensure a safe and firm grasp. From there, the stack optimisation for optimal box placement is performed to ensure the boxes are placed properly and compact while taking into consideration the pallet's dimensions, support constraints, and safety margins. After the pox is optimally placed, then the other can be placed on the pallet and the operation continues until there are no more boxes to be placed.

The thesis will have the following structure:

- Related Works
- System Design
- Implementation
- Testing
- Discussion and Conclusion

The first part of the thesis following the introduction is **Related Works** in which the relevant literature will be evaluated in order to understand the challenges involving depalletizing/palletizing operations, and to identify possible gaps that can be explored in a solution.

After this part of the thesis, the **System Design** will be presented with the hardware and software used to make the proposed solution. A general pipeline is presented to help illustrate the processes performed in the depalletizing operation.

This section of the thesis will be followed by **Implementation** which explains the actual performed work to achieve the proposed solution, where flowcharts and ROS2 environments and nodes are explained.

The developed solution is then examined in the **Testing** section where the vision and language systems of the solution are mainly tested.

The results from the testing are critically evaluated in the **Discussion and Conclusion** section of the thesis. The achievements of the thesis will be discussed and the future directions will be proposed followed by a conclusion of the thesis in a general manner.

This thesis is based on a use case of company named RiACT from Denmark, as they have use cases in both depalletizing and palletizing operations. RiACT provides automations solutions with skill-based robot control. [7]

2 Related Works

It is on interest to analyse the available depalletizing solutions. Different sensors and vision techniques can be used to perform depalletizing. After the analysis of different frameworks used for depalletizing, a more general observation will be made into vision techniques. Mainly, object detection and segmentation as well as 6D object pose estimation. These topics are crucial in understanding how to detect objects and also find their pose in the world coordinate system. These will be followed with the presentation of Human-Robot Interaction and the possibilities of integrating Large Language Models to allow flexible and effective interaction between operators/users and robots.

2.1 Depalletizing Frameworks

In this section, depalletizing methods seen in the literature will be analysed, as there are several ways to structure an automated depalletizing framework. General aspects as cost, efficiency, complexity and others will be used to examine the proposed solutions from the literature.

A relevant work in the depalletizing literature is seen in [8], in which the authors proposed a method for depalletizing using a gantry robot and a telescopic arm. The solution has a large workspace and is capable of detecting packages stacked in complex manners through the use of clustering of depth points and morphological processing of the RGB images gathered. This solution although proved to be of robustness it has a high cost to implement and it has a high degree of complexity in its implementation.

The authors of [9] have proposed a method using 2D range imagery in order to depalletize. Their system can be used in real time and is not dependent of light due to it using a time of flight sensor to operate. The system was, however, only tested in cases where the dimensions of the boxes were already known and the layers were properly structured. This means that the system has not been tested in unstructured and complex pallet configuration and without the knowledge of the boxes dimensions beforehand.

The literature also is composed of [10] where the authors have also proposed another method based on the use of time of flight sensor and the scanning of the upper layer of the boxes on the pallet. From the scanning the edges are detected and line fitting is used to find the 3D vertices, which are then forwarded to an object recognition system. The approach is simple, versatile and robust as proven by the authors, however in the case of neatly placed boxes in a pallet, the system then struggles to identify them, and also requires an extensive amount of resources in order to compute solutions.

Another method has been proposed in more recent years as seen in [11], where both a hardware design and a a system for box detection where a linear actuator has been implemented a suction gripper and a RGB-D camera to detect the boxes. The system uses a detection algorithm where the top surfaces of the boxes are viewed by the camera and used to define the length and width of packages. The system has shown a detection precision of around 10 mm, however it can not detect robustly in case of boxes with logos and images printed on it. The system also requires a large workspace to function and the cost and maintenance are of the solution are high.

Relevant literature is in the field of depalletizing is present in [12], as the author has investigated a flexible depalletizing system in the area of supermarket logistics. The solution consists of an industrial manipulator with a suction grip and also has a RGB-D camera in the workspace to detect and estimate the placement of the boxes. The robot is placed in the center where the pallets are in its surrounding. The testing, however was made for small experiments, for example a total of only 10 boxes, and its scalability and application to the industry and not only supermarket logistics is yet to be proven.

The use of a lidar based recognition and location framework has peen proposed by in [13], where horizontal scans are made at different heights for each layer of stacked boxes. The system is able to provide precise pose estimation of the boxes and uses corner features in its method for box recognition. The system requires, however an operator to input the height, length and maximum number of layers, and the conducted experiment was made with standardized carton boxes all of with same dimensions and neatly stacked layers. The system is therefore not yet proven to work with unknown box dimensions and random pallet configurations.

The author of [14] has proposed in his solution an approach to train CNN models for depalletizing tasks with the aim to reduce the amount of data needed as well as the time to be able to perform depalletizing operations setup. The method proposed utilizes of data augmentation and transfer learning strategies to minimize the amount of data and time needed to have a functional CNN for object detection.

The work performed in [15] proposes an approach to detect and localize cases in mixed pallets, which are often seen in supermarket logistics. The proposed method uses a single RGB-D camera as the perception sensor. The system is composed of a database of the pallets containing, information about the number of boxes present, their dimensions and images of the box faces. SIFT is used in their detection algorithm and the knowledge of the box dimensions is used for comparing the detected width and height to their reference.

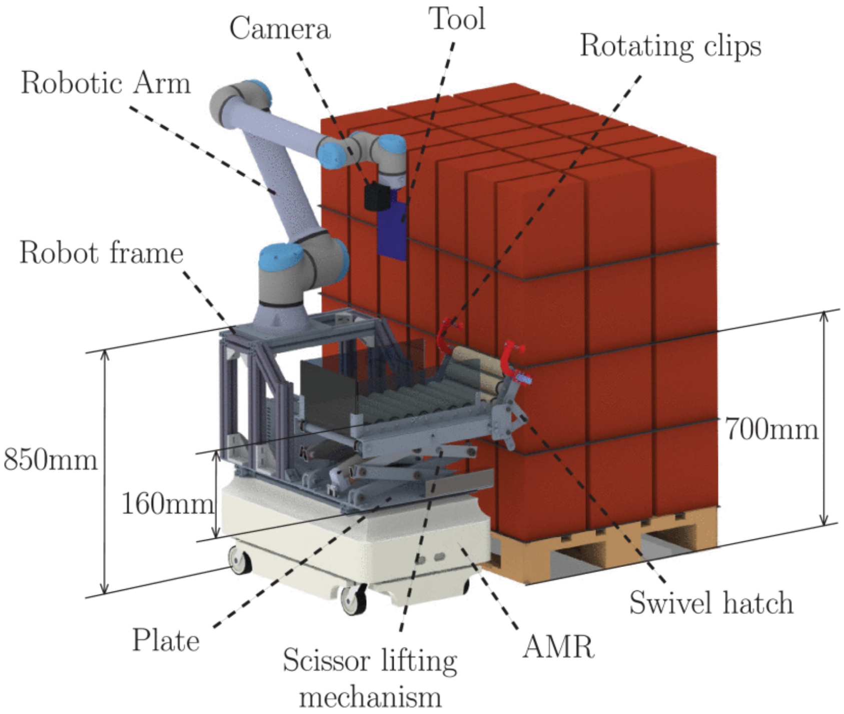


Figure 2.1: Mobile depalletizing system that uses conveyor belts and RGB-D technology to pick up the parcels. Setup is composed of MiR100 and UR10 cobot. The conveyor belt can move in the vertical direction if needed. [16]

Another relevant work has been made in [16], in which the author has proposed a method for a mobile system capable of depalletizing operations. The system consists of a mobile robot, namely MiR100, a UR10 cobot, and a lifting mechanism to lift the conveyor mounted on the MiR100. For identification of the parcel boxes, the system uses a RGB-D camera and is able to detect boxes with a single RGB-D frame as explained in [17]. The designed setup is seen in figure 2.1. Their method for object detection consists of the use of a Canny edge detector to a light compensated image and for the depth edge detection a distance threshold is used to determine whether or not a point is part of and edge of the top flat layer plane. The plane is found through RANSAC, and it assumes therefore that the top layer is flat. After that the points identified as edges are used to extract lines through Hough Transform. The set of lines are used to find where intersections occur and used to delimit them as corners. The corners are then used as nodes to form a connectivity graph, where corners connected by the same line are also connected in the graph. After that, quadrilaterals re found by checking the lengths of the possible box face, while also maintaining a close to 90°angle. These conditions are used to identify candidates that actually resemble the box's true dimensions and geometrical structure. An objective function is made to find the most plausible set of boxes of the boxes. Where, the main principle behind the function is to select a subset of boxes that: (i) cover an extensive area without outlier pixels, (ii) have dimensions close to the expected ones, and (iii) do not have significant overlap with other boxes. The optimal solution to the objective function is computed through a genetic algorithm in order to lower computational resources.

The work made in the [18] has proposed a solution for depalletizing tasks consisting of two-stage deep learning methods. The proposed method consists of using YOLOv3 for object detection and using YOLOv8 for keypoint estimation. The author also highlights the limitations of classical algorithms in for depalletizing tasks. The lack of flexibility and capability of handling occlusion and more complex scenes are mentioned. Real-time performance and sensitivity to light are also negative points of more traditional methods as mentioned by the author, which is used to justify the use of Deep Learning methods.

There is also work [19] that uses RGB-D camera as its sensory input and the use of deep learning methods to efficiently identify the edges of the boxes. The proposed solution is even capable of accurately detecting parcels with uneven surfaces and irregular shapes. Rotated parcels most shown face are also detected with the proposed method. This work is mainly focused on a vision detection system rather than a full depalletizing system. The use case from the paper is specifically for truck unloading, where the tests consisted of rotated and normal parcel orientations.

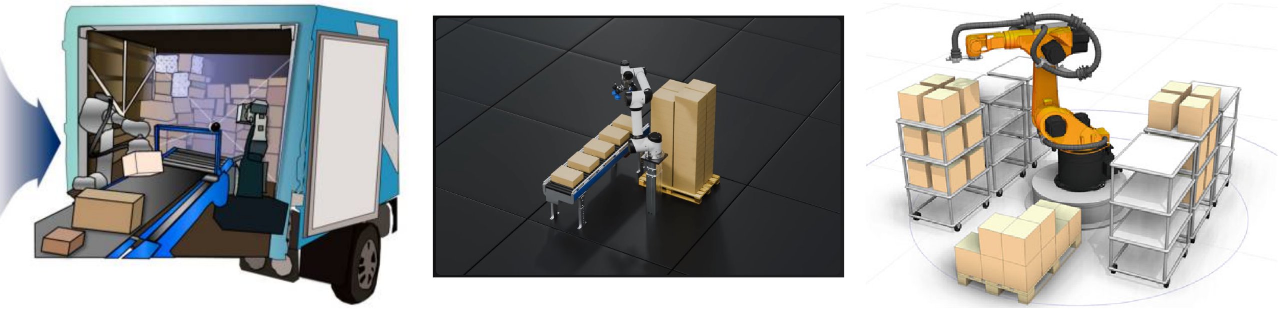
The authors of [20] have investigated a robust approach to recognize boxes in for depalletizing under challenging conditions, which can often be the case in depalletizing operations, as cluttered image, labels, overlapping boxes might occur when depalletizing. Their approach is made if a two-stage deep learning framework and it relies on RGB-D imagery. The first stage is a pre-processing of the data with Cycle-GAN, Cycle-Generative Adversarial Network. The purpose of Cycle-GAN is to remove visual noise (labels, stickers, tags) from the images. Cycle-GAN consists of 3 convolutional layers, 6 residual blocks and 3 deconvoluitional layers with this architecture Cycle-GAN is able to transform images with labels and stickers to "cleaner" images without those. In order to train Cycle-GAN images of two domains were used, images of boxes with and without tapes, labels and tags. The boxes were also arranged in three forms: stacked, inclined and overlapping stacks. The second stage, of the two-stage method, utilizes Mask R-CNN, which was trained on both normal images and images generated by Cycle-GAN. The operation stage occurs in three main process. The first process is the elimination of background and noise through ROI extraction. The second phase is the image modification by the use of Cycle-GAN. Lastly, the boundary detection is applied based on the deviation of depth of the points present in the depth image. The authors show the comparison of their methods with and without the additions proposed. The first method shown is the use of solely Mask-RCNN, the second is the use of Mask-RCNN and Cycle-GAN. The third and proposed method is the combination of Cycle-GAN and boundary detection in combination with Mask-RCNN to segment the images. The authors mention other's work[21] in their findings of an acceptable threshold of 30 mm for depalletizing, and use it as a base for their testing. Two arrangement types were evaluated in the testing, boxes apart from each other and boxes at close position to make it more difficult to detect their edges. In the first method, without boundary detection and -Cylcle-GAN, the detection rate under the 30 mm threshold was 100%, however under the 15 mm deviation was only 57.5%, as seen in table2.1. The 100% detection rate was only achieved by the proposed method for deviations under 15 mm in discrete box positions. In the case of complicated and close boxes arrangement, the selective methods 1 and 2 did not have a reintegration with a deviation under 30 mm in all cases. It failed respectively 39% and 12% of the time. In the case of the proposed method it maintained all of the recognitions under 30 mm and 96% under 15 mm while also having the lowest average deviation. The authors of the paper have proved how the challenges associated with stickers, labels and symbols can be solved by the use of Cycle-GAN, which is helpful to reduce time and resources used in gathering an extensive amount of training data. The study though only analysed images containing 4 boxes in their testing and it is therefore not yet proved to work in larger scale with more complicated arrangements and more boxes.

Table 2.1: Quantitative accuracy of the experiment in each case.

Arrangement type	**Method**	**Average deviation**	**Successful recognition rate** (Under 15 mm of deviation)	**Failed recognition rate** (Over 30 mm of deviation)
Type 1 with discrete box positions	Selective method 1	13.00 mm	57.5%	0%
	Selective method 2	9.35 mm	74.5%	0%
	Proposed method	**3.79 mm**	**100%**	**0%**
Type 2 with complicated box positions	Selective method 1	32.94 mm	28.5%	39%
	Selective method 2	19.58 mm	43%	12.5%
	Proposed method	**7.034 mm**	**96%**	**0%**

Depalletizing Frameworks Analysis Summary

Most of the literature consists of systems that have many constraints in order to make a depalletizing system function. As seen in the literature review, systems consist of pre-defined placements for the boxes with known dimensions. This lack of flexibility is also pointed in the literature[6], whom states that most of depalletizing systems are highly standardized and do not present flexibility. There are few systems that have tried more challenging cases with the use of CNNs, however most of them require extensive training data, which can take sufficient time to gather. There is a clear gap in the literature for a flexible, generalizable depalletizing framework that allows for more flexibility in box placement and dimensions. Three main forms of depalletizing/palletizing operations stand out, truck unloading, conveyor belt unloading, and pallet unloading.



Truck being unloaded with robotic help.[19]

Palletizing operation by RiACT, where boxes come from conveyor belts and are stacked on pallets.[7]

Unloading of pallets in a supermarket environment.[22]

Figure 2.2: Three types of depalletizing scenarios, first being truck unloading, conveyor belt unloading and pallet unloading.

Each of the afore mentioned cases can be seen in figure 2.2. The first use case with truck unloading presents challenges in detection/segmentation of the parcels as these can be rotated from the movement of the truck and the faces might be only partially visible. The parcels are also close to each other making edge detection more challenging. As for the case of conveyor belt with one parcel at a time, detection is more simple as there is contrast with the background with the parcel. The case of pallet unloading is similar to the truck unloading where parcels are close to each other and might be rotated in the transportation phase. For an intelligent and flexible system, the use of RGB-D camera present a viable solution, as the use of deep learning techniques with object detection/segmentation allow for accurate detection of parcels of different sizes and unknown dimensions. For these types of solution a two-stage approach with object detection/segmentation followed by object pose estimation is the standard approach seen in the literature.

2.2 Object Detection and Segmentation

There are three main stages in traditional object detection methods [23]. The first stage is region selection, since the object can appear in different parts of an image it is necessary to identify where it is located . This identification is done through a sliding window approach with several scales. This is though resource demanding. The second phase is the feature extraction, which uses approaches like HOG [24], SIFT [25] and Haar-like [26] in order to retrieve features that represent the object in a significant way. A challenge is though lighting, object orientation and background, which makes it difficult to recognize objects when all these conditions may vary. The last stage is the classification where the use of classifier recognize the desired object, an example of a classifier would be Adaboost [27]. The recent advancements in deep learning have though helped solve some of the challenges faced by traditional methods, and subsequently the detection rate has substantially improved, since the use of these novel approaches. In deep learning there are two main categories of object detectors: two-stage- and one-stage detectors.

Two-stage Object Detection

In this object detection approach, the task is divided into two general parts, object localization and then classification. The object localization consists of identifying region proposals where the object might be located. After that, these regions are analysed an objects are classified. This structure is known as Region-based framework [28]. This type of object detection has a high accuracy due to its double stage, however they are slower as well in computation time due to requiring more resources.

RCNN

Region-based convolutional neural network, or RCNN, seen in [29]. This object detector works in four phases. The first phase is region proposal, in which selective search is used to produced approximately 2000 proposals per image. The second phase consists of scaling the regions proposed to match the input size of the CNN, and then features extracted. The third stage is composed of object classification with the use of SVM classifier. The last stage consists of generating bounding boxes with the aid of linear regression to make them tighter to the objects themselves and therefore not including unnecessary parts of the image. Even though RCNN showed an increase in detection rate it still had some deficiencies, as slow processing time, multi-stage training.

SPP-Net Due to RCNN generating 2000 region proposals for each input image, the extraction of features from the selected areas was the main challenge in RCNN. In regards to the input size constraint of RCNN, it only happens due to the fully connected layers after the convolutional layers, SPP-Net [30] adds a spatial pyramid pooling layer, which makes the input size of fixed-length due to it having a constant number of bins regardless of image size. This method increased speed in computation time while maintaining the same detection accuracy, since the complete image only needs to be run once in order to extract features from it, and resulting in fixed-length input for the fully connected layers.

Fast RCNN
From the same author of the poposed RCNN, Fast RCNN [31] was developed to improve some of the shortages seen in the previous work of the author. As for the 2000 times the CNN is run per image, in this version it is only done once. A RoI pooling layers was also included between the final convolutional layer and the first fully connected layer in order to generate fixed-length vectors from each region proposal. The procedure can de divided into four phases. The first being the feeding of the whole image though the CNN to generate the feature map. The second phase consists of the use of selective search to find Regions of Interest (RoI). The third stage is the RoI pooling lauer to generate fixed-length vectors for all region proposals. The last stage is composed by the vectors being sent through the fully-connected layers for classification and localization through a softmax layer and a regression layer. This method improved detection quality and reduced computational, even though it is still relied on selective search for RoI.

Faster RCNN The improvement in speed and accuracy seen in Fast RCNN [32] could still be further enhanced by resolving its slective search method to find RoIs. This defficiency was addressed with Faster RCNN by changing selective search method with a Region Proposal Network (RPN). The process of Faster RCNN consists of the following. First, the image is fed through the CNN and a feature map is made. RPN is used on the feature map to result in Regions of Interest and scores for their objectness. After the RoIs are extracted, the RoI pooling layers scales them to a fixed-lenght to be fed into the fully connected layer. The feeding of said vectors into the fully connected layers is the last phase of the procedure. Softmax and regression layers are then used to output bounding boxes and the classification of objects.

Mask RCNN The aim of Mask RCNN [33] is to provided a pixel level segmentation. This means that instead of simply providing the classification and the bounding box, it also adds a third output, which is the object mask. Mask RCNN still uses the same RPN proposed for Faster RCNN and it also adds an RoIAlign layer such that the features extracted have object positions associated to it. This method accomplished state-of-the-art accuracy for instance segmentation. Figure 2.3 shows teh architecture of Mask RCNN, starting from RGB image input to final segmentation mask.

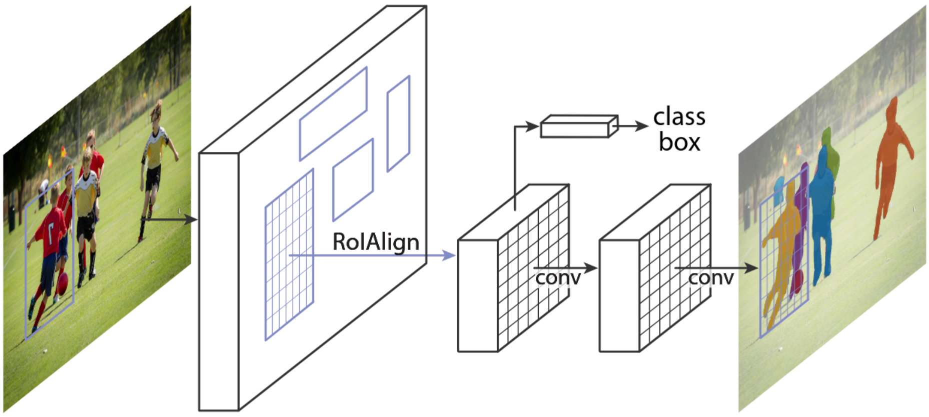


Figure 2.3: Mask RCNN architecture with RoI network and final segmentation mask. [33]

One-stage Object Detectors

As the computation time increases with two-stage object detection methods, region proposal free frameworks can be useful to detect objects. One-stage object detection methods locate and classify the objects in a simultaneous manner through Deep Convolutional Neural Networks (DCNNs). This approach to object detection requires a single pass through the network and it makes predictions for all bounding boxes at the same time. The image pixels are mapped to class probabilities and bounding box coordinates.

DetectorNet This method treats object detection as a regression problem, DetectorNet [34] is capable of learning features in order to classify objects and it can obtain geometric context. AlexNet is used as its backbone, where the softmax layer is replaced with a regression layer. The pixels are predicted, with the division of the image into a coarse grid. It is slow to train due to it being trained for each type of object.

OverFeat This method presents a single unified architecture with CNNs to locate, classify and detect objects through a multi-scale sliding window approach. This work as the first DCNN one-stage object detector to only need a single pass through its fully convolutional layers. [35]

YOLO This one-stage detector, You Only Look Once, or YOLO [36], implements a regression problem approach to the task of detecting objects. YOLO makes predictions of the objects' bounding boxes and probability score to the object it is associated with. YOLO is able to predict the object detection with a limited portion of candidate regions. It is though different from region based methods as it is not using features from a specific region only but from the complete image. YOLO divides images into grid as S x S size and each grid cell is responsible for the object detection within themselves. Each grid contains bounding boxes and for each bounding box 5 values are stored, x, y, w, h and confidence score. The confidence score is the object class probability and the IoU (Intersection over Union) between the bounding box and the ground truth. YOLO also utilizes of Non-Max Suppression to remove overlapping bounding boxes.

SSD Single-shot detector [37] is a unified detector, faster than the aforementioned YOLO, while still achieving significant accuracy. It uses regression and anchors in its algorithm. SSD has low computational requirements, due to its regression approach. The anchor procedure allows for a robust feature extraction for different sizes. SSD also implements a feed-forward to generate bounding boxes, which have defined dimensions, and their objectness scores for the object classes. Non-maximum suppression is also applied to reduce overlapping boxes. The use of anchor boxes allows for a great detection accuracy while still being fast.

Transformer-based Object Detection/Segmentation Models

The previously mentioned methods relied on deep learning approaches that were proposed before 2017. In 2017 the paper, "Attention Is All You Need" [38] proposed a new architecture named "Transformer". Since then, new models in the field of AI have been developed, also in the field of Computer Vision. Some of the object detection and segmentation models will now be presented.

GroundingDINO [39] This transformer-based detector combines object detection with natural language understanding to enable open-vocabulary detection. GroundingDINO uses a dual-encoder architecture where both image and text features are processed through separate encoders before being fused in a transformer decoder. The model can detect objects described in natural language prompts without requiring training on those specific categories. It divides the detection task into feature extraction, cross-modal fusion, and prediction phases. The image encoder processes visual features while the text encoder handles language descriptions, and the decoder performs cross-attention between modalities. This approach allows for zero-shot detection of novel objects by simply providing text descriptions, making it highly flexible for real-world applications where object categories may not be predefined.

SAM (Segment Anything Model) [40] This promptable segmentation model uses a Vision Transformer architecture to generate high-quality object masks from various input prompts. SAM consists of three main components: an image encoder based on Vision Transformer (ViT), a prompt encoder that handles different input types, and a lightweight mask decoder. The model can accept prompts in multiple forms including points, bounding boxes, or text descriptions to specify what should be segmented. SAM was trained on a massive dataset of over 1 billion masks, enabling it to generalize across diverse domains and object types. The architecture processes the entire image through the ViT encoder to create rich feature representations, then uses the prompt information to guide the mask generation process, producing precise segmentation masks even for objects it hasn't seen during training.

YOLO-World [41] This hybrid detector combines CNN-based YOLO architecture with transformer-based text processing to enable open-vocabulary object detection in real-time. YOLO-World uses a YOLOv8 backbone for image feature extraction while employing a CLIP text encoder to process natural language descriptions of target objects. The model integrates vision and language features through cross-modal attention mechanisms, allowing it to detect objects described in text prompts without specific training on those categories. Unlike fully transformer-based approaches, YOLO-World maintains the efficiency of CNN-based detection while adding language understanding capabilities. The architecture processes images through the CNN backbone to extract visual features, encodes text prompts through the transformer-based text encoder, and fuses these modalities to predict object locations and classifications. This design enables zero-shot detection

Evaluation Metrics

In order to determine the quality of object detection metrics are used. These metrics are, Precision, Recall, IoU, Average Precision (AP) and also Mean Average Precision, (mAP). Object detection can be described by three main elements, an object class, the bounding box around the object and its confidence score, where the more accurate bounding box and the higher confidence values, the better the object detector is at detecting the class objects.
IoU - Intersection over Union When making a prediction IoU can be used to identify the ratio of the predicted area with the total area of both prediction and ground truth. In this way, IoU can be defined as the area overlapping between the predicted bounding box and the ground truth divided by the total area of prediction and ground truth. In the case of a perfect prediction the IoU is 1, and it can only decrease from there. In object detection IoU is used a detection as truthful or not, and to do so the following terms will be established.

- **TP** - True Positive - The prediction is positive and it is ture.
- **TN** - True Negative - The prediction is negative and it is indeed negative.
- **FP** - False Positive - The prediction is positive and it is actually negative
- **FN** - False Negative - The prediction is negative and is instead positive

Accuracy A metric that incorporates all classifications, is accuracy. This takes the number of total true predictions and divides it to the total number of predictions made, both false and true. The equation for accuracy can be seen bellow in equation 2.1:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \tag{2.1}$$

Precision This metric describes how much of the positives predicted, how many of them are actually true. This can be seen in equation 2.2.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.2}$$

It is important to note that precision does not show the whole iamge, as it is possible to make few predictions and be precise while still lacking to detect all of the objects.

Recall A complementary metric to precision is recall, and it is described as the ratio between the number of positive predictions and the total number of positives. This is to say that if there are 5 objects in the image and all of the are identified, the recall value would be one, however if more identifications were made the recall would still be one, as to say a total 15 identifications were made and the 5 true ones are a part of the 15, then the recall would be 1, whereas the precision would not be 1, since both look at different aspects of the identification.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.3}$$

Average Precision One of the most important metrics, if not the most important, is the average precision. This metric is based on the area under the curve for the precision and recall plots. Where a bigger area indicates good recall and precision. To achieve this metric the condifence scores are ordered from highest to lowest of the avialble predictions. Then an IoU threshold is applied to determine the predicitons as TP or FP, and then for each prediction the cumulative precision and recall are evaluated, since it is accumulative recall may increase and precision may decrease along the way, this is simply due to the increase in sample count and its relation to precision and recall. After these cumulative precision and recall points are calculated for the predictions, they are plotted. Then interpolation methods can be used calculate the area under this curve, which is the Average Precision. It can be seen in equation 2.4, the calculation used for AP, where P_{interp} is the maximum precision value for all present and future iterations, and the $R_i - R_{i-1}$ are the present and the previous recall values, respectively.[42]

$$AP = \sum_{i=1}^{n} (R_i - R_{i-1}) P_{interp} \tag{2.4}$$

F1.-score This metric also take into consideration precision and recall, however it is more simplistic than AP in its calculation. This is the harmonic mean of precision and recall, and therefore can show if those are balanced or not. As bigger disparity will lead to a lower F1-score, and if the values are equal to each other then F1 is simply that value. The equation for F1-score cajn be seen in 2.5.

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{2.5}$$

2.3 6D Object Pose Estimation

As seen in the depalletizing tasks, 6D pose estimation is a crucial part of the procedure's pipeline in order to enable accurate grasping. Object pose estimation has been a topic of research and there are several proposed method and they can be divided into two main categories based on whether a CAD model of the object is needed or not. Instance-Level Pose Estimation is the category when CAD models are used and Category-Level Pose Estimation is where a 3D model is not present. It is important to note that a full literature review is outside of the scope, which results in some of the methods not being evaluated on this project. The evaluation of 6D Object Pose estimation will be based on the works of [43].

Instance-Level Methods for 6D Object Pose Estimation

As for this type of 6D pose estimation, there are three categories based on the sensory information used. These are RGB-, Point Cloud-, and RGB-D-Based methods.

RGB-Based Methods

The progression of deep learning has enabled the advancement of RGB methods used for 6D object pose estimation. As RGB images are composed if significant visual input, as features and texture, methods have been able to learn how objects are oriented on space based on that. Figure 2.4 shows the pipeline for RGB methods in instance 6D pose estimation.

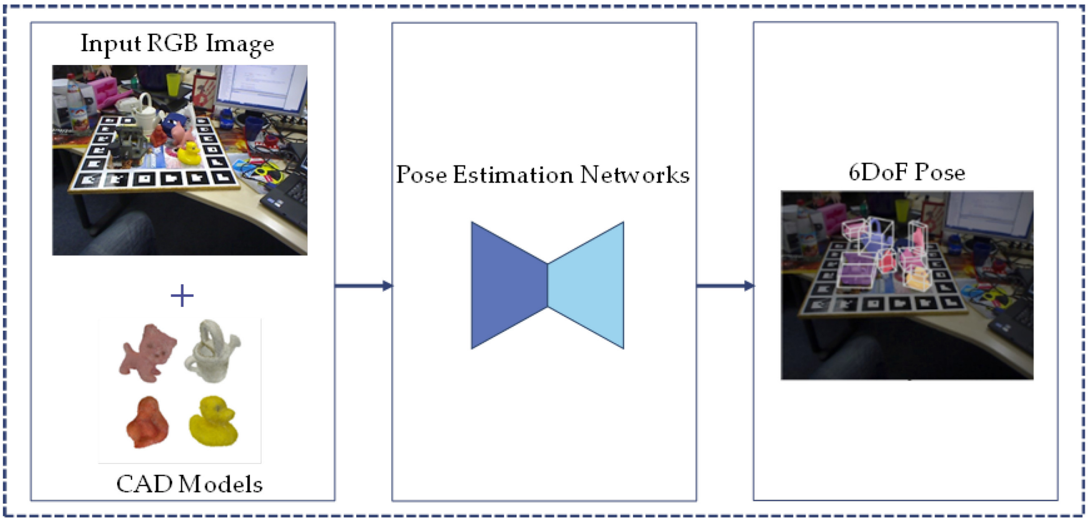


Figure 2.4: RGB methods that require both an image and the CAD model of the objects tod detect their poses. [43]

The challenges present in these methods are mainly occlusion and insufficient texture from the objects.

Regression-Based Methods One of the simplest approaches is the use of regression to determine the pose of the object from solely the RGB image and no keypoint representation. One of the first approaches in end-to-end methods for 6D pose estimation is PoseNet [44]. This approach though only used for human pose estimation showed the possibility of using deep learning to estimate the pose of object by only the use RGB images.
The methods based on regression have achieved good results in 6D object pose estimation. There are though some shortages when handling more difficult conditions, like occluded objects, different illumination levels, its real-time capabilities, and also the generalization to more objects and scenes.

Template-Based Methods This approach to 6D object pose estimation relies usually on finding the best fit for the template. The search is made with several templates with their true poses albeled and then making the 6D pose estimation.
This class of methods have positive aspects such as low complexity, speed, and flexibility when the objects have modifications in their appearance. Lack of texture in objects is neither a significant challenge for template-base approaches. These methods have challenges when presented to more complicated and occluded scenes, different levels of illumination, or few object features. It is clear that local representation is necessary for accurate template-based methods, since when global representations are used, the background can degrade the quality of the estimation, specially for unseen objects, which demands, thus, the use of local representation to increase its accuracy.
Feature-Based Methods This category of approaches is very popular in 6D object pose estimation. The main process is revolved around feature extraction from the target object in RGB image, then matching these features with the 3D model, after that the matching of 2D to 3D points using the Perspective-n-Point, PnP algorithm. The PnP procedure makes the correspondence of 3D points into 2D, it also demonstrates the camera pose estimation in case of known 3D points. The 2D to 3D feature matching, the 2D to 3D relationship can be determined and the pose of the object precisely estimated.

Refinement Methods This class of methods rely on the refinement of the pose estimation through techniques such as ICP or deep refinement networks, which make estimation more accurate. The difference between the rendered image and the input image is reduced with this approach. In case of occlusions, the process of refinement can be significantly useful. There is though a high computation requirement associated to it.

Point Cloud or Depth Methods

These types of methods can be particularly useful in cases where the RGB data is not adequate enough to make a precise pose estimation, and the point cloud, or depth map, can be used to provide geometric information.

Point Cloud Methods

This technique uses the extraction of global and local features that can be used to represent the shape and geometry of the target object. One of the first works using point clouds for 6D object pose estimation is PointNet. The method used symmetric aggregation functions in order to handle the variances in permutation. Based on this work, PointNet++ was developed which improved the process of localized data in point cloud.

Depth based Methdos This category of methods usually relies on the conversion of depth images into point clouds to then make a pose estimation of the objects in the image. SinDePose is a work that is based on the SwinTrasnformer, and it combines said transformer and pose estimation it is able to precisely do the estimation by the use of point cloud data and vector information from the depth map. IT can also estimate the position of occluded objects. OVE6D is another method that was trained solely on synthetic data. It requires a depth map and a segmentation mask to estimate the pose of objects. The pipeline is divided into three stages, viewpoint, the in-plane rotation and finally the object's translation. Point cloud or depth based methods may face issues in the case of reflection on object surfaces due to sensor information being affected. There are still not many methods based on point cloud or depth information only, however they can be used in other methods such as PVNet that has its approach based on PointNet.

RGB-D Methods

In the case of only using RGB information, the background, cluttering and lighting can pose a significant challenge to its accurate and reliable 6D pose estimation. The additional depth data can be used to achieve more accurate estimates due to the extra geometric information on the object. It is though a difficult task to effectively combine both RGB and depth information. Some of the earlier methods relied on processing RGB and depth in a separate manner. RGB-D based methods can be dived into three main categories, fusion, keypoints based methods and other. Figure 2.5 shows the general pipeline for RGB-D methods in pose estimation.

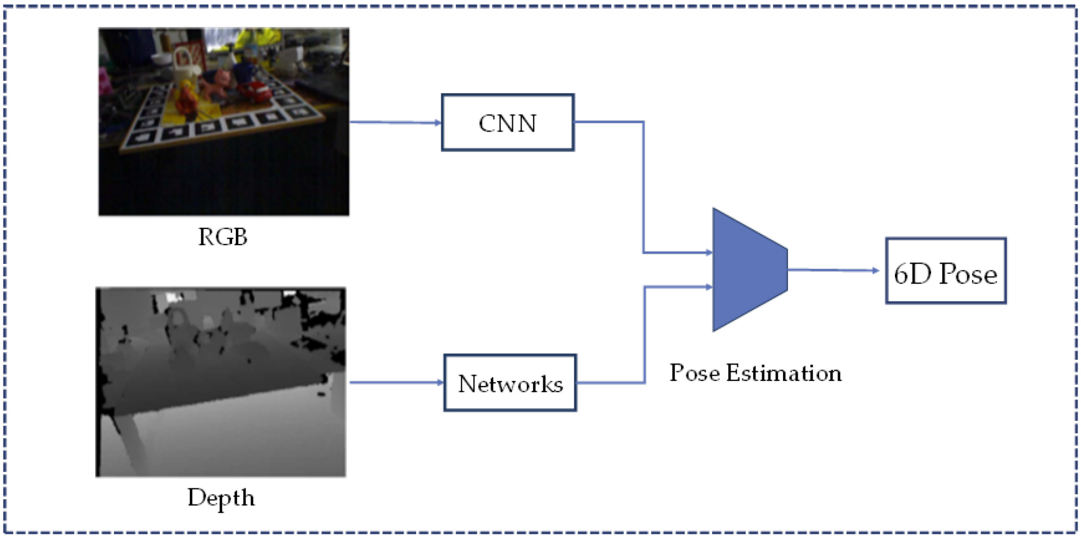


Figure 2.5: Pipeline for RGB-D methods in 6D pose estimation, highlighting separated processes for depth and RGB. [43]

Fusion Methods

Dense Fusion is a method that makes use of distinct feature extraction and dense fusion of RGB and depth data. Each pixel has its pose estimated and the end result is achieved by voting. The leveraging of structural data of its depth information enables precise pose estimation. The downside of the method is that it needs high-quality depth information and significant computational power.

In robotics, MoreFusion has been used to estimate the pose of known objects. This approach combines the masks produced by segmentation in a volumetric space in order to divided occupied and free space. Peripheral context, and voxel reconstruction are taken into account by this method, and it has shown better performance in occluded scenes when compared to DenseFsion.

Keypoint Methods

The use of keypoints is present as part of the RGB-D class of pose estimation methods. The search of keypoints with the use correspondences present in the object enables the pose estimation. Point Fusion is a method that uses PointNet as base, but it is not robust in finding local features in the point cloud. PVN3D handles this issue with its two-phase method. It has several modules in it: feature extraction, keypoint finding, semantic segmentation, and lastly centroid voting module. The interaction between these modules enable the finding of keypoints. After said keypoints are found, the method uses least squares to find a solution.

This subclass of methods can be used even in relatively cluttered scenes with enough precision, however it is dependable on the detection of its keypoints.

Category-level Methods for 6D Object Pose Estimation

This type of pose estimation intends to estimate the translation, rotation and dimension of objects in a scene, without the use of a prior knowledge from the objects 3D models, as it is not always available for all applications.

Regression Methods

One of the most simple approaches is the regression one, where typically RGB and depth are used to estimate the object's pose. One of the biggest difficulties in category-level pose estimation is the lack of ground truth data. NOCS resolves this problem by using a context-aware mixed reality method, and is one of the first of its kind. NOCS stands for Normalized Object Coordinate Space and the authors of the method published a dataset, which became reference in the field, namely the NOCS dataset. In

order to handle objects with symmetry, the approach makes use of axis of symmetry in its training data. This method has robust results for unseen objects in real-life environments.
DualPoseNet makes use of a implicit and explicit decoder to predict object poses. The structure enables supervision during its training phase for consistent predictions for both encoders. The bounding boxes predicted are compact and the poses are accurately estimated wit this method. In general regression methods are more simple, though they have problems change.

Prior-Based methods
The uses of prior knowledge can result in more accurate pose estimation in category-level tasks. One of the most known approaches is SPF, which implemented a prior-based framework to handle variance within a class of objects, and it has become a popular approach. ACR-Pose is a method that was published after SPD, it uses an adversarial training scheme to improve the reconstruction of canonical representations, making the pose estimation more precise, which is heavily seen in the intra class objects. This approach is though susceptible to degradation in pose estimation when occlusion is presence.

Other Methods
There are also other methods that do not fall necessarily within the two aforementioned categories. Works on this category have been done, for example, to apply a single-stage approach using RGB-D information, to jointly make prediction on the target object's shape and pose. The use of semantic primitives in the generative model is able to retrieve information about the object's shape and also connect the point cloud with implicitly generated representations. The work uses a SIM(3) invariant descriptor and has shown promising results.

Another work that does not require CAD models to find the object's pose makes use of synthetizing images from different viewpoints and utilizing generative adversarial network. This approach makes use of gradient-based fitting procedure with parametric neural image synthesis module that enables the appearance,shape and pose of an object to be estimated.

OnePose is a work that only relies on RGB images to reconstruct object representations, however s ingle image is not enough, but a video scan is necessary to estimate the object's pose. It relies in local features and its performance can be degraded for objects without sufficient texture.

Another relevant method is RANSAC, specially in depalletizing, as seen in the other depalletizing frameworks. However, RANSAC does not fall into the deep learning category like the other methods presented in this section, as it uses 3D points to calculate the plane equation and fit points that are inliers to such planes. This method can be effective when used with packages as they have cuboid shape and their faces can be mathematically described as planes in 3D space. RANSAC is also low computationally in comparison to deep learning methods.

2.4 Human-Robot Interaction

In the recent years, the collaboration between humans and robots has been more studied and its importance has only increased. With the increase of collaborative robots, and the advancements of transformer-based models like Large Language Models, also known as LLMs the possibility of interaction between human and robot has made available in an efficient and simple manner. This collaboration between humans and robots is only possible when certain requirements are taken into consideration. The interaction needs to be safe, trustful and transparent. [45]

Safety-layered architectures have been implemented in systems where HRI is crucial . This approach enables safety certification of human-friendly robots across diverse HRI scenarios, from industrial manufacturing environments to domestic care settings. The challenge lies in balancing robotic capabilities with safety requirements while maintaining natural and intuitive interaction patterns. Figure 2.6 shows the different layers of safety in HRI. For example, impact safety ensures that workers or people surrounded the robots do not get hit and collisions are avoided. Maximum force allowed is also a parameter in this situation. Another relevant point is psychological safety, which is the user's perceived safety when interacting with the robot. Minimization of fear and stress can be done with eliminating sudden movements and lowering joint velocities for helping the user/operator. [45] The integration of LLMs can enable robots to cooperate with humans in an intelligent manner. The use of prompts and and significant data can be used to bridge the gap between humans and robots. One example, seen in a study made in [46], illustrates how LLMs can use relevant information to interact with users. The study was made based on a hospital case, however the approach can be used into different use cases, as depalletizing. In the study's proposed approach, the LLMs would have information about what the robot's states and also the relevant user information, while also being able to discern static and dynamical knowledge. In this case the user would be able to ask personalised questions and the LLM, when taking all information about the user and the robot could engage and provide useful information to the user [46]

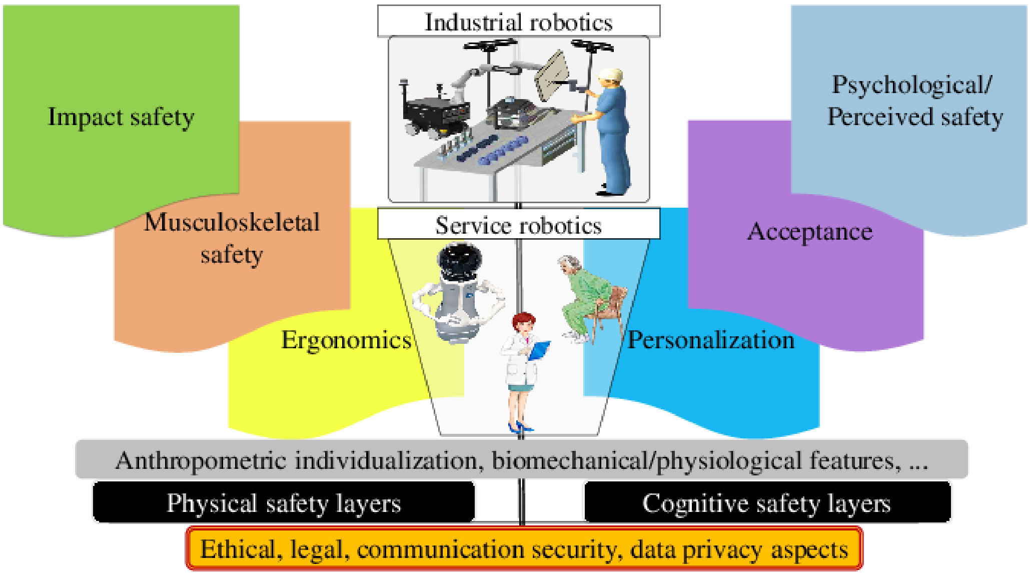


Figure 2.6: Safety overview in HRI, where both physical and psychological safety are taken into account both for industrial and service robots. [45]

This approach of integrating information about the robot and the user could also be applied in a depalletizing operation. If an operator that is controlling the process notices, for instance, that a robot is not operating, then it could ask the LLM helper why that is the case and the LLM could retrieve relevant information such as the camera images used to identify the parcels. It could then, in the case of insufficient lighting, and data from the vision system can not identify more boxes, conclude that the lighting is bad and is the reason why the operation is not continuing, which could be of crucial importance for an operator. LLMs could also be used to enable prompt based operations if the operator wants to pick a specific object, by giving a command to the LLM, promptable vision systems could then use the prompt and the camera images to perform the operation instructed by the operator.

3 System Design

3.1 Use Case

The task of depalletizing needs to be established in order to design a solution capable of completing said task. The use case is of an operator in an industrial environment that wants to transport boxes from one place to the other, for example, from one pallet to another location. There are some conditions that are assumed in this use case. A camera is used to detect and estimate the pose of the boxes. This camera is static and external from the robot. The robot and the gripper are responsible for transporting the boxes from one place to another. The boxes to be detected should be in the FOV of the camera and the placement are position is known beforehand. The operator is able to communicate to the robot via a text command to initiate the operation.

3.2 Hardware

All hardware components were provided by Aalborg University and the Robotics and Automation Group. Therefore all development and testing of the system shall be performed at Aalborg University. The system hardware consists of a:

- **UR10** robot, from Univesal Robots
- **VG10** OnRobot vacuum gripper
- Intel RealSense **D455** RGBD camera

Figure 3.1: UR10 cobot from Universal Robots. [47]

The UR10 cobot from Universal Robots can be seen in figure 3.1, and its relevant specifications in table 3.1. Its payload of 10kg and reach of 1300 mm allow for enough lifting capacity as well as a sufficient range for depalletizing operations. It is also a collaborative robot so human-robot interaction can be allowed with the help of the robot.

Parameter	**Specification**
Degrees of Freedom	6 rotating joints
Payload	10 kg / 22 lbs
Reach	1300 mm / 51.2 in
Joint ranges	±360°
Speed	Base and Shoulder: 120°/s Elbow, Wrist 1, 2, 3: 180°/s
Repeatability	±0.1 mm / ±0.0039 in
Weight	28.9 kg / 63.7 lbs
Footprint	∅190 mm / 7.5 in
Temperature range	0-50°C
Power consumption	Typical 250W (Min 90W, Max 500W)
IP classification	IP54

Table 3.1: UR10 Robot Specifications. [47]

The end-effector available for the project is the VG10 vacuum gripper from OnRobot. It also has a recommended payload of 10 kg which is equal to the UR10. It is possible to control the vacuum gripper through TCP Modbus, and it requires the installation of the OnRobot URCap on the UR10's teach pendant for integration through "External Control". It is possible to regulate the the air pressure up to 80 KPa according to the user's desire. This allows for the handling of boxes in a depalletizing operation.

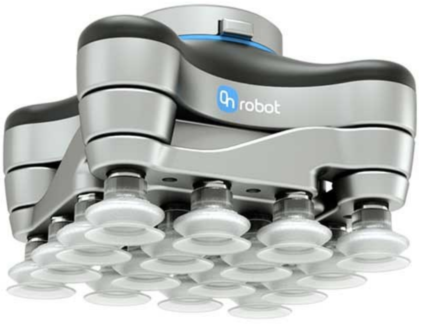


Figure 3.2: VG10 vacuum gripper from OnRobot. [48]

The specifications of the VG10 vacuum gripper can be seen in table 3.2, and an image of the gripper is present in figure 3.2. It is important to note the gripper's dimensions and its possibility of unfolding if needed, however this unfolding is simply mechanical so there is n possibility of adapting the gripper's dimensions dependent on the object at hand. Furthermore, gripping time and releasing time are important aspects when building a system around a suction gripper and must not be forgotten for ideal grasping.

Parameter	**Specification**
Payload	Rated: 10 kg / 22 lbs Maximum: 15 kg / 33 lbs
Weight	1.62 kg / 3.57 lbs
Dimensions (folded)	105 x 146 x 146 mm
Dimensions (unfolded)	105 x 390 x 390 mm
Vacuum	5-80%
Air flow	0-12 L/min
Gripping time	0.35 s (typical)
Releasing time	0.20 s (typical)
Power supply	24V (20.4-28.8V)
IP classification	IP54
Operating temperature	0-50°C

Table 3.2: OnRobot VG10 (v2) Vacuum Gripper Specifications. [48]

The camera available for the depalletizing operation is the Intel RealSense D455 seen in figure 3.3. The RealSense family is oen of the most used in robotics and the integration in a robotic system is simple with the libraries and packages provided by Intel.

Figure 3.3: Intel RealSense D455 RGBD Camera.[49]

The camera has both RGB and depth information and can be used indoors and outdoors. It is important to note the minimum range , the ideal range and the depth accuracy, as these are crucial in robotic operations that require precise handling as in depalletizing. The depth and RGB FOVs are wide and provide great information of the environment. More specifications can be seen in table 3.3.

Parameter	**Specification**
Use environment	Indoor/Outdoor
Ideal range	0.6 m to 6 m
Minimum depth distance	~52 cm
Depth accuracy	<2% at 4 m
Depth field of view	87° × 58°
Depth output resolution	Up to 1280 × 720
Depth frame rate	Up to 90 fps
RGB resolution	Up to 1280 × 800
RGB frame rate	30 fps
RGB sensor FOV	90° × 65°
Dimensions	124 mm × 26 mm × 29 mm
Connectors	USB-C 3.1 Gen 1
Mounting mechanism	One 1/4-20 UNC thread mounting point Two M4 thread mounting points Tripod

Table 3.3: Intel RealSense D455 Camera Specifications. [50]

3.3 System Architecture

The robot is controlled through the MoveIt2 framework in ROS2. In order to enable this control, an URCap was installed on the teach pendant to allow for external control. All of the hardware components are connected through a ROS2 system.

The system design can be seen in figure 3.4. At the highest level, there is the user layer, in this case an operator at a company that wants to depalletize simply would send a text command, as ”Please pick up the boxes” and then the operation would start.

From the operator command in the user layer, the connection to the control layer is made through the Large Language Models, or LLMs. In this layer the control of the robot is made by processing the user’s command. The four subsystems used are the LLMs, the image segmentation, the pose estimation and the stack optimisation. Each of these subsystems have a purpose. As for the LLMs, the main responsibility would be to process the user’s command into something that could be used by an image segmentation method, for example an object: ”box”. The image segmentation system should be able to segment the desired object from an image based on the user’s command. The segmentation would then be used to locate the object in the image and use that information to estimate its pose since the RGBD camera allows for the location of the object in 3D space. After the object is found and its dimensions are known, the the stack optimisation system would determine the optimal position to the box on the pallet according to an objective function.
Next is the Hardware API layer, which is simply composed of ROS2. ROS2, which stands for Robot Operating System 2, allows for hardware integration in robotic applications. ROS2 is to be used to communciate with the hardware: UR10 robot, VG10 vacuum gripper and Intel RealSense D455 RGBD camera.
The lowest level of the system architecture is the Hardware layer. It is composed of the robot, UR10 from Universal Robots, the VG10 vacuum gripper from OnRobot which use ethernet to communicate

with the deployed computer and the Intel RealSense D455 RGBD camera that uses USB protocol with the deployed computer.

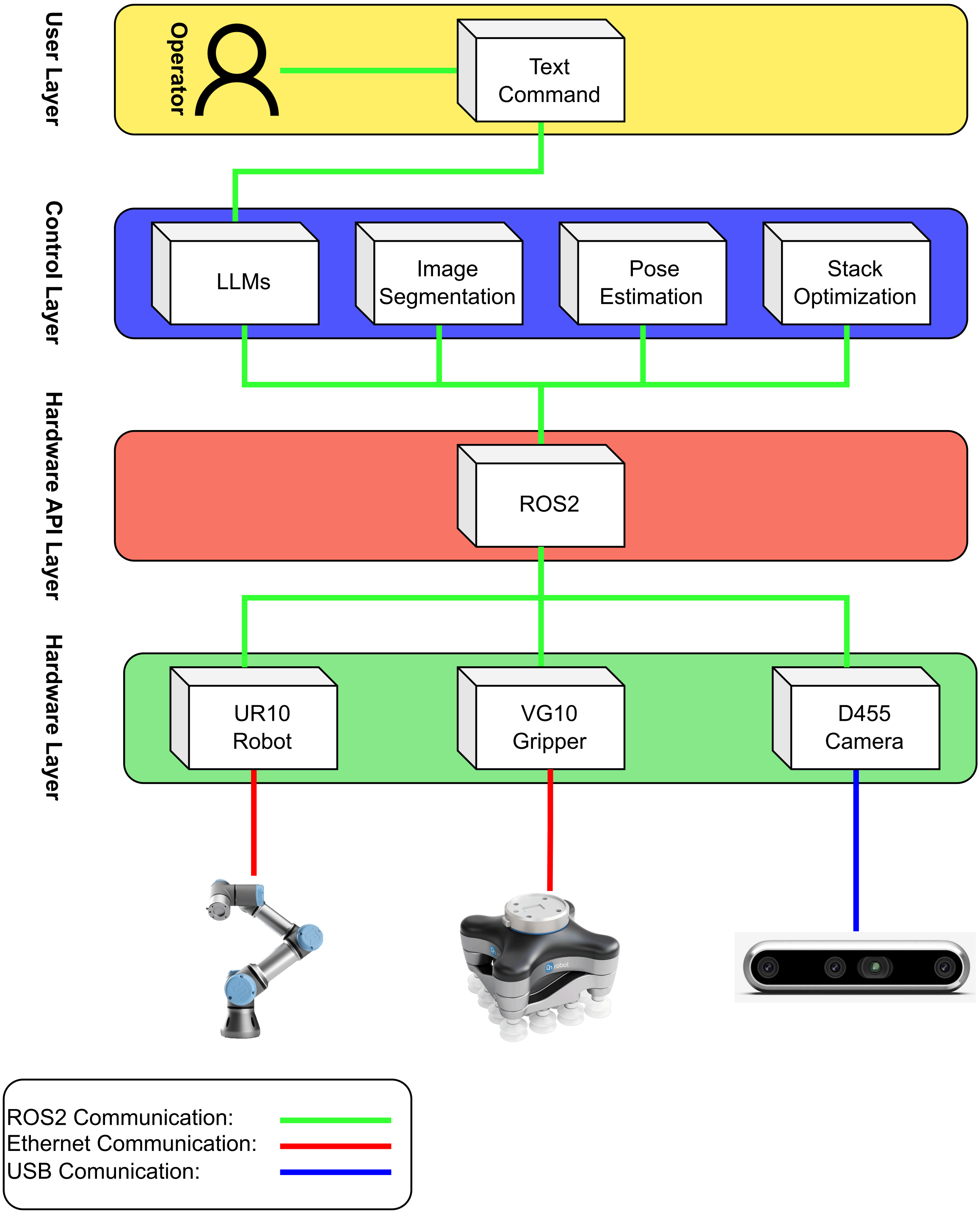


Figure 3.4: System architecture divided into layers. At the highest the user layer where the operator sends a command to the robot and then in order to accomplish the task assigned by the operator, then control layer receives the request and process it and control the hardware necessary to accomplish the task wih the help of ROS2.

Network Setup

There are five main elements present in the network setup of the solution. The deployed computer is connected to both a network switch and the Intel RealSense camera. The network switch makes it possible to connect both with the VG10 vacuum gripper and the UR10, it is important to note that in figure 3.5, the connection appears to be ”directly” to the hardware, however the Ethernet is connected to the control boxes of the VG10 and UR10 respectively. The deployed has a static address of 192.168.1.27, whereas the UR10 and the VG10 have respectively: 192.168.1.2, 192.168.1.1. The communication to the UR10 is done with the ”External Control” URCap that needs to run when deploying the robot. The VG10 vacuum gripper is connected with help of Modbus, where registers 1, 258 are used to set and read the vacuum levels. The Intel RealSense D455 camera is connected through a USB cable and the deployed computer has the Intel RealSense ROS2 wrapper which enables the interface.

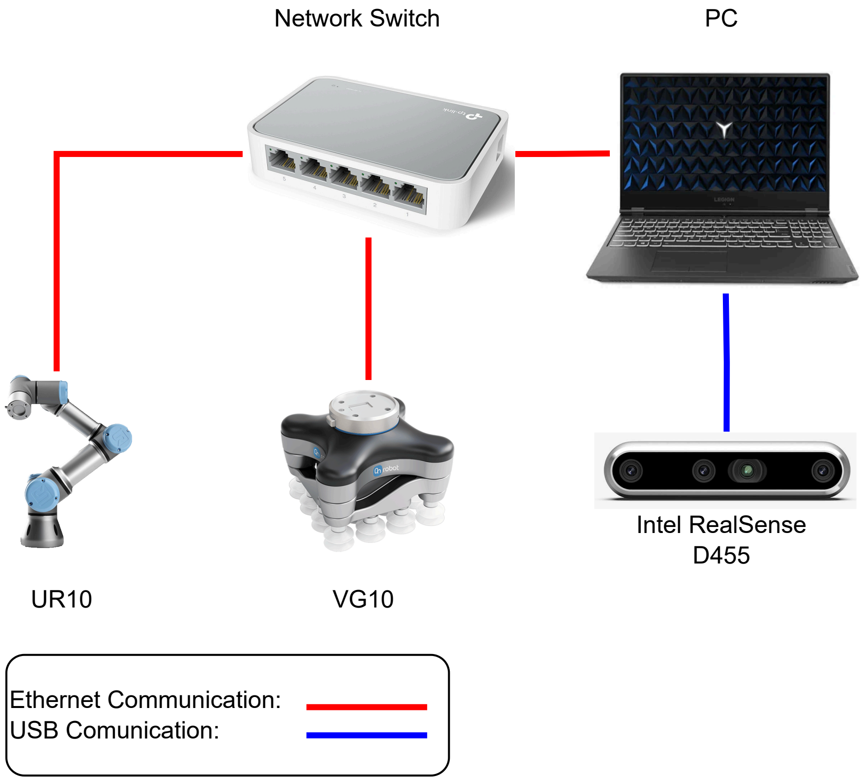


Figure 3.5: Network Setup with all the connections present in the robotic system. Switch allows for interaction between UR10, VG10 and PC and Intel camera is connected to the deployed PC via USB.

Control Layer

The system has to be able to process the operator's request as specified in the use case. In order automate a depalletizing operation, intelligent behaviour needs to be achieved. In figure 3.6, the four main modules, that will allow the robot to operate in an effective manner, are shown.

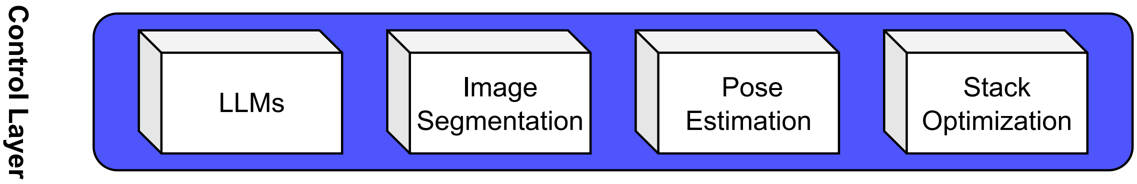
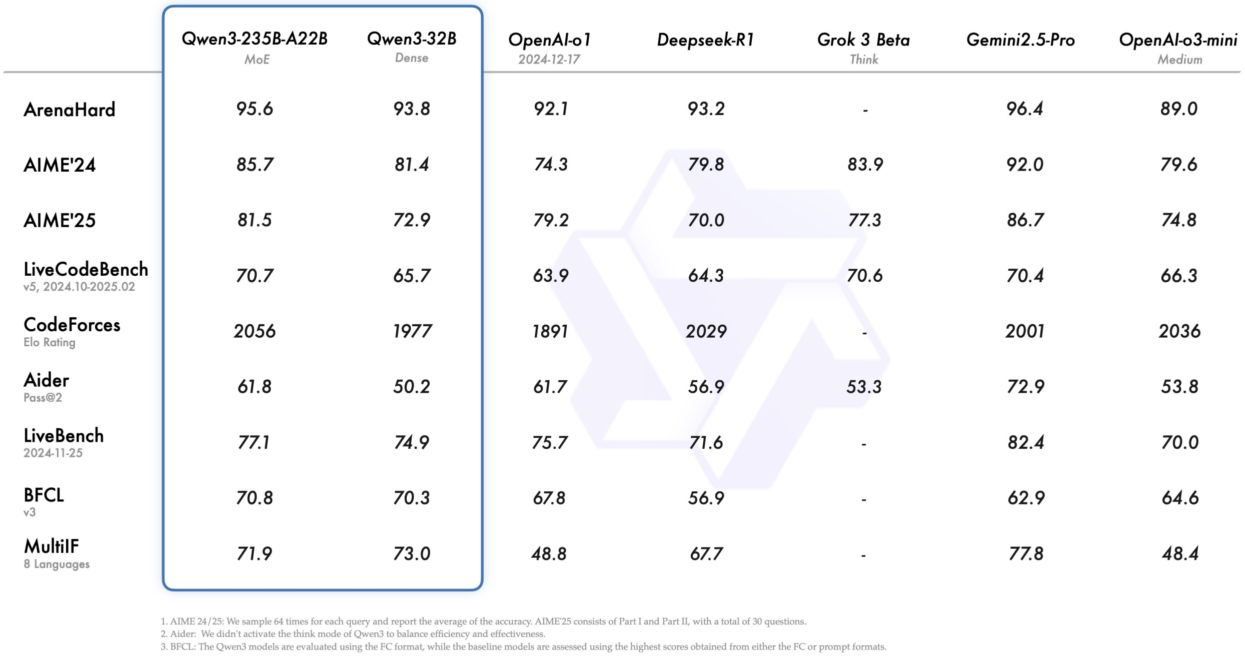


Figure 3.6: Control Layer for system architecture with four main modules: LLMs, Image Segmentation, Pose Estimation and Stack Optimization.

The modules are: LLMs, Image Segmentation, Pose Estimation and Stack Optimization. These will be now explained.

LLMs

It is necessary to understand the operator's command and extract the essential information necessary to continue with the operation. This creates the need to a close Human-Robot Interaction, which can be achieved through Large Language Models. Qwen3 from Alibaba has been chosen as the LLM for the depalletizing operation. It is an open-source model with state-of-the-art performance that can be run locally on a machine, therefore there is no need for an external third-party server if the user wants to maintain his data private. A benchmark of Qwen3 can be seen in figure 3.7, where it is shown that the model is capable of achieving similar performance to OpenAI's o1 model, Google's Gemini 2.5 pro and Deepseek R-1.



	Qwen3-235B-A22B MoE	Qwen3-32B Dense	OpenAI-o1 2024-12-17	Deepseek-R1	Grok 3 Beta Think	Gemini2.5-Pro	OpenAI-o3-mini Medium
ArenaHard	95.6	93.8	92.1	93.2	-	96.4	89.0
AIME'24	85.7	81.4	74.3	79.8	83.9	92.0	79.6
AIME'25	81.5	72.9	79.2	70.0	77.3	86.7	74.8
LiveCodeBench v5, 2024.10-2025.02	70.7	65.7	63.9	64.3	70.6	70.4	66.3
CodeForces Elo Rating	2056	1977	1891	2029	-	2001	2036
Aider Pass@2	61.8	50.2	61.7	56.9	53.3	72.9	53.8
LiveBench 2024-11-25	77.1	74.9	75.7	71.6	-	82.4	70.0
BFCL v3	70.8	70.3	67.8	56.9	-	62.9	64.6
MultiIF 8 Languages	71.9	73.0	48.8	67.7	-	77.8	48.4

1. AIME 24/25: We sample 64 times for each query and report the average of the accuracy. AIME'25 consists of Part I and Part II, with a total of 30 questions.
2. Aider: We didn't activate the think mode of Qwen3 to balance efficiency and effectiveness.
3. BFCL: The Qwen3 models are evaluated using the FC format, while the baseline models are assessed using the highest scores obtained from either the FC or prompt formats.

Figure 3.7: Benchmark comparison of Qwen3 against other leading LLMs, illustrating its competitive performance across various evaluation metrics. [51]

Image Segmentation

As the system uses a RGB-D camera, the RGB data can be used to segment the objects in the image. It is also needed to segment the objects according to the extracted object from the LLMs. The chosen method to achieve this functionality is the combination of GroudingDINO and SAM2. GroundingDINO is an object detector that allows for text prompt and it yields a bounding box. This bounding box can then be given to SAM and the segmentation can be done. This is great as it allows for greater interaction between human and robot, since the human can describe the object needed to be picked and the segmentation can be done without the need of fine-tuning.

Pose Estimation

The pose estimation method used is RANSAC, as it is assumed the boxes are approximately flat (approximately cuboids). The grasping using the vacuum gripper will be performed form the top face of the box, the plane will then be estimated using RANSAC and the rectangle fitting will be used to find the four corners and consequently the center of the top face. RANSAC is a good alternative as it is low in computational power and does not need any CAD model and eliminates the need of a deep learning approach to find the pose of the box.

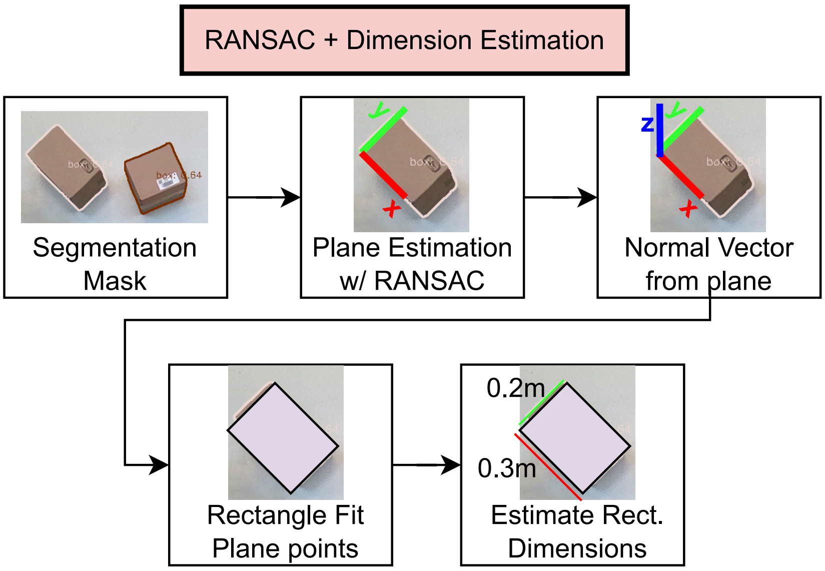

RANSAC + Dimension Estimation
Segmentation Mask
Plane Estimation w/ RANSAC
Normal Vector from plane
Rectangle Fit Plane points
0.2m
0.3m
Estimate Rect. Dimensions
x
y
z

Figure 3.8: RANSAC and Dimension Estimation pipeline based on segmentation from SAM with GroundingDINO.

The pose estimation pipeline can be seen in figure 3.8. The pipeline is dependent on a segmentation mask from SAM with GroundingDINO, from which the correspondent depth values are used to find the plane and then the normal vector from that plane. The points from the top plane are then used to perform rectangle fitting and the corners are estimated. The center point of the rectangle is the point of contact given to the robot and the width and length of the box are used to the stack optimiser that finds the optimal box placement.

Stack Optimiser

The box dimensions estimated in the pose estimation are to be used in order to calculate the optimal placement for the boxes. The ideal of optimality will be explained with an objective function that takes into account four main aspects. These elements can be seen in figure 3.9, and hey are the boxes' dimensions, that need to fit into the pallet and minimise the height of the stack. It is also important to take into account safety margins and depth information might not be perfect and finally, it is crucial to have support constraint to ensure boxes that stacked on top of others with enough support to not fall and destroy the stack and make the boxes go out of the pallet's boundaries.

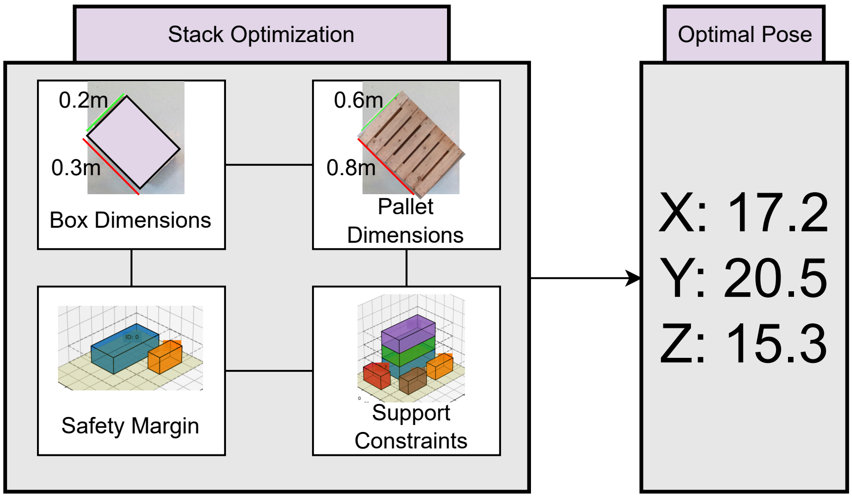


Figure 3.9: Overview of the Stack Optimisation module, illustrating the key inputs (Box Dimensions, Pallet Dimensions) and constraints (Safety Margin, Support Constraints) considered to determine the Optimal Pose for placing a box.

Based on all of the aforementioned elements, the optimiser should find the optimal pose for each given box. This optimal pose will then be set as a goal pose for the robot in order to stack the box accordingly.

General Pipeline

After the explanation of the Control Layer, a general description of the operation will be now be done. The first step in the pipeline from figure 3.10 is the operator command. This command is the trigger for the operation to start, the LLMs which receive this command are then used to specify the object description. In the case of the operator saying: "pick up the boxes", the LLMs should extract box, since the object box needs to be segmented by SAM with GroundingDINO. The segmentation provided is then forwarded to the Pose Estimation part of the system. It utilises RANSAC to find the box's top plane and also estimates the box's dimensions based on rectangle fitting.

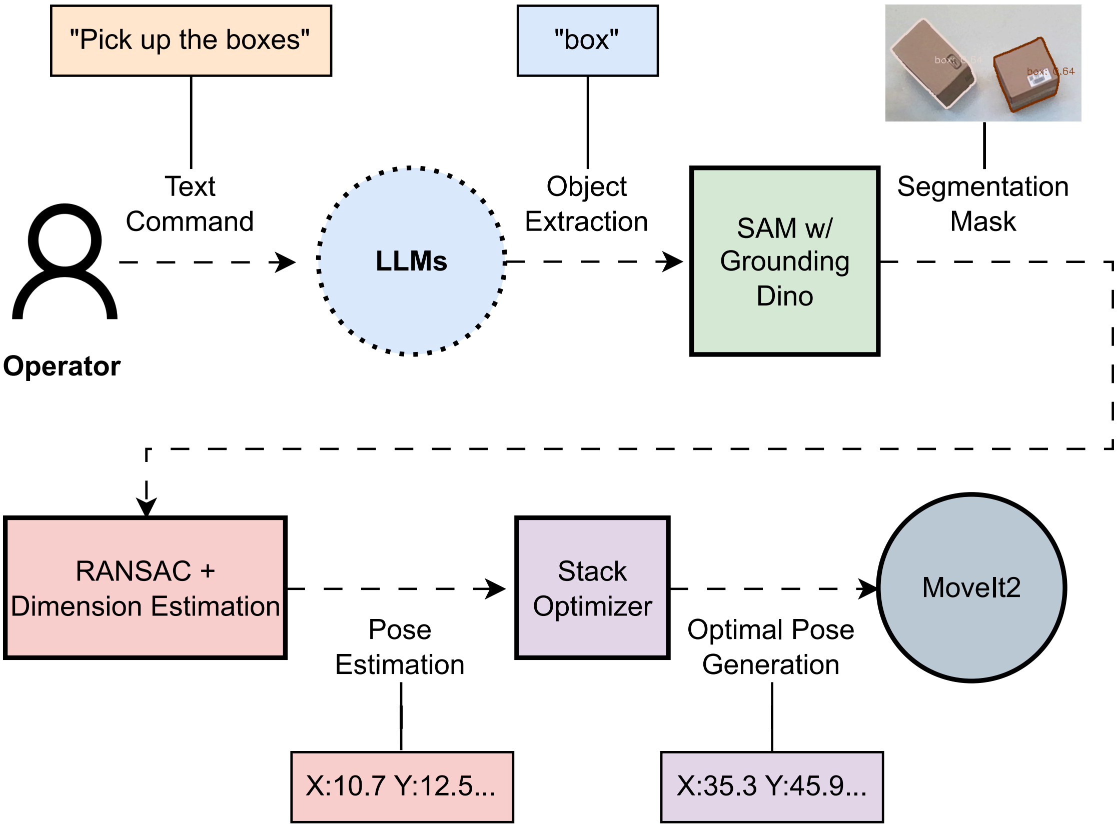


Figure 3.10: General operational pipeline of the Smart Vision-Guided Robotic Depalletizing System, from operator text command processing by LLMs, through image segmentation and pose estimation, to optimal pose generation for robotic pick and place using MoveIt2

After the dimensions are estimated and the pose estimation is calculated, the estimated dimensions should then be snt to the stack optimiser. The stack optimiser is responsible for generating the box's optimal pose on the pallet that the box will be placed. Both the estimated pose of the box and the optimal pose for the placement will be used in the MoveIt2 framework to make the robot pick and then place the box.

4 Implementation

4.1 Intrinsic Camera Calibration

In order to use depth information from the RGB-D camera, it is necessary to have the intrinsic parameters from the camera available. This can be done through intrinsic camera calibration. First it is necessary to know what is needed to get world coordinates from an image with depth information.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Where:

- s is a scaling factor
- (X, Y, Z) are 3D world coordinates
- (u, v) are 2D pixel coordinates
- $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix
- $\mathbf{t} \in \mathbb{R}^{3\times1}$ is the translation vector
- $\mathbf{K}$ is the camera **intrinsic matrix**

This can be simplified to:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{K} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Where:

- (x, y, z) are the homogeneous pixel coordinates
- (X_c, Y_c, Z_c) are the 3D coordinates in the camera frame
- $\mathbf{K}$ is the camera **intrinsic matrix**,

The intrinsic matrix $\mathbf{K}$ can be defined as:

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

Where:

- f_x, f_y are the focal lengths in pixels along the x and y axes
- s is the skew coefficient between the x and y axes (usually 0)

- (c_x, c_y) is the principal point (usually near the image center)

When performing matrix-vector multiplication K and the 3D coordinates in the camera frame, the result is seen in equation 4.1:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} X \cdot f_x + Z \cdot c_x \\ Y \cdot f_y + Z \cdot c_y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{X \cdot f_x}{Z} + c_x \\ \frac{Y \cdot f_y}{Z} + c_y \\ 1 \end{bmatrix} \tag{4.1}$$

Isolating for the 3D coordinates yields equation 4.2:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{(x - c_x) \cdot Z}{f_x} \\ \frac{(y - c_y) \cdot Z}{f_y} \\ \text{depth}(x, y) \end{bmatrix} \tag{4.2}$$

Performing Intrinsic Camera Calibration

Intrinsic camera calibration was done using OpenCV and a ChArUco board provided by OpenCV, the information about the board can be seen below in table 4.1. A total of 20 images were taken to verifiy the camera calibration.

Parameter	**Description**
`dictionary`	`DICT_5X5_250` — predefined ArUco dictionary with 5x5 bits and 250 markers
`size`	(4, 3) — number of squares (cols, rows) on the board
`squareLength`	0.068 m — length of each square's side
`markerLength`	0.045 m — length of each ArUco marker inside a square
`detector_params`	Default detector parameters for ArUco marker detection

Table 4.1: Parameters defining the ChArUco board in OpenCV.

An image of the ChArUco board provided by OpenCV can be seen in figure 4.1. The combiantion of both a checker board and a ArUco board is in order to provide a more refined calibration.

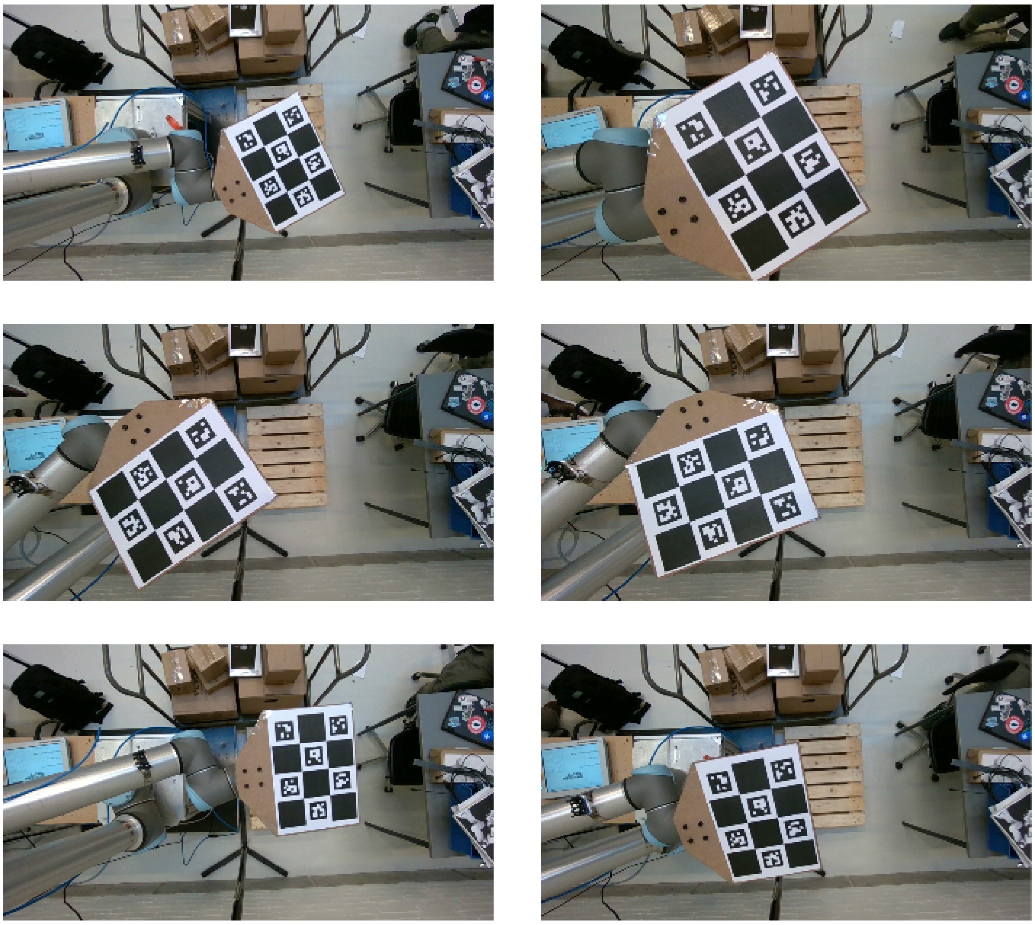


Figure 4.2: Sample images of the ChArUco board, attached to the UR10 robot's end-effector, captured from various orientations and distances during the intrinsic camera calibration process.

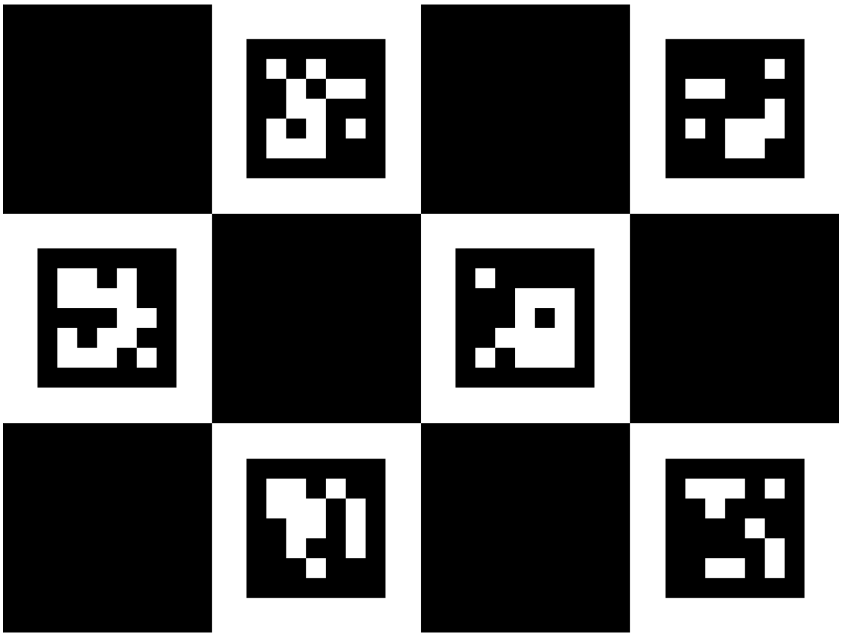


Figure 4.1: ChArUco board with the parameters seen in table 4.1.

Examples of images taken can be seen in figure 4.2, where six images can be seen where the ChArUco board is attached to the end-effector of the UR10 robot and different orientations and distances were used to provide a variation in data gathered. From the images taken the calibration results were calculated.

Intrinsic Camera Calibration Results

The results form the intrinsic camera calibration can be seen below: **Reprojection error:**

$$\text{Reprojection error} = 0.06211595621291655$$

Camera matrix:

$$\mathbf{K} = \begin{bmatrix} 645.998837 & 0 & 650.31852791 \\ 0 & 646.34751883 & 361.47141469 \\ 0 & 0 & 1 \end{bmatrix}$$

Distortion coefficients:

$$\mathbf{D} = \begin{bmatrix} -0.06254493 & 0.07836239 & 0.00043746 & 0.00122365 & -0.05503944 \end{bmatrix}$$

A low reprojection error is a positive sign of a well-calibrated camera and of a significant varied dataset.

4.2 Extrinsic Camera Calibration

It is also necessary to know the transformation from the robot's coordinate system to the camera's coordinate system. This can be done through Hand-Eye calibration. The theory behind it now be explained.

Hand-Eye Calibration Model (Eye-to-Hand)

The fundamental equation for hand-eye calibration in an eye-to-hand setup is:

$$\mathbf{A}_i\mathbf{X} = \mathbf{X}\mathbf{B}_i$$

Where:

- $\mathbf{A}_i \in SE(3)$: Robot motion between two end-effector poses (in robot base frame)
- $\mathbf{B}_i \in SE(3)$: Calibration board motion between two poses (in camera frame)
- $\mathbf{X} \in SE(3)$: Rigid transformation from calibration board (on robot) to camera frame

The goal is to solve for $\mathbf{X}$, representing the extrinsic calibration between the robot and the camera.

$$\mathbf{X} = \begin{bmatrix} \mathbf{R}_X & \mathbf{t}_X \\ \mathbf{0} & 1 \end{bmatrix}$$

This transformation enables converting 3D coordinates observed by the camera into the robot's coordinate system.

The objective is to find the transformation from the camera to the robot base, denoted as:

$${}^{b}\mathbf{T}_c$$

Where:

- ${}^{b}\mathbf{T}_{c}$: transformation from camera frame to robot base
- ${}^{c}\mathbf{T}_{t}^{(i)}$: transformation from target (calibration board) to camera in pose i
- ${}^{g}\mathbf{T}_{b}^{(i)}$: transformation from robot base to end-effector in pose i
- ${}^{g}\mathbf{T}_{t}$: transformation from target to end-effector

The calibration requires multiple poses to be taken at different positions where:

Pose 1:

$$ {}^{g}\mathbf{T}_{b}^{(1)} \cdot {}^{b}\mathbf{T}_{c} \cdot {}^{c}\mathbf{T}_{t}^{(1)} = {}^{g}\mathbf{T}_{t} $$

And then after moving the end-effector for the second pose:

Pose 2:

$$ {}^{g}\mathbf{T}_{b}^{(2)} \cdot {}^{b}\mathbf{T}_{c} \cdot {}^{c}\mathbf{T}_{t}^{(2)} = {}^{g}\mathbf{T}_{t} $$

Eliminating ${}^{g}\mathbf{T}_{t}$:

$$ {}^{g}\mathbf{T}_{b}^{(1)} \cdot {}^{b}\mathbf{T}_{c} \cdot {}^{c}\mathbf{T}_{t}^{(1)} = {}^{g}\mathbf{T}_{b}^{(2)} \cdot {}^{b}\mathbf{T}_{c} \cdot {}^{c}\mathbf{T}_{t}^{(2)} $$

Rearranged form: This structure follows the general hand-eye calibration equation:

$$ \mathbf{A}_i \cdot \mathbf{X} = \mathbf{X} \cdot \mathbf{B}_i $$

Where:

$$ \mathbf{A}_i = ({}^{g}\mathbf{T}_{b}^{(2)})^{-1} \cdot {}^{g}\mathbf{T}_{b}^{(1)}, \quad \mathbf{B}_i = {}^{c}\mathbf{T}_{t}^{(2)} \cdot ({}^{c}\mathbf{T}_{t}^{(1)})^{-1}, \quad \mathbf{X} = {}^{b}\mathbf{T}_{c} $$

Rearranged, the full equation reads:

$$ ({}^{g}\mathbf{T}_{b}^{(2)})^{-1} \cdot {}^{g}\mathbf{T}_{b}^{(1)} \cdot {}^{b}\mathbf{T}_{c} = {}^{b}\mathbf{T}_{c} \cdot {}^{c}\mathbf{T}_{t}^{(2)} \cdot ({}^{c}\mathbf{T}_{t}^{(1)})^{-1} $$

An example of the transformation aforementioned can be seen in figure 4.3, where a chequerboard has been attached to a robot and the camera is external. By taking several images with the end-effector at different poses, it is possible to find the transformation from camera to robot base.

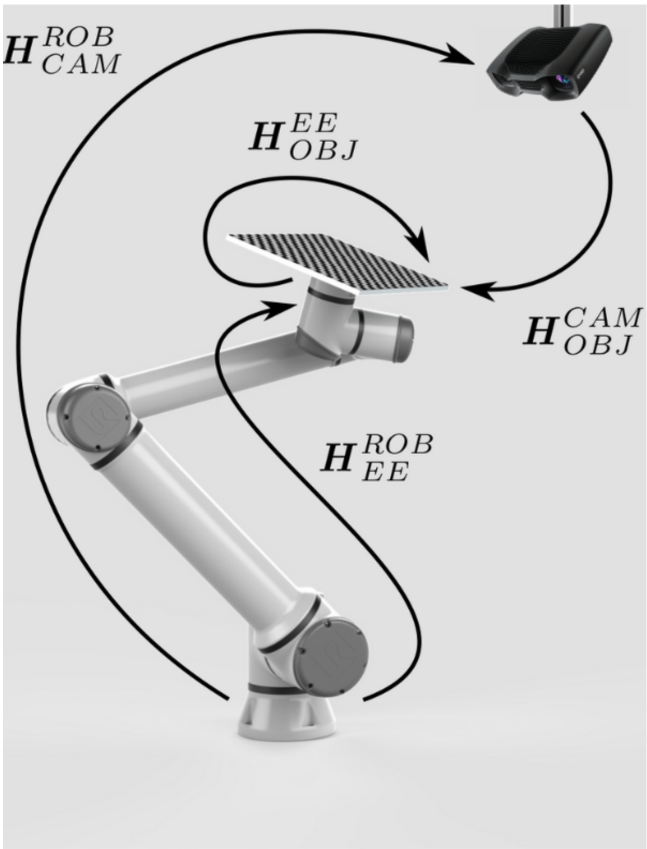

$\boldsymbol{H}^{ROB}_{CAM}$
$\boldsymbol{H}^{EE}_{OBJ}$
$\boldsymbol{H}^{CAM}_{OBJ}$
$\boldsymbol{H}^{ROB}_{EE}$

Figure 4.3: Eye-on-base transformations on a Univeral Robot with external camera.[52]

The same ChArUco board, from figure 4.1, attached on the robot for intrinsic camera calibration was also used to find the transformation from camera to robot base. A total of 70 images were taken to perform the extrinsic calibration, and the results can be seen on table 4.2, where the results for all the calibrations methods presented similar results, indicating a successful calibration. The chosen method is Horaud as stated on the paper from the author, it has presented more robust results than the others and is also a more recent approach. [53]

Method	Tx (cm)	Ty (cm)	Tz (cm)	Euler XYZ (°)
TSAI	49.9196	6.9363	111.1288	177.3894, 0.0518, 4.5484
PARK	49.8940	6.5317	111.2696	179.6665, -0.3803, 5.4582
HORAUD	49.8940	6.5320	111.2697	179.6571, -0.3867, 5.4524
ANDREFF	49.7521	6.5150	110.4031	179.6553, -0.3838, 5.4515

Table 4.2: Comparison of camera-to-base extrinsic calibration results using different hand-eye calibration methods.

Depth Data Statistics

The Intel RealSense D455 RGB-D camera has demonstrated undesired behaviour regarding its consistency in depth values. The camera presented constant flickering when used in terms of depth data, the same (x, y) point could even oscillate almost 100 mm as seen in table 4.3. It has also been observed that taken the mean of the same points at different times yields a difference of around 20- to 30 mm which is significant and will lead to the develop system being adapted this challenge in order to still produce reliable results.

Pixel	Current	Count	Mean	StdDev	Min	Max	Range	Last 5	Last 10	Last 20
(518, 38)	2097 mm	82	2095.3 mm	16.7 mm	2057 mm	2134 mm	77 mm	2102.0 mm	2094.5 mm	2093.6 mm
(561, 76)	1892 mm	82	1884.4 mm	14.3 mm	1849 mm	1908 mm	59 mm	1889.2 mm	1886.6 mm	1886.4 mm
(489, 62)	2051 mm	82	2082.9 mm	21.0 mm	2026 mm	2120 mm	94 mm	2072.2 mm	2078.5 mm	2079.6 mm
(541, 107)	1857 mm	82	1882.2 mm	16.7 mm	1844 mm	1908 mm	64 mm	1876.8 mm	1880.1 mm	1874.7 mm

Table 4.3: Statistics for selected depth points over 82 frames.

4.3 Setup

Figure 4.4 shows the setup of project. It consists of an Intel RealSense D455 RGB-D camera, external and statically placed to ensure its transform to the robot base does not change. The camera has also view of the placing pallet where packages shall be placed when operating. The other two highlighted objects are the UR10 robot and the VG10 vacuum gripper. These are placed on metal table that provides enough stability for the movement of robot with the gripper.

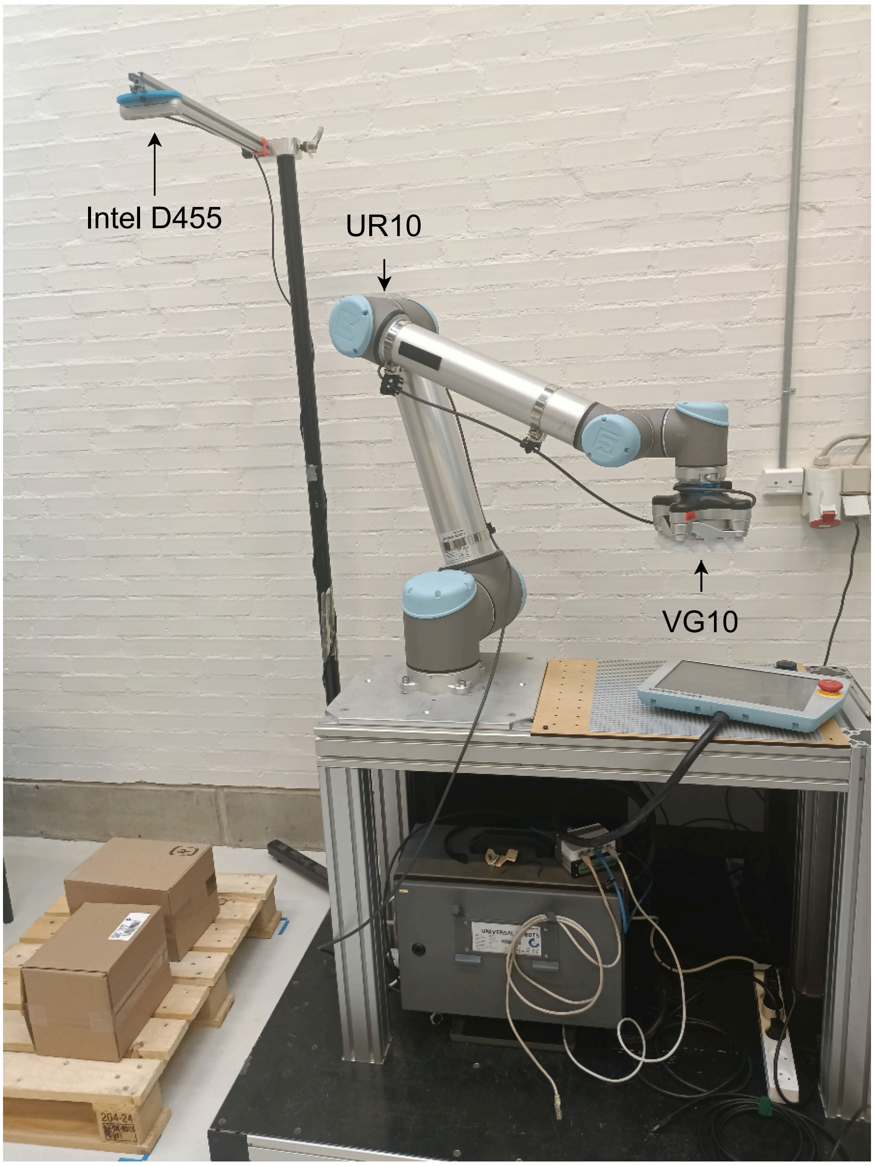


Figure 4.4: Robotic Depalletizing Setup with the main elements highlighted: Intel RealSense D455 RGB-D camera, UR10 robot and VG10 vacuum gripper.

Software Setup

The setup is seen as sufficient to perform depalletizing operation. As for the software setup, MoveIt2 has been used. A custom MoveIt config package was made based on the Universal Robot standard config package, however a new URDF was made to include the VG10 gripper and the Intel RealSense D455 camera. The camera frame has its translation and orientation retrieved from the extrinsic camera calibration values seen previously in table 4.2. The URDF has also the table in which the robot has been placed on in order to avoid collisions while planning trajectories, as well as the camera mount so no part of the setup can collide with each other. The setup visualised in Rviz is present in figure 4.5.

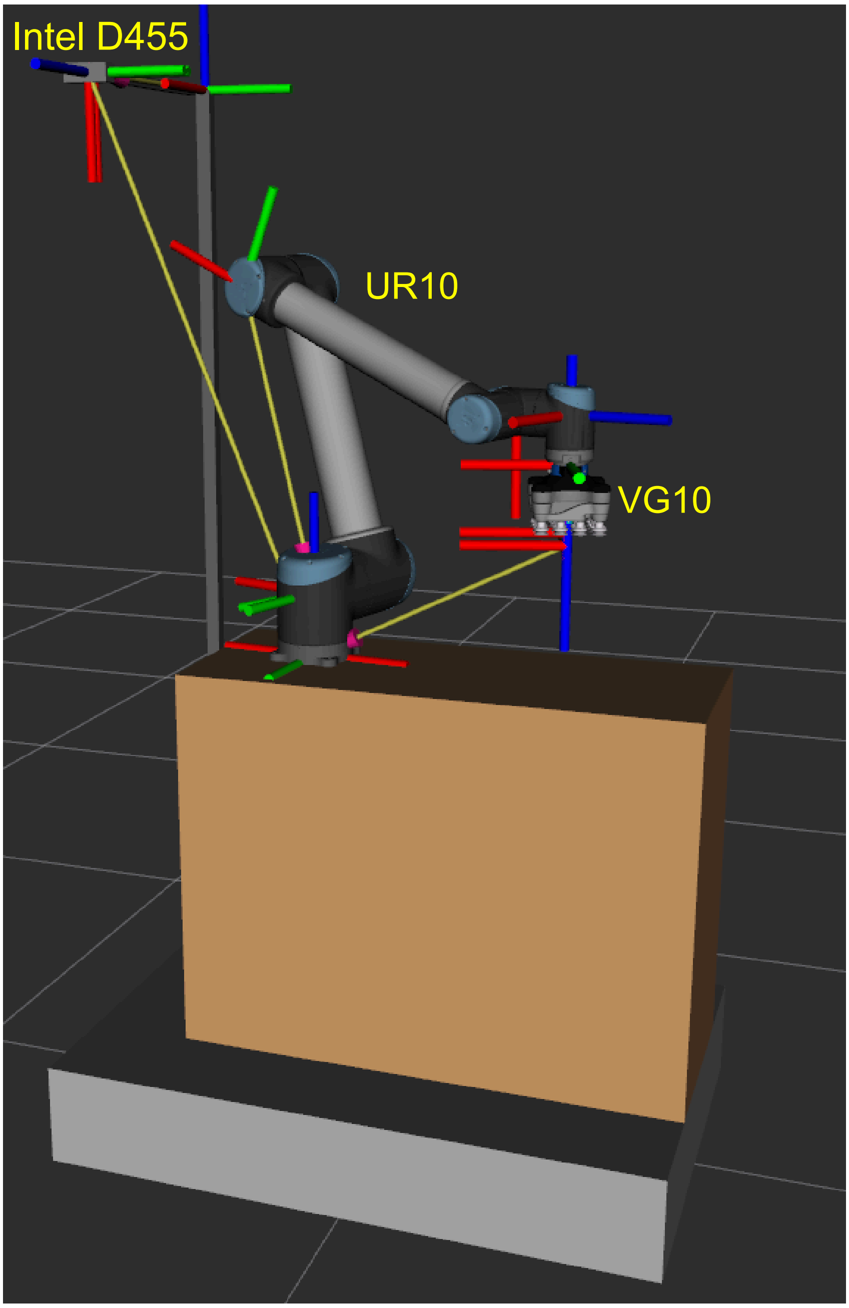


Figure 4.5: Robotic Depalletizing Setup in MoveIt2 visualized in Rviz with coordinate frames shown and with the main elements highlighted: Intel RealSense D455 RGB-D camera, UR10 robot and VG10 vacuum gripper.

ROS2 Packages

The MoveIt2 setup allows for a ROS2 system to be built around the capabilities of programming robot behaviour. Due to this possibility ROS2 environment was developed to enable a solution capable of depalletizing. Figure 4.6 shows an overview of the custom ROS2 packages developed for the solution. It is important to note the modularity provided by said packages as they can be applied to another system with minimal to none modification.

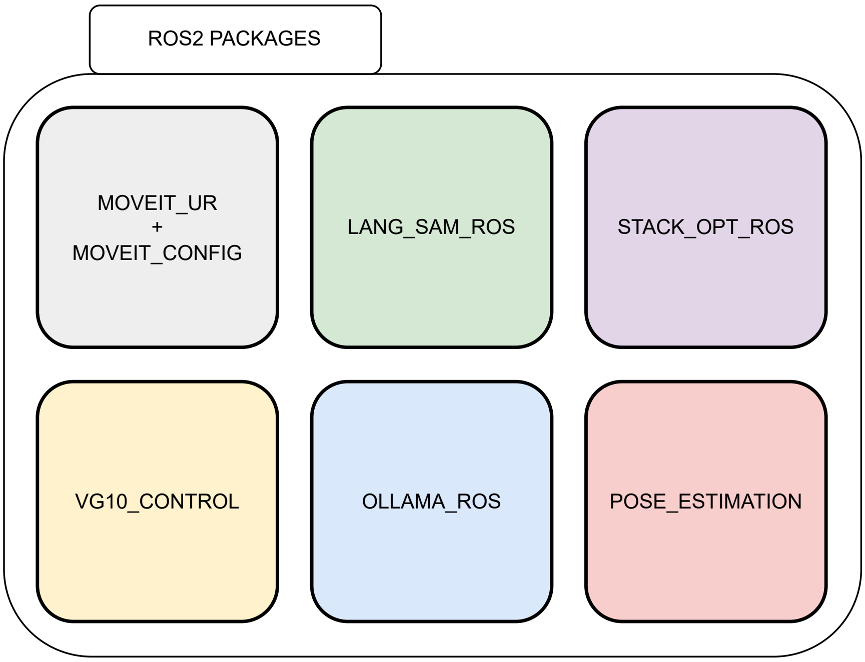


Figure 4.6: Overview of the custom ROS2 package architecture for the depalletizing system, highlighting the modular components including MoveIt2 control, language-guided segmentation (LANG SAM ROS), stack optimization (STACK OPT ROS), gripper control (VG10 CONTROL), LLM interface (OLLAMA ROS), and pose estimation.

As seen in figure 4.6, there are six main elements in the ROS2 architecture of the depalletizing system. From the top left is the MoveIt2 setup necessary to control the robot, where **"MOVEIT_UR"** is the main control package of the robot solution where the MoveGroup interface is used to program robot behavior and **"MOVEIT_CONFIG"** refers to the custom MoveIt config package previously mentioned and also the description for the URDF and the ros2 control packages necessary to set up the robot. As for the control of the robot, the ros2 control is used with the scaled joint trajectory controller. The mentioned MoveIt2 configuration and setup is essential to allow for intelligent behaviour of the robot in the developed ROS2 environment, however it is not enough for a complete solution and therefore other modules are necessary.

The next module on figure 4.6 is the **"LANG_SAM_ROS"** refering to Language SAM in ROS. This package is responsible for the image segmentation used identify the packages from the RGB images received from the D455 camera. A server node runs allowing for the segmentation of an image given a text prompt of the desired segmented object, the service is called, in the **"MOVEIT_UR"** package as it is necessary to use the image segmentation to locate the objects in the image but also in the 3D world coordinates.

The **"STACK_OPT_ROS"** is the module responsible for finding the optimal box poses on the pallet given the objective function. When the dimensions of the packages are estimated based on the depth information, they are then used by the optimizer to be optimally placed on the pallet. This package makes used of ORTools, an operation research tools library in python that is used to create the optimization problem and also solve it.

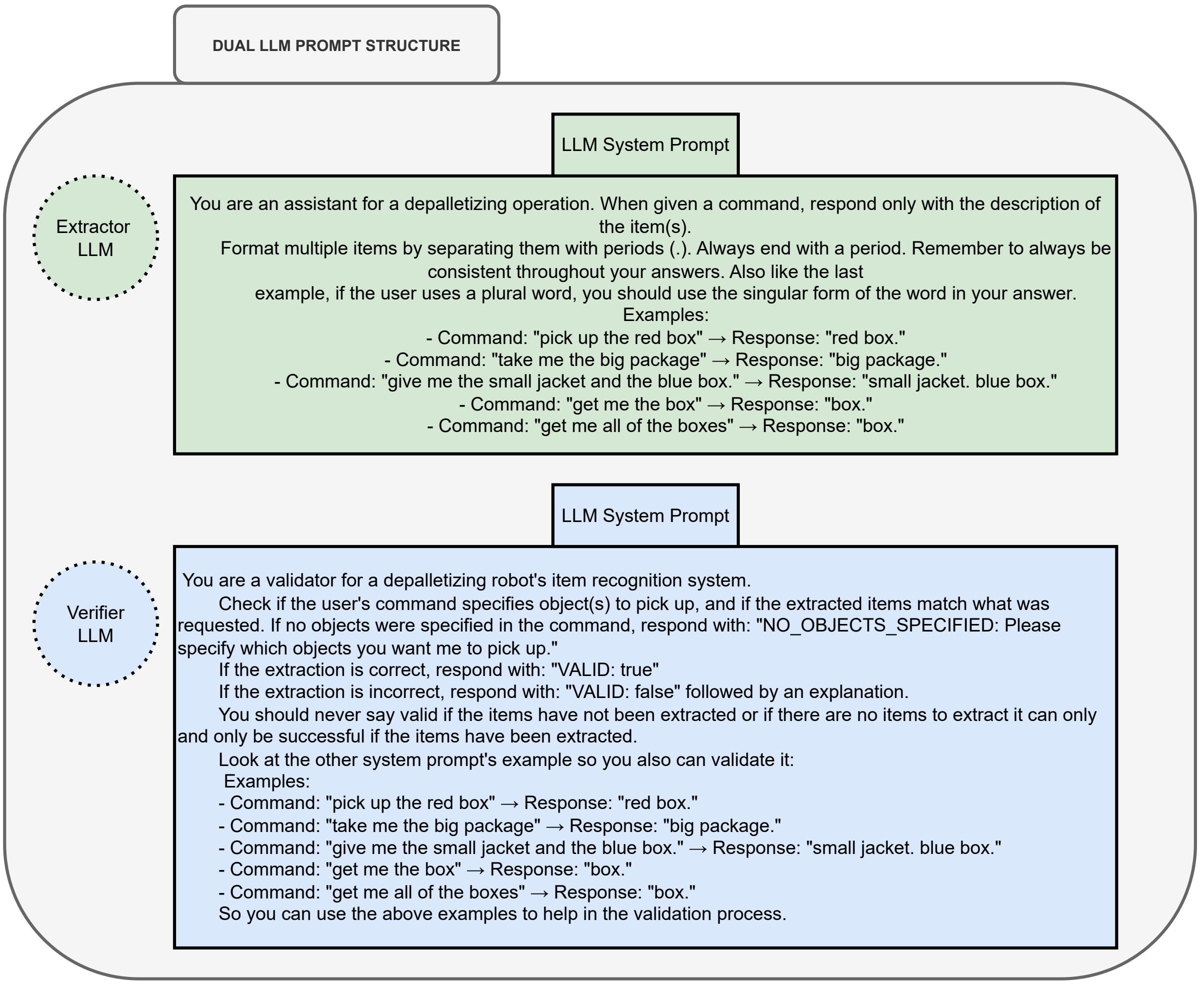
The **”VG10_CONTROL”** package is the interface module between the gripper VG10 and the other parts of the system. In this part of the system a Modbus is used to both determine and read the vacuum levels of both channels of the gripper, which allows for a closed loop control based on the vacuum readings. The vacuums of each channel can be set through a ROS2 service call and the actual levels are published on topics for a control throughout the whole operation.

The **”OLLAMA_ROS”** package is the Large Language Model part of the ROS2 environment. In ”OLLAMA_ROS” the dual LLM architecture is present and interfaced through ROS2 actions where the operator can send text commands and allow the depalletizing operation to be started. As the name of the package suggests, Ollama is used to run the LLMs locally on the project’s computer.

The package **”POSE_ESTIMATION”** is the one responsible for generating an estimated pose of the packages in the scene. The package receives the segmented objects and uses RANSAC to identify the top plane and find the normal vector. This enables the robot to grasp the packages from above while operating.

OLLAMA_ROS - ROS2 LLMs Package

The system is composed of LLMs used to process operator commands and enable an easy collaboration between robot and human. In order to process the operator’s commands a dual LLM architecture is used to extract the objects which the operator wants to pick, and also to validate if the extraction of said objects was correctly made. Figure 4.7 shows the designed prompts for the LLMs used.



You are an assistant for a depalletizing operation. When given a command, respond only with the description of the item(s).
Format multiple items by separating them with periods (.). Always end with a period. Remember to always be consistent throughout your answers. Also like the last
example, if the user uses a plural word, you should use the singular form of the word in your answer.
Examples:
- Command: "pick up the red box" → Response: "red box."
- Command: "take me the big package" → Response: "big package."
- Command: "give me the small jacket and the blue box." → Response: "small jacket. blue box."
- Command: "get me the box" → Response: "box."
- Command: "get me all of the boxes" → Response: "box."

You are a validator for a depalletizing robot's item recognition system.
Check if the user's command specifies object(s) to pick up, and if the extracted items match what was requested. If no objects were specified in the command, respond with: "NO_OBJECTS_SPECIFIED: Please specify which objects you want me to pick up."
If the extraction is correct, respond with: "VALID: true"
If the extraction is incorrect, respond with: "VALID: false" followed by an explanation.
You should never say valid if the items have not been extracted or if there are no items to extract it can only and only be successful if the items have been extracted.
Look at the other system prompt's example so you also can validate it:
Examples:
- Command: "pick up the red box" → Response: "red box."
- Command: "take me the big package" → Response: "big package."
- Command: "give me the small jacket and the blue box." → Response: "small jacket. blue box."
- Command: "get me the box" → Response: "box."
- Command: "get me all of the boxes" → Response: "box."
So you can use the above examples to help in the validation process.

Figure 4.7: Prompt structure for the dual LLM architecture, showcasing the system prompts designed for the Extractor LLM (to identify objects and attributes from user commands) and the Verifier LLM (to validate the Extractor's output).

The prompts were assigned to two Qwen3 models with 1.7 billion parameters through Ollama. Ollama models can be accessed through its python packages, which allow for behavior customization through it. The system prompts allows the LLMs to follow a set of instructions if there is need for a certain behaviour. The Qwen3 models allow for reasoning making them useful for the types of task where accuracy and logical thinking is needed. The first LLM is the extractor LLM that is assigned to retrieve the objects the operator wants to pick up for the operation, as seen in its instruction prompt, it has also to be able to extract the characteristics of the objects in the given prompt, so if the operator specifies he wants to pick a ”red box”, the LLM should be able to retrieve the object itself and the description adjective as well, therefore the extracted text should be: ”red box”. This extraction with not only the object itself but the description is important to the segmentation being used, where both a text description of the object and an image is used to find the desired objects in the image. As for the verifier LLM, the job assigned to it is of a validator of the text extraction performed. The LLM receives the text from the user and the answer from the previous LLM and evaluates if the objects were correctly extracted; if so, it is possible to perform the operation. Figure 4.8 shows the operational flow of the prompt based object extraction. In the example given, an operator starts the conversation with an instruction command and the first LLM extracts the object and the second LLM is used to verify if the extraction was correctly made.

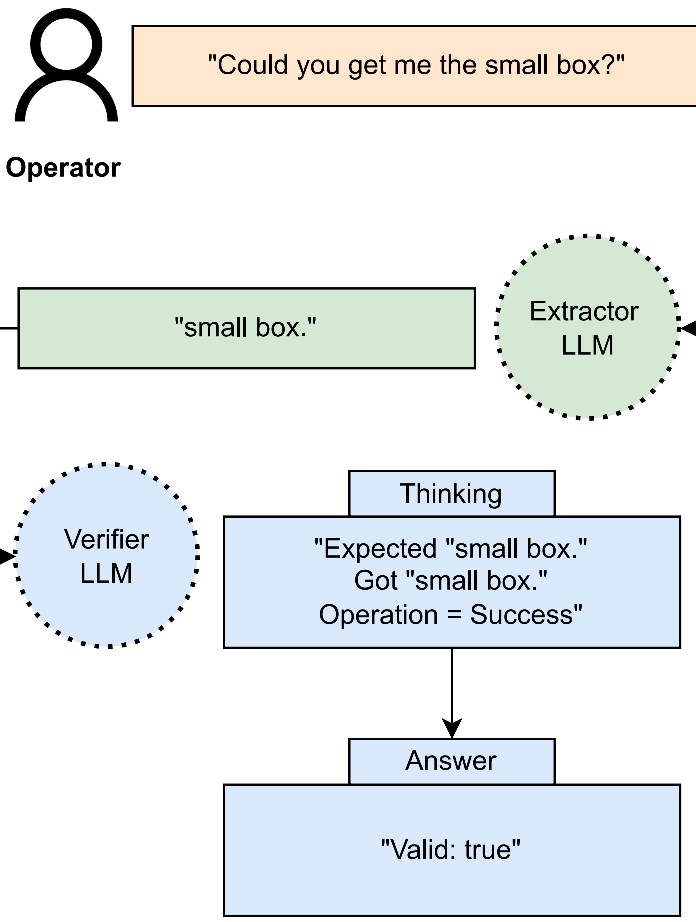


Figure 4.8: Example workflow of the dual LLM system: An operator's command is processed by the Extractor LLM to identify the target object ("small box"), and the Verifier LLM confirms the extraction's validity, enabling the operation to proceed.

LANG_SAM_ROS - ROS2 Image Segmentation Package

The "LANG_SAM_ROS" module is responsible for the image segmentation of the operation pipeline. In this module, a server node runs to enable a ROS2 service call for segmenting an object, given through a string when calling the service as well as the image where the object is present. It is possible to define more than one desired segmentation object, however it is important to note that it needs to divided ".", so for example if there is a blue box and a red box that needs to be segmented, then the string given would be: "blue box. red box." in this way both objects will be segmented in the image. This type of prompt based segmentation is possible by using a combination of GroundingDINO and SAM. GroundingDINO allows for the text commands to be used in combination with SAM, since SAM as of itself is not able to take text commands. GroundingDINO is an object detector that also can take prompts as for example the ones specified before with red and blue boxes. Whereas SAM stands for Segment Anything Model, a segmentation model trained by META. The use of the two allow for powerful segmentation based on object characteristics without the need to fine-tune the models. Whereas models like YOLO usually need fine-tuning for desirable performance. The outputs of Language SAM are the segmentation mask with the specified objects as well as confidence scores. In the implementation, a confidence score of at least 0.3 is needed for an object to be taken into account, otherwise it is considered a false positive and then disregarded. Figure 4.9 demonstrates the logic used in LANG SAM ROS service node where the aforementioned process is handled.

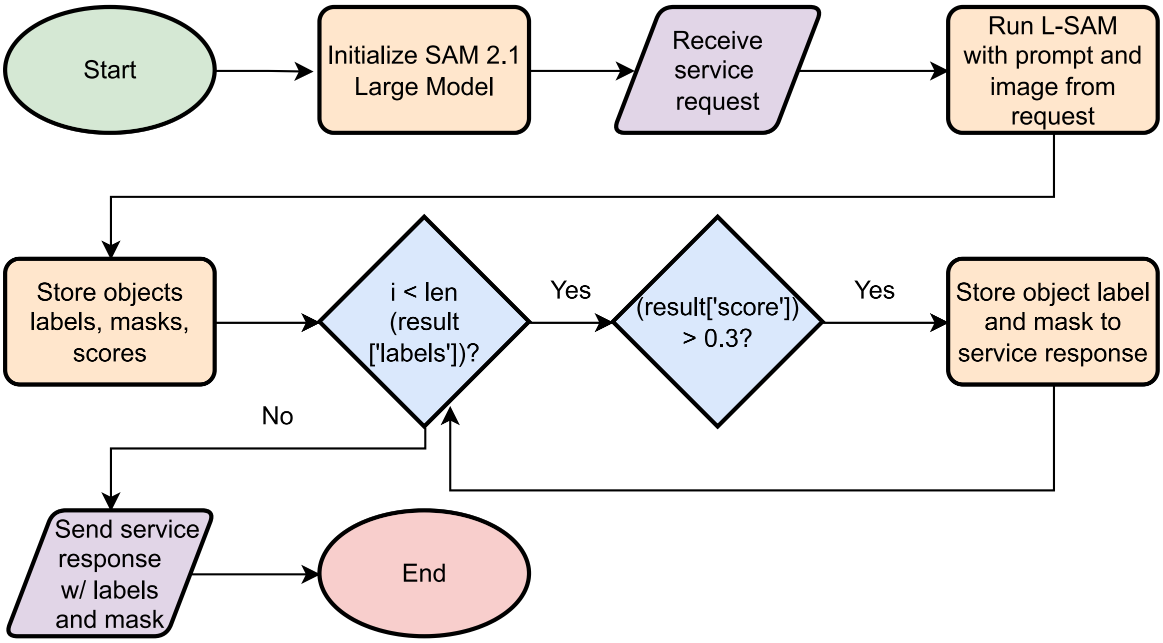

Start
Initialize SAM 2.1 Large Model
Receive service request
Run L-SAM with prompt and image from request
Store objects labels, masks, scores
i < len (result ['labels'])?
Yes
(result['score']) > 0.3?
Yes
Store object label and mask to service response
No
Send service response w/ labels and mask
End

Figure 4.9: Flowchart illustrating the operational logic of the LANG SAM ROS service node, which receives an image and text prompt, performs segmentation using Language-SAM (GroundingDINO + SAM), filters results by confidence score, and returns the object labels and masks.

POSE_ESTIMATION - ROS2 Package

After the objects are segmented in the image, it is necessary to know their pose in 3D space, for each segmented object in the image the top plane is found through RANSAC, the first object to be picked is the one where it has the highest z value in the coordinate system. This is done to ensure no boxes fall and make the stack unstable. In this way always the top boxes are moved. The segmentation mask in the image is not only composed of top plane pixels, so it is crucial to guarantee that the plane found on the 3D coordinate system is the top plane and not the side plane from one of the other box faces. A total of 100 iterations with 3 points from the segmented image is done and the tolerance of 1.25 cm is used to defined if points are in the same plane or not. The plane found with most inliers is then chosen as assumed that the majority of the segmentation mask is composed by the box's top face due to the camera placement from above. Figure 4.10 shows the flow chart of the Pose Estimation Service.

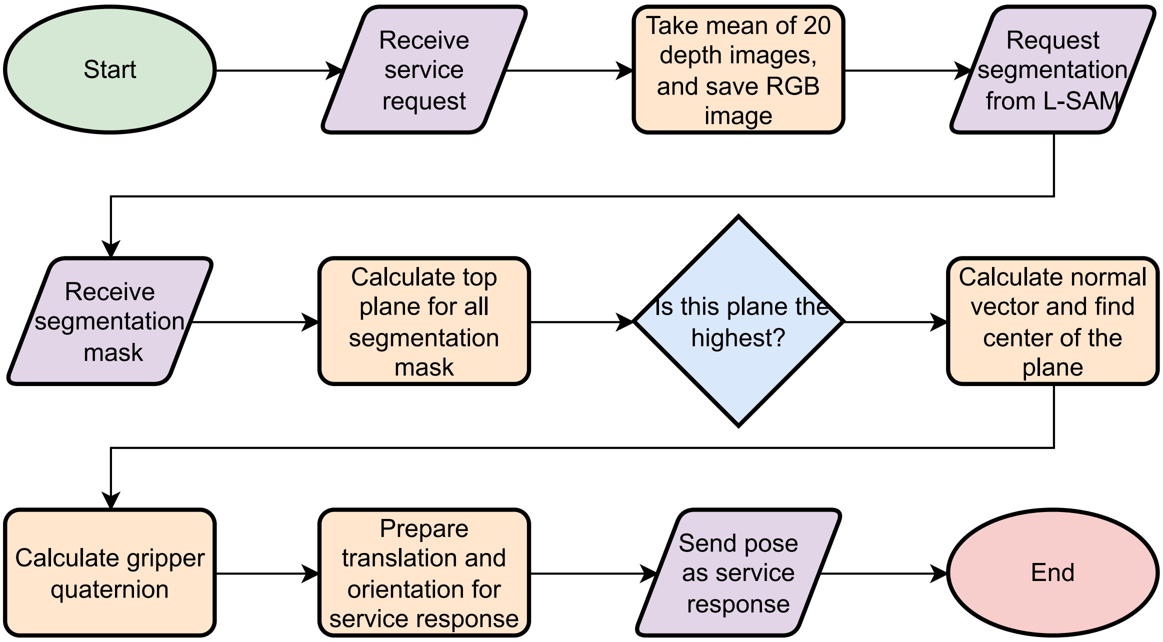

Start
Receive service request
Take mean of 20 depth images, and save RGB image
Request segmentation from L-SAM
Receive segmentation mask
Calculate top plane for all segmentation mask
Is this plane the highest?
Calculate normal vector and find center of the plane
Calculate gripper quaternion
Prepare translation and orientation for service response
Send pose as service response
End

Figure 4.10: Flowchart of the Pose Estimation ROS2 service, detailing the process from receiving a request, acquiring and processing depth/RGB images, obtaining segmentation masks via L-SAM, calculating the top plane for each segmented object, identifying the highest plane, and then determining its normal vector and center point for grasping

When the top plane is found, its 3D points are used to find the center X and Y points and the estimated Z is the height of the plane in the center point and the normal vector is calculated for the gripper orientation, such that the grasp is performed from above. The 3D points of the plane are then reprojected on the image and used perform rectangle fitting on them. The corner pixels from this rectangle fitting are used to find the orientation of the box in 3D space, so the 2D points are projected in 3D and their orientation is calculated. The boxes are assumed to be perpendicular to the z-axis of the robot base, the calculation of the quaternion for the gripper orientation adjusts to always align the gripper with the z-axis of the robot base. Lastly, the four 3D corner points if the box's top face are used to estimate the box's dimensions. These estimated dimensions are then used in the stack optimiser to place the boxes optimally according to the objective function. Figure 4.11 shows a the operational logic used to calculate the gripper's orientation and the estimated box's dimensions.

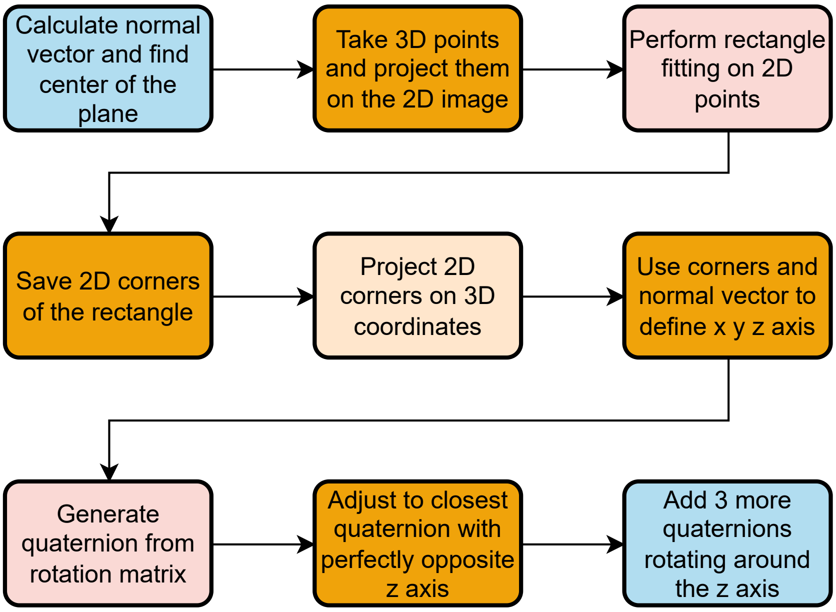

Calculate normal vector and find center of the plane
Take 3D points and project them on the 2D image
Perform rectangle fitting on 2D points
Save 2D corners of the rectangle
Project 2D corners on 3D coordinates
Use corners and normal vector to define x y z axis
Generate quaternion from rotation matrix
Adjust to closest quaternion with perfectly opposite z axis
Add 3 more quaternions rotating around the z axis

Figure 4.11: Pipeline for calculating gripper orientation and box dimensions from the identified top plane: 3D plane points are projected to 2D for rectangle fitting, 2D corners are reprojected to 3D to define box axes and orientation, and the gripper quaternion is generated and adjusted.

STACK_OPT_ROS - ROS2 Stack Optimisation Package

The module "STACK_OPT_ROS" is used with the dimensions estimated from the previously described pose estimation method, where it also estimates the box's dimensions. The height is possible to be estimated through the depth information provided by the RGB-D Intel RealSense camera, however another method for more accurate estimation is used. Since the ground's height in relation to the robot base is known, it is possible to estimate the boxes's height by taking the z component at the grasping position and using it to know the distance from the top face to the ground. The length and width of the top face are though estimated based on depth information as explained in the pose estimation module. When the dimensions of the box are known, then it is possible to calculate the optimal pose for their placement according to the objective function. In order to create the optimisation problem, ORTools python library provided by Google is used. This library allows for constrained optimisation problems to be made, and for parameters like solving time to be set A flowchart of the stack optimisation service is shown in figure 4.12.

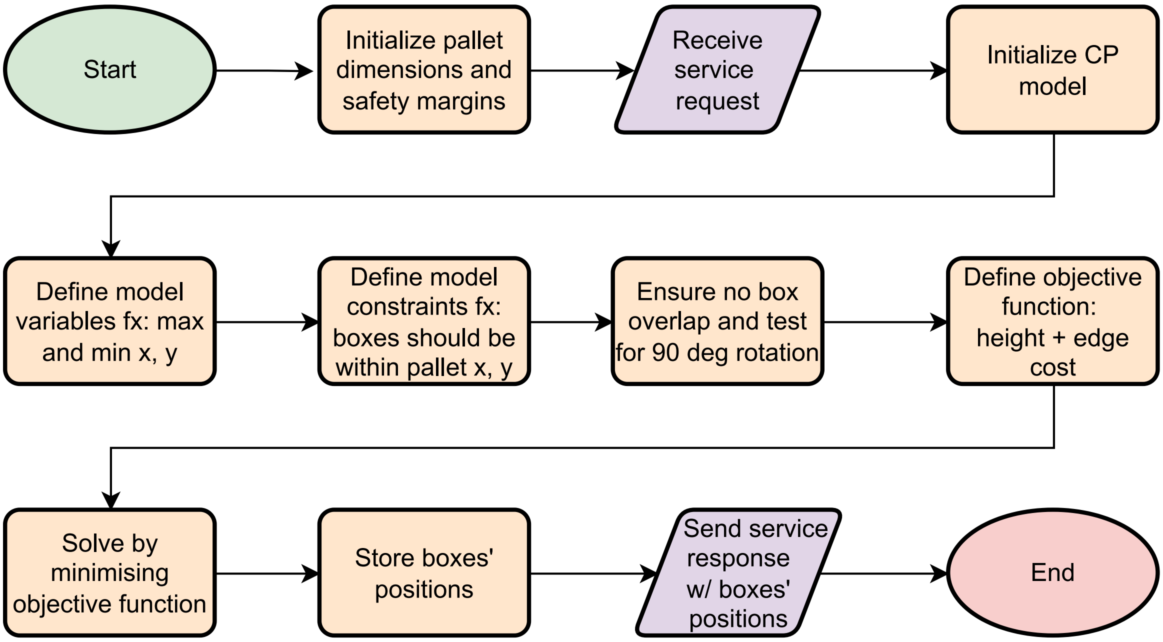

Start
Initialize pallet dimensions and safety margins
Receive service request
Initialize CP model
Define model variables fx: max and min x, y
Define model constraints fx: boxes should be within pallet x, y
Ensure no box overlap and test for 90 deg rotation
Define objective function: height + edge cost
Solve by minimising objective function
Store boxes' positions
Send service response w/ boxes' positions
End

Figure 4.12: Flowchart of the STACK OPT ROS service for stack optimization, outlining the initialization of pallet parameters, definition of model variables and constraints (e.g., pallet boundaries, no-overlap), formulation of the objective function (minimizing height and edge cost), and solving using a CP model to output optimal box positions.

The implementation consisted of four main elements, the boxes' heights, the pallet dimensions, the safety margin and the support constraints. The boxes need to be placed inside the pallet with all the dimensions inside of it, however due to imprecise depth information, the boxes' dimensions are not perfectly estimated and to take this into account a safety margin is implemented, such that every box places on the pallet has to be 10 cm inside in both the x and y axis of the pallet, and the boxes also have 10 cm margin from each other when placed on the pallet. This is done to minimise the effects of unreliable depth data, however in case of more precise information, these safety margins could be reconsidered. The boxes' heights are also minimised, though mainly in the sense that if there is ground available for boxes to placed on the pallet while fulfilling the safety margins, the boxes are placed on the pallet and not on top of each other. The capability of rotating boxes if height is greater than one of the other two box's dimensions has not been implemented. There are though 90° rotations around the boxes' z axis so if a box where to fit if rotated by 90° then this option would the optimal. Figure 4.13 shows the importance right box placement. The support constraints have also been set in case of boxes being placed on previously placed boxes, the required support area needed is of at least 50% otherwise, it is assumed to be unstable when stacking the box without the minimum support area. This therefore, means that boxes may not have am optimal placement depending on the dimensions and the stack of boxes at operation time. The optimisation problem and the objective function for new boxes to be placed is set in equation 4.3:

$$\min \quad w_{\text{height}} \cdot z_{\text{new}} + w_{\text{stability}} \cdot \text{unsupported\_area} + w_{\text{edge}} \cdot \text{edge\_distance} \tag{4.3}$$

where:

$$\begin{aligned} w_{\text{height}} &= 15 \quad \text{(penalizes placing the box higher)} \\ w_{\text{stability}} &= 10 \quad \text{(penalizes unsupported footprint)} \\ w_{\text{edge}} &= 5 \quad \text{(encourages placement near edges)} \end{aligned}$$

and:

> *edge_distance* is the sum of the horizontal and vertical distances from the new box to the closest edges of the pallet. This encourages placing boxes near the corners or along the pallet boundaries, which tends to improve packing efficiency as space in the middle can be saved.
>
> z_{new} *is the new box height in relation to the ground*

It is important to mention that the constraints still need to be respected otherwise there is no feasible solution for the optimisation problem.

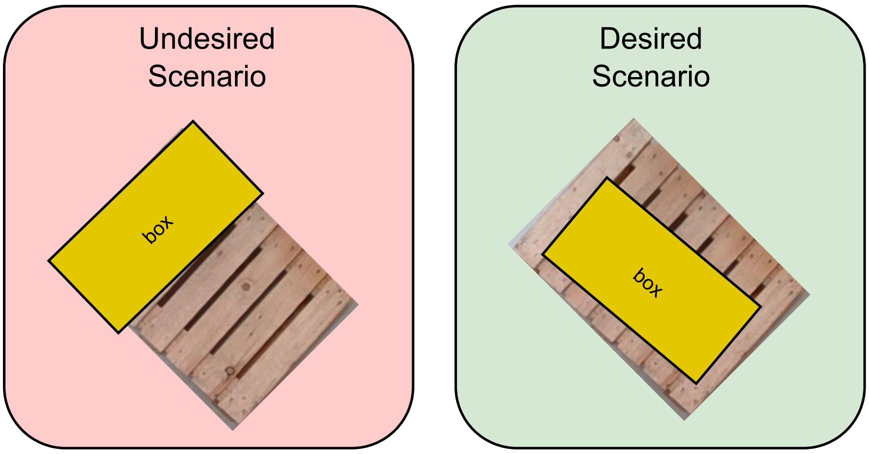


Figure 4.13: Illustration of desired vs. undesired box placement scenarios, highlighting the importance of support constraints in the stack optimization. (Left) An undesired scenario where a box is placed with insufficient support on another. (Right) A desired scenario with adequate support ensuring stack stability.

Grasp Planning

It is crucial to have a secure and firm grasp when performing pick and place operations. In order to accomplish a firm grasp, the use of the vacuum gripper's actual pressure were necessary. As the RGB-D camera did not provide ideal depth data, and the assurance of a sufficient grasping can only be done with a type of verification, the actual vacuum levels were read and used to evaluate the grasping.

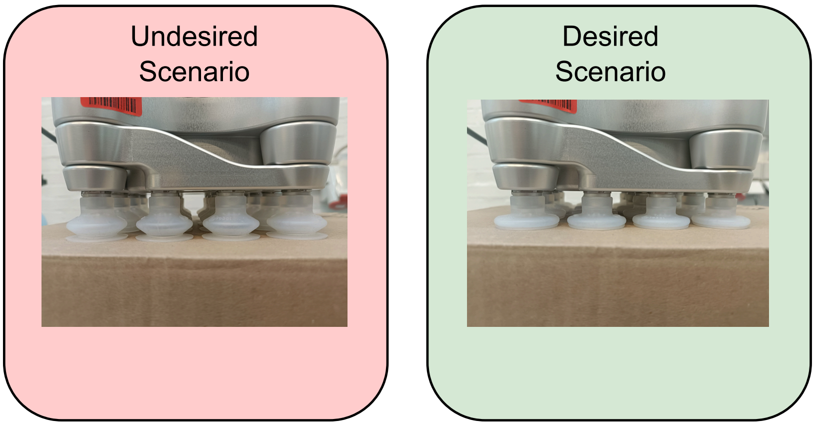


Figure 4.14: Illustration of the adaptive grasp planning logic. (Left) An undesired scenario where the gripper has not established sufficient vacuum pressure for a firm grasp. (Right) A desired scenario where the gripper has moved down incrementally to achieve adequate suction and a secure grasp on the box.

The logic applied needed to take into account both the inadequate depth data and the necessity of having a firm grasp. Therefore, the grasp position was set to 3-5 cm above the box (depending on the quality of the depth data), and the vacuum gripper was activated. From there, the gripper moved 0.5cm down each step, and then the actual vacuum levels were read, in case of the vacuum levels being sufficient to grasp a box then it could proceed with the operation, otherwise it should still move down to ensure an adequate grasp. Figure 4.14 shows the physical appearance of the logic implemented, where in the undesired scenario the gripper still does not have enough pressure to hold the box firmly. Then it should move further down and provide a firm grasp as seen in the desired scenario where the suction cups have a stronger hold of the cardboard box. Another practical observation noted when testing grasping strategies was that the use of vacuum before the grasping contact provided more successful results as it gave a more stable surface contact between the suction cups and the surface, rather then activating them after the contact, which could cause small leaks of air and therefore reducing the success of the grasping.

Main Script

In figure 4.15, an illustration of the node structure shows the interaction between nodes when performing the whole integrated operation. The main MoveIt2 sends a text command to the LLMs in the Ollama LLMs in string format and receives another string from it, which is the object description that is used in the image segmentation. This is string is sent, however through the Pose Estimation service, which means that those two services are chained to each other. Every time the Pose Estimation service is called it automatically calls the image segmentation service from LANG-SAM, and uses the segmentation mask to run RANSAC and estimate the pose of the object. The box's dimensions are also estimated as explained previously, and these dimensions are then used in the stack optimisation service. This service responds with the optimal pose for the given box according to the objective function. The gripper is also used to pick up the boxes and release them, each channel of the gripper can receive a certain vacuum level command and their actual vacuum levels can also be read in order to enable the grasping logic.

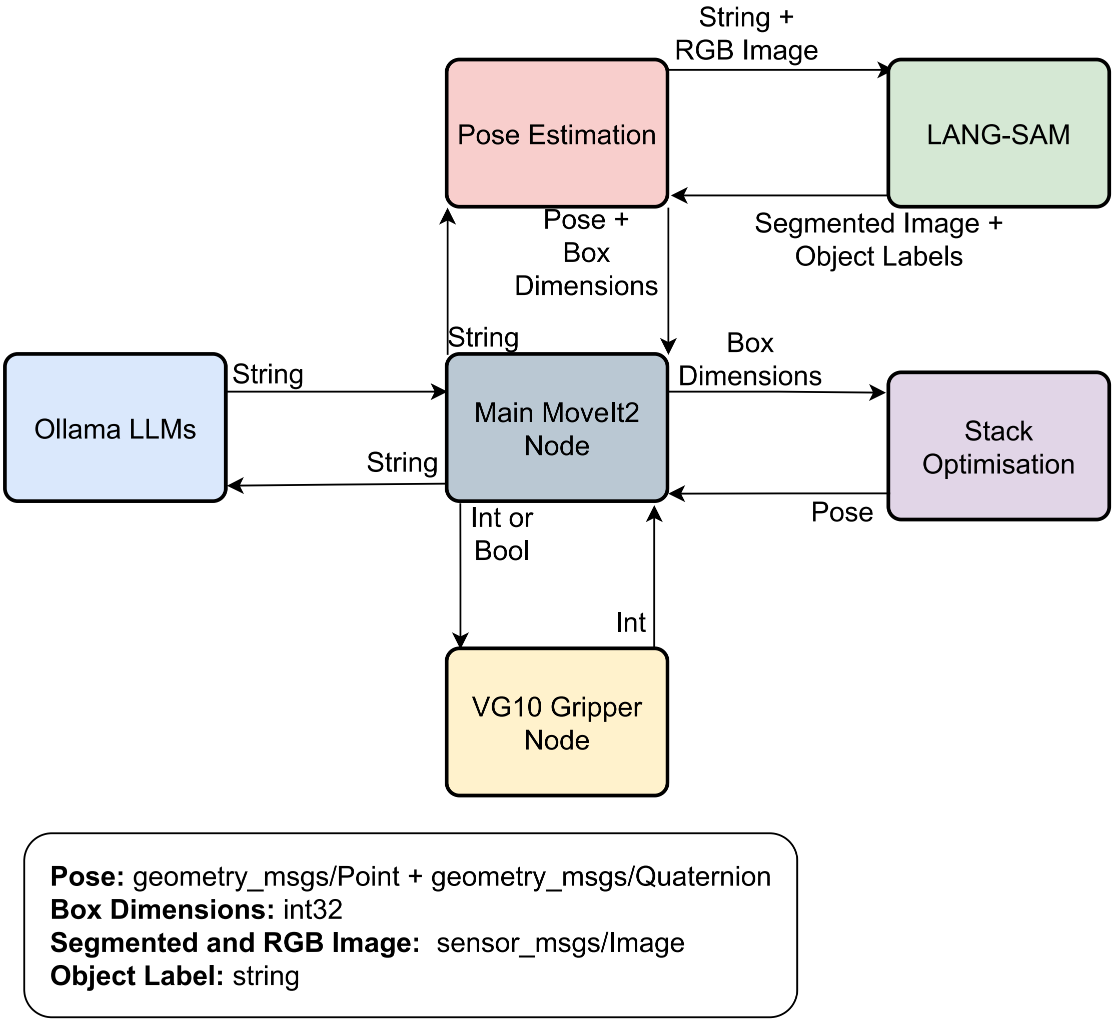


Figure 4.15: Main script node diagram with all of the custom node interactions used in order integrate the depalletizing system.

Figure 4.16 shows the main four service nodes and their sequence in the main script. From the extracted object prompt from the LLMs to the stack optimisation box pose. This figure illustrates how the sequence works in a step-by-step manner and also provides more concrete examples of the information that is being sent from one node to the other, without having the main MoveIt2 node as a middle-man, but only the other nodes in the script, and how they interact with each other.

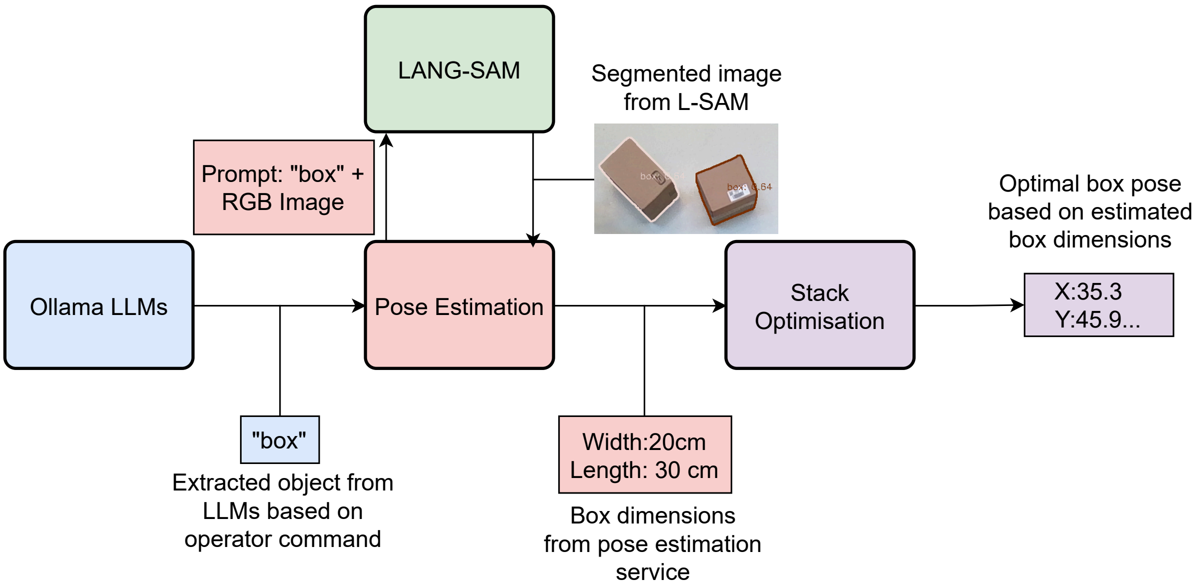

LANG-SAM
Segmented image from L-SAM
Prompt: "box" + RGB Image
Optimal box pose based on estimated box dimensions
Ollama LLMs
Pose Estimation
Stack Optimisation
X:35.3
Y:45.9...
"box"
Extracted object from LLMs based on operator command
Width:20cm
Length: 30 cm
Box dimensions from pose estimation service

Figure 4.16: Main script sequential flow for each of the nodes in order of processing in the main script. The main MoveIt2 node is omitted for illustration purposes.

It is important to note that figure 4.16 is a simplification of the pipeline and some information may be omitted for illustration and clearness purposes. As for example, the optimal box pose generation also take into account the pallet's dimensions and support constraints.

5 Testing

The system after being developed needs to be tested in order to evaluate its performance. As it is composed of subsystems, each of them will be tested and evaluated individually, however a fully integrated system test will also be performed, with the focus on the Vision capabilities of the system as these are the ones that can present inconsistent behaviour. Other implemented parts such as stack optimisation and grasping are assumed to work. The test involving all of the necessary components for an operation is named Vision System Test. The LLM dual system will also be separately tested in this section to evaluate its object extraction, even though a full system integration has occured and been documented. Videos for the tests are available on the Appendix A, and the author's YouTube channel, as well as the code made in the project.

5.1 Vision System Test

There will be 4 main divisions for the vision system of the proposed solution, from one to four boxes. The test case is defined as following: One or more boxes will be placed in the FOV of the RGB-D camera and the vision system will both segment and estimate the dimensions of the box and its 6D pose. As mentioned previously, the depth data is not seen as reliable so the estimation is not strictly evaluated. The segmentation is though required to be sufficient in order to make a pick and place operation from the ground on to the pallet. From the segmentation it is crucial to find the top plane of the box in order to pick it up, if both conditions are fulfilled. The test is seen as successful, if the box segmentation and pose- and dimension estimation are sufficient to place them on the pallet. This also assumes all of the other parts of the system such as grasping and stack optimisation to be working. The tests are:

- Vision System Test 1 box
- Vision System Test 2 boxes
- Vision System Test 3 boxes
- Vision System Test 4 boxes

The boxes that are going to be tested are as present in figure 5.1:

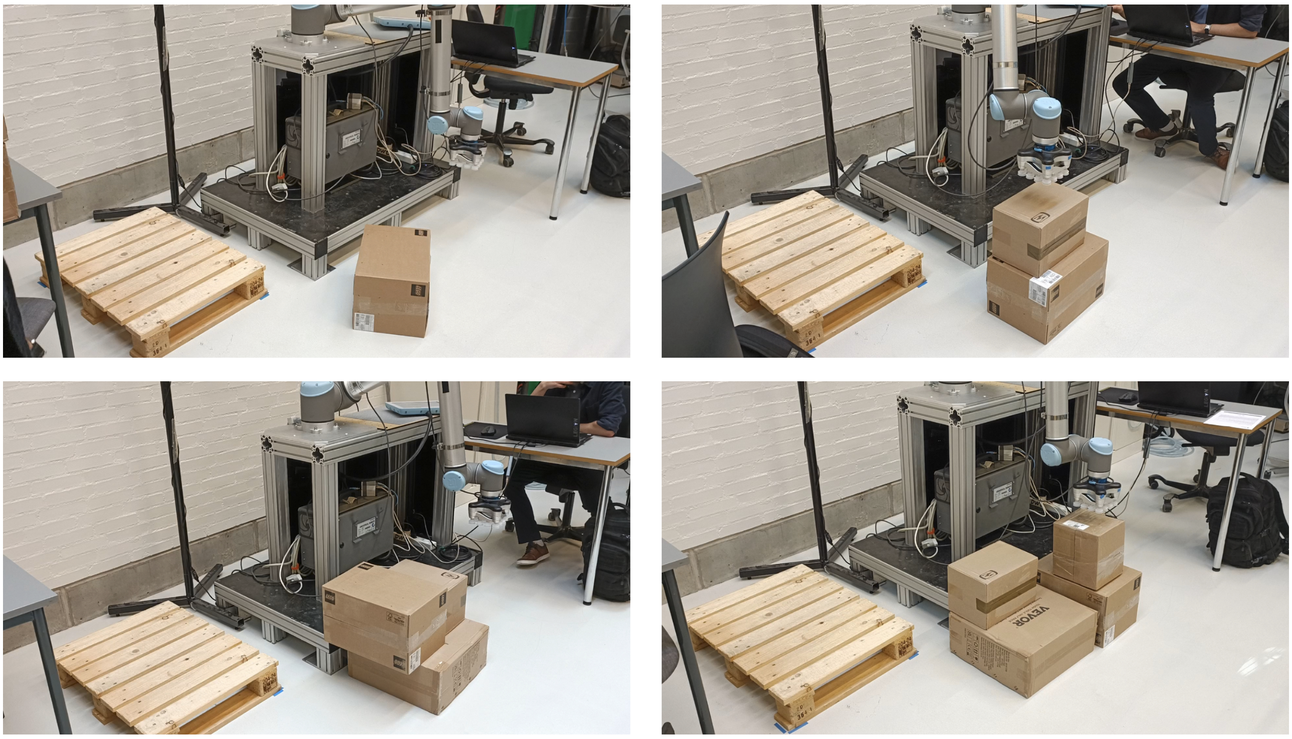


Figure 5.2: Example of the four testing scenarios. Top left is an example with a single box, on the top right two boxes are presented for the scenario with two boxes. Bottom left and bottom right images are respectively scenarios with three and four boxes.

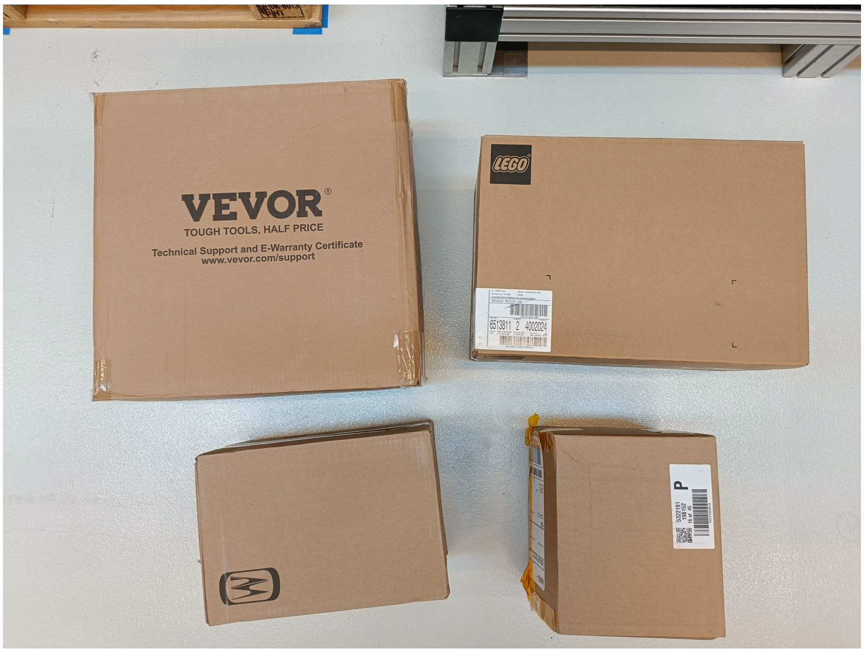


Figure 5.1: Testing boxes for the pick and place operation with the vision system.

An example of each test scenario can be seen in figure 5.2, where the the stack of boxes increases one box at a time up to 4 boxes. This test also shows the full system working as to perform the pick and place operation each of submodules of the whole integrated system must work.

Results

The results for the vision system tests are shown in table 5.1 and it is important to mention the test with 3 boxes (new) and 4 boxes(new), these were done testing a new method for choosing the highest plane of the same segmented object. The new approach showed promising results and will be interpreted in the Discussion section of the thesis.

Table 5.1: Integrated Vision Test Results

Test Group	Boxes/Test	Iterations	Total Boxes	Successful	Failed	Success Rate (%)
1 Box	1	40	40	40	0	100.0
2 Boxes	2	10	20	20	0	100.0
3 Boxes	3	10	30	23	7	76.7
3 Boxes (New)	3	10	30	30	0	100.0
4 Boxes (New)	4	5	20	20	0	100.0
Total (All)	–	–	**140**	**133**	**7**	**95.0**
Total (No ”New”)	–	–	**90**	**83**	**7**	**92.2**
Total (1, 2, 3N, 4N)	–	–	**110**	**110**	**0**	**100.0**

As mentioned previously, the grasping and stack optimisation were expected to work and indeed did for all of the iterations. An example of stack optimisation visualisation and grasping can be seen in the testing videos. In figure 5.3 a plot with four stacked boxes is seen, there id a total of four processed boxes as in the 4 Boxes test. The grasping and the stack optimisation were reliable in each test iteration.

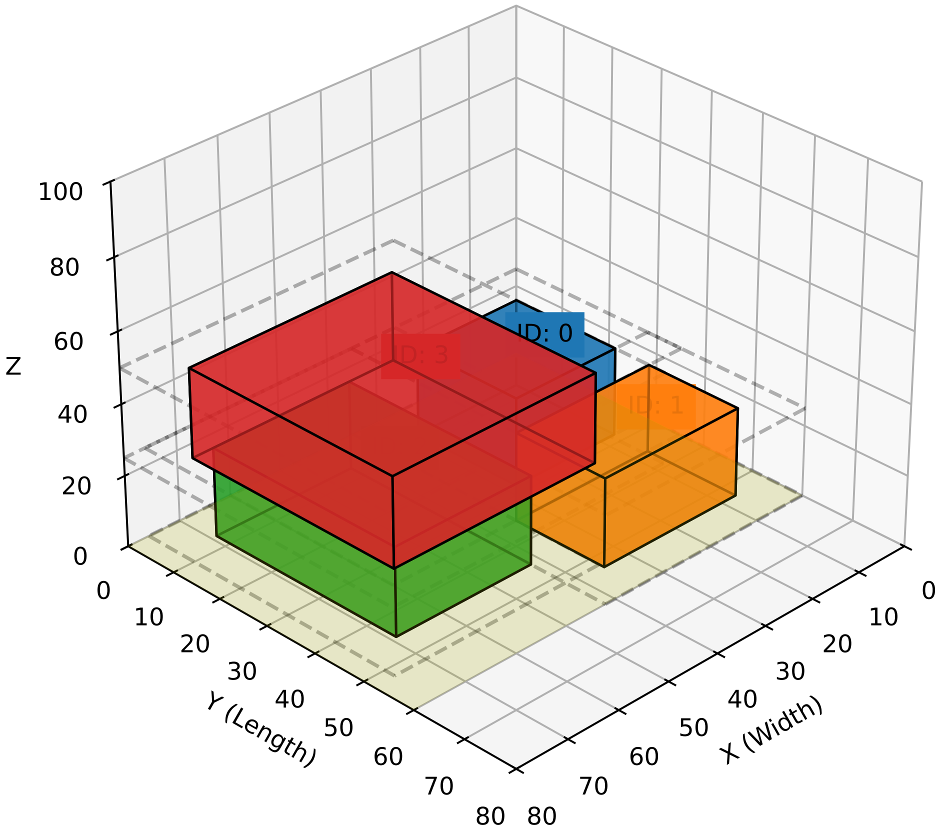


Figure 5.3: Example of visualisation of stack optimisation for the four boxes. The biggest box is on top due to it being picked up last.

5.2 Pose Estimation

The following test was made separately from the previous one in order to only evaluate the pose estimation accuracy of the system and also the dimension estimation. It is though important to mention that the number of the table do not necessarily reflect reality as assumptions are made. A perfect measurement would require a simulation environment, and even still would not translate the same to reality. There are two metrics being used to calculate the accuracy of the system's pose estimation. These are the estimated width and estimated height. These are then used to calculate the distance to the mid point of the rectangle composed by width and height against the ground truth width and height. The equations can used for the calculations can be seen in equations 5.1 to equation 5.6.

$$\Delta W = \frac{|\text{Ground Truth Width} - \text{Estimated Width}|}{2} \tag{5.1}$$

$$\Delta H = \frac{|\text{Ground Truth Height} - \text{Estimated Height}|}{2} \tag{5.2}$$

$$\Delta x = \left| \frac{\text{Ground Truth Width}}{2} - \frac{\text{Estimated Width}}{2} \right| \tag{5.3}$$

$$\Delta y = \left| \frac{\text{Ground Truth Height}}{2} - \frac{\text{Estimated Height}}{2} \right| \tag{5.4}$$

$$D = \sqrt{(\Delta x)^2 + (\Delta y)^2} \tag{5.5}$$

$$D = \sqrt{\left(\frac{\text{Ground Truth Width}}{2} - \frac{\text{Estimated Width}}{2} \right)^2 + \left(\frac{\text{Ground Truth Height}}{2} - \frac{\text{Estimated Height}}{2} \right)^2} \tag{5.6}$$

Each of the four test boxes were measured and these numbers represent the ground truth of the top face. The pose estimation was then performed and the numbers were calculated in deviation of the given ground truth against the actual values. The results assume perfect rotation in the boxes' pose estimation which is not always an accurate assumption. The numbers assume therefore an error in the 3D dimensions of the box by making the assumption that both the estimated and ground truth boxes have the same start coordinate at a local coordinate system and then their midpoints distance is calculated.

Results

The results of pose estimation for each of the boxes are now presented and the threshold proposed by the work in [21] and[20] of 15 mm and 30 mm maximum deviation, seen in the Related Works section of the thesis, are going to be used.:

Table 5.2: Ground Truth and Estimated Dimensions for Box 1 (10 Images)

Dimension Type	**Width (mm)**	**Height (mm)**
Ground Truth (all)	**397.0**	**275.0**
Estimated Image 1	388.4	272.5
Estimated Image 2	388.7	265.4
Estimated Image 3	381.8	267.9
Estimated Image 4	383.1	264.7
Estimated Image 5	399.9	264.0
Estimated Image 6	382.7	261.1
Estimated Image 7	391.8	265.7
Estimated Image 8	378.2	261.5
Estimated Image 9	385.1	274.7
Estimated Image 10	379.7	258.5

Table 5.3: Box 1 – Dimension Differences and Midpoint Shifts Over 10 Captures

Image #	**Width Diff (mm)**	**Height Diff (mm)**	**Midpoint ΔX (mm)**	**Midpoint ΔY (mm)**	**Euclidean Dist (mm)**
1	4.30	1.25	4.30	1.25	4.48
2	8.30	9.60	4.15	4.80	6.34
3	15.20	7.10	7.60	3.55	8.39
4	13.90	10.30	6.95	5.15	8.65
5	2.90	11.00	1.45	5.50	5.69
6	14.30	13.90	7.15	6.95	9.97
7	5.20	9.30	2.60	4.65	5.32
8	18.80	13.50	9.40	6.75	11.57
9	11.90	0.30	5.95	0.15	5.95
10	17.30	16.50	8.65	8.25	11.95
Average	**11.21**	**9.28**	**5.82**	**4.70**	**7.83**
Minimum	**2.90**	**0.30**	**1.45**	**0.15**	**4.48**
Maximum	**18.80**	**16.50**	**9.40**	**8.25**	**11.95**

Table 5.4: Ground Truth and Estimated Dimensions for Box 2 (10 Images)

Dimension Type	Width (mm)	Height (mm)
Ground Truth (all)	**300.0**	**205.0**
Estimated Image 1	262.5	181.2
Estimated Image 2	254.1	190.8
Estimated Image 3	279.4	188.5
Estimated Image 4	255.4	177.1
Estimated Image 5	277.1	156.0
Estimated Image 6	289.7	186.6
Estimated Image 7	292.1	177.0
Estimated Image 8	280.3	181.8
Estimated Image 9	288.7	186.2
Estimated Image 10	281.8	184.4

Table 5.5: Box 2 – Dimension Differences and Midpoint Shifts Over 10 Captures

Image #	Width Diff (mm)	Height Diff (mm)	Midpoint ΔX (mm)	Midpoint ΔY (mm)	Euclidean Dist (mm)
1	18.75	11.90	18.75	11.90	22.20
2	22.95	7.10	22.95	7.10	24.03
3	10.30	8.25	10.30	8.25	13.19
4	22.30	13.95	22.30	13.95	26.30
5	11.45	24.50	11.45	24.50	27.04
6	5.15	9.20	5.15	9.20	10.54
7	3.95	14.00	3.95	14.00	14.55
8	9.85	11.60	9.85	11.60	15.22
9	5.65	9.40	5.65	9.40	10.97
10	9.10	10.30	9.10	10.30	13.75
Average	**11.95**	**12.02**	**11.95**	**12.02**	**17.78**
Minimum	**3.95**	**7.10**	**3.95**	**7.10**	**10.54**
Maximum	**22.95**	**24.50**	**22.95**	**24.50**	**27.04**

Table 5.6: Ground Truth and Estimated Dimensions for Box 3 (10 Images)

Dimension Type / Image #	Width (mm)	Height (mm)
Ground Truth (all)	**430.0**	**415.0**
Estimated Image 1	404.3	393.7
Estimated Image 2	410.9	408.7
Estimated Image 3	407.9	402.7
Estimated Image 4	392.9	376.5
Estimated Image 5	418.3	399.0
Estimated Image 6	407.4	404.0
Estimated Image 7	418.5	356.1
Estimated Image 8	399.6	382.0
Estimated Image 9	405.8	395.3
Estimated Image 10	403.6	397.8

Table 5.7: Box 3 – Dimension Differences and Midpoint Shifts Over 10 Captures

Image #	Width Diff (mm)	Height Diff (mm)	Midpoint ΔX (mm)	Midpoint ΔY (mm)	Euclidean Dist (mm)
1	18.15	5.35	18.15	5.35	18.91
2	9.55	3.15	9.55	3.15	10.05
3	13.65	3.55	13.65	3.55	14.10
4	18.55	19.25	18.55	19.25	26.72
5	5.85	8.00	5.85	8.00	9.91
6	11.30	5.50	11.30	5.50	12.57
7	5.75	29.45	5.75	29.45	30.01
8	15.20	16.50	15.20	16.50	22.42
9	12.10	9.85	12.10	9.85	15.60
10	13.20	8.60	13.20	8.60	15.75
Average	**12.53**	**10.82**	**12.53**	**10.82**	**17.51**
Minimum	**5.75**	**3.15**	**5.75**	**3.15**	**9.91**
Maximum	**18.55**	**29.45**	**18.55**	**29.45**	**30.01**

Table 5.8: Ground Truth and Estimated Dimensions for Box 4 (10 Images)

Image #	Estimated Width (mm)	Estimated Height (mm)
1	224.6	196.7
2	223.3	187.4
3	221.3	189.4
4	225.7	221.4
5	215.8	185.6
6	213.8	195.1
7	197.2	185.9
8	239.6	191.5
9	226.7	175.4
10	243.6	223.0

Table 5.9: Box 4 – Dimension Differences and Midpoint Shifts Over 10 Captures

Image #	Width Diff (mm)	Height Diff (mm)	Midpoint ΔX (mm)	Midpoint ΔY (mm)	Euclidean Dist (mm)
1	5.20	4.15	5.20	4.15	6.65
2	5.85	8.80	5.85	8.80	10.57
3	6.85	7.80	6.85	7.80	10.38
4	4.65	8.20	4.65	8.20	9.42
5	9.60	9.70	9.60	9.70	13.64
6	10.60	4.95	10.60	4.95	11.70
7	18.90	9.55	18.90	9.55	21.17
8	2.30	6.75	2.30	6.75	7.13
9	4.15	14.80	4.15	14.80	15.37
10	4.30	9.00	4.30	9.00	9.97
Average	**7.14**	**8.58**	**7.14**	**8.58**	**11.00**
Minimum	**2.30**	**4.15**	**2.30**	**4.15**	**6.65**
Maximum	**18.90**	**14.80**	**18.90**	**14.80**	**21.17**

Summary of the euclidean distance threshold for the boxes for up to 15 mm and 30 mm can be seen in table 5.10.

Table 5.10: Euclidean Distance Threshold Summary Across Boxes

Threshold	Box 1	Box 2	Box 3	Box 4	Total
Up to 15 mm	10 (100%)	5 (50%)	4 (40%)	8 (80%)	27 (67.5%)
Up to 30 mm	10 (100%)	10 (100%)	9 (90%)	10 (100%)	39 (97.5%)

5.3 Extra Data on Image Segmentation and Pose Estimation

The total of images used in the testing section both for the test with the 4 boxes and pose estimation were evaluated as well as some images while still developing other parts of the system. There were counted the total number of instances present in the image where in cases of boxes stacked on each other were counted as one single instance as some could be occluded due to the stacking and the camera angle. Table 5.11 shows the results for the segmentation with the total number of objects and correct segmentation and its percentage. A percentage of 99.1% was achieved as seen in the referred table.

Table 5.11: Segmentation Accuracy Summary

Metric	**Count**	**Percentage**
Total Objects Present	341	100%
Correctly Segmented	338	99.1%
Incorrectly Segmented	3	0.9%

Effects of tape on Pose Estimation

Another relevant observation was gathered when exploring the the effects of tape in the object segmentation and pose estimation outside of the performed tests, as tape can cause the depth information to be unreliable. The pose of boxes of two different heights were seen as having the same plane and making a wrong Pose Estimation. This was not present in any of the tests, however it is seen as a valuable insight.

L-SAM Capabilities

As the focus of the project is on depalletizing with the help of intelligent vision, some examples of the L-SAM capabilities are going to be shown. The main purpose behind this test is to highlight the abilities of flexible interaction between humans and robot, where it is possible to specific about an object and receive ”out of the box” segmentation based on the operator/user description. In figure 5.4, it can be seen that the model only segments the top left package according to the user specification ”big package”. None of the other three packages were segmented due to the model having dimension understanding.

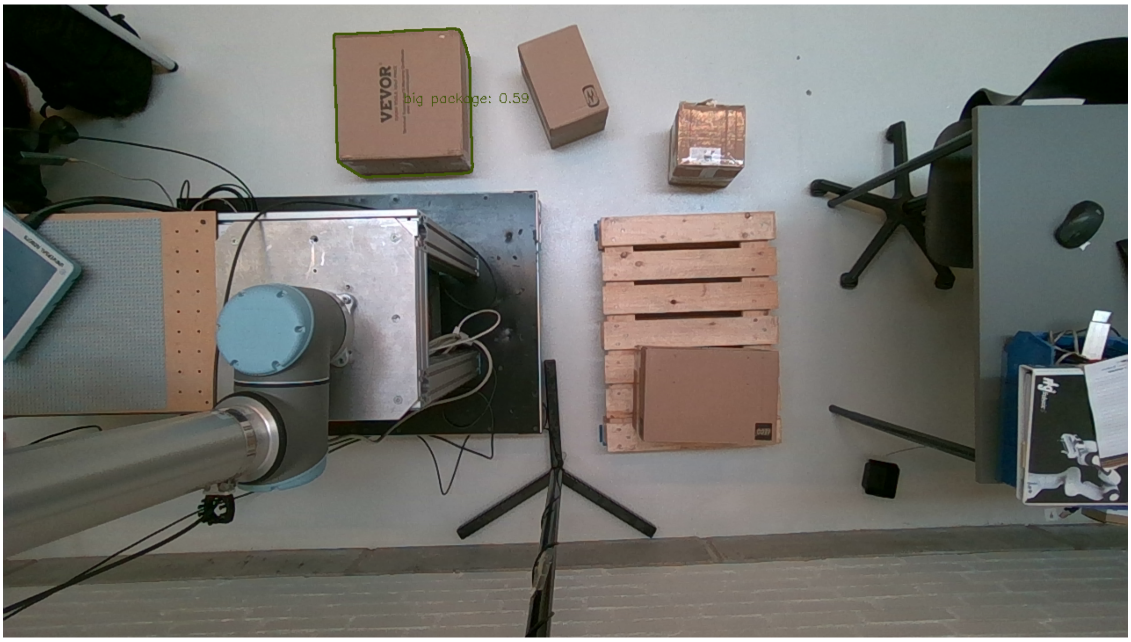


Figure 5.4: The only package segmented in the image is the big package, where the other three are not options due to them not being ”big”.

The other opposite example would be the user specifying a ”small package” if there were the need of it. In figure 5.5, only the top right package is segmented as small package. This is another example of L-SAM having spatial understanding when segmenting objects.

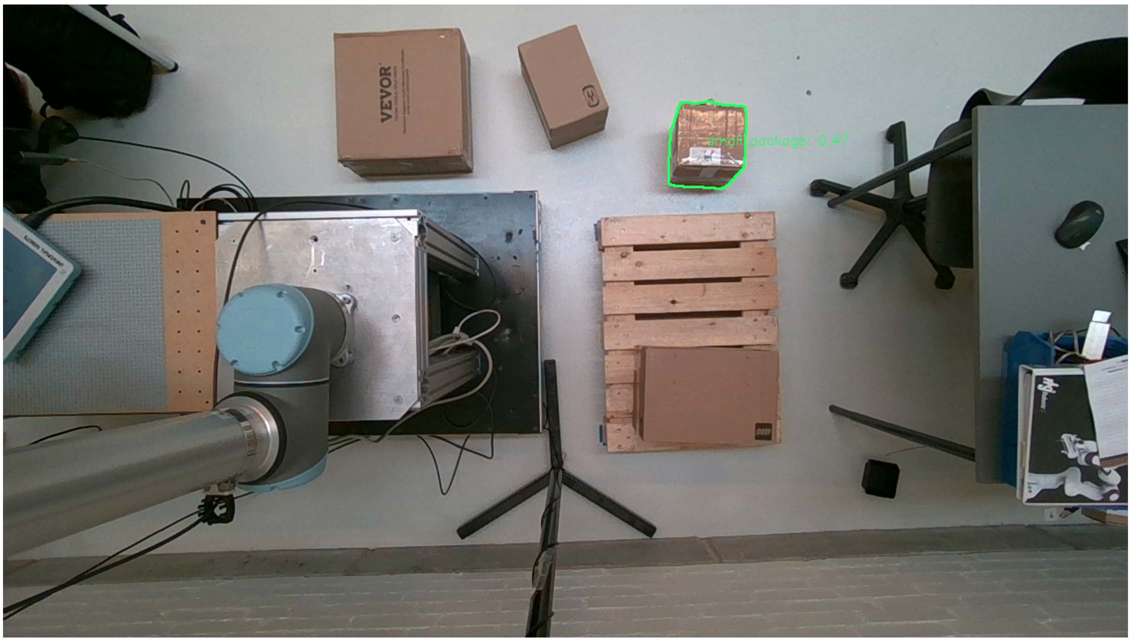


Figure 5.5: L-SAM is capable of taking object descriptions such as ”small” into consideration when segmenting the objects, as only the smallest package is segmented.

In the case of using colours as identifiable features L-SAM also provides the desired segmentation. The user prompt is: ”white package”. and the only package segmented is the white package on the top right of the four packages. It is also possible to use colours for identifying objects, which allows for a closer human-robot interaction. Figure 5.6 shows the case with L-SAM only segmenting the white package.

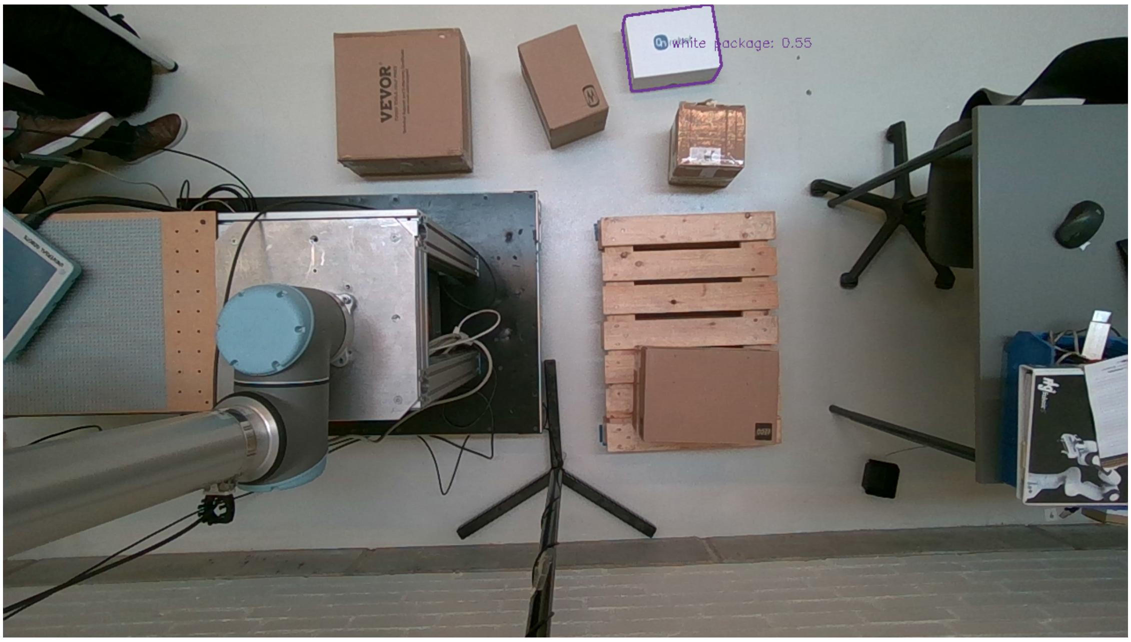


Figure 5.6: L-SAM is able to use attributes like colour to segment objects. The prompt ”white package” was used and only the white package was segmented of the five packages present in the image.

Another example of how L-SAM operates is seen in figure 5.7, where four of the five boxes are segmented whereas the one box on the pallet is not segmented. This can be attributed to the similar colours of the pallet and the box, also by the fact of the box not being close enough to the camera, which makes it more difficult to segment the box. This can also be used to highlight the importance of camera placement and angle in relation to one’s workspace and robot, as different angles create different images, segmentations might differ from one angle/placement to another.

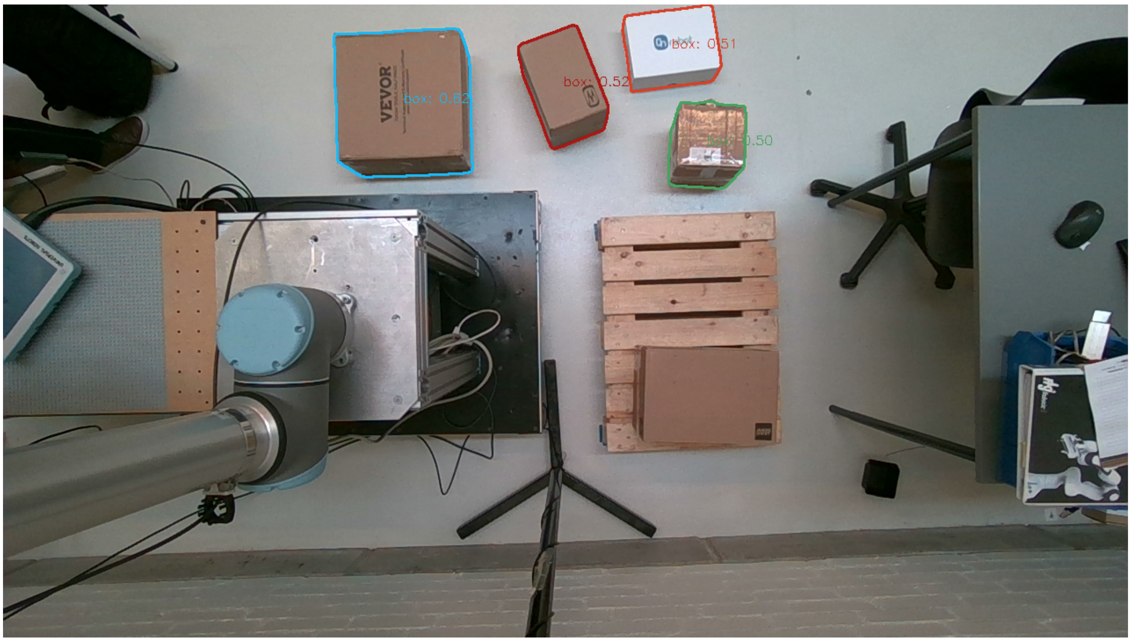


Figure 5.7: L-SAM segmenting four packages laying on the ground regardless of colour and size. There is though one package on the pallet that is not being segmented due to the low contrast to the background, making it seem as part of the pallet.

5.4 LLMs Object Extraction Test

The system is also composed of a dual LLM architecture made for robust language understanding and object extraction. These are now going to be evaluated across five different categories:

- Basic Object Identification
- Object + Attribute Extraction
- Multiple Object References
- Negation and Exclusion
- Noise / Extra Clauses

As the main task of the dual LLM system is to extract objects and their characteristics. This capability will be tested to determine whether it is reliable in all of the five scenarios.

The **Basic Object Identification** refers to the most simple case where the operator writes the command with simply the object name and no additional description. For example:*"can you get me the packages please?"* and the system should then extract: *"package."* as the object of interest. As it is basic object identification no adjectives are used to describe the object.

The **Object + Attribute Extraction** category refers to the prompts that do have an object attribute to be extracted. This attribute is simply a description of the object, as a colour or shape or any characteristic visually seen. An example of input prompt could be: *"can you get me the red box?"*. The system should then extract *"red box."* as the object of interest.

Multiple Object References is the category in which the operator asks for different objects once and all of them should correctly be extracted. An example would be: *"Can you take the big box and the small box and place them on the pallet?"*. In this case the system should extract: *"big box. small box."*

The **Negation and Exclusion** test refers to prompts where the system is challenged by misleading prompts. One example would be:*"can you get me the white box and the green one but not actually the green one just the white one."*. In this case the operator starts by asking to retrieve both the white and green boxes, however he realizes it does not need any more the green box. This could simulate a voice command where the operator could save time by simply adding the last part rather than starting a command from beginning. In this case the object extracted should be: *"white box."*.

The **Noise / Extra Clauses** is a category where the prompts are made to mislead the system and test its ability to discern of useful and not useful information. An example would be: *"The red box is not needed right now. Please retrieve the blue box."*. In this example the red box is not needed and is there to make noise and try to mislead the system. The dual LLM system should then extract: *"blue box."* and not be mislead by the prompt.

Results

A total of 10 prompts per category were crafted to test the system and the results can be seen in table 5.12. All of the text prompts are available on the Appendix B.

Category	**Number of Prompts**	**Successful Extractions**	**Success Rate (%)**
Basic Object Identification	10	10	100%
Object + Attribute Extraction	10	10	100%
Multiple Object References	7	10	70%
Negation and Exclusion	10	10	100%
Noise / Extra Clauses	9	10	90%
Total	46	50	**92%**

Table 5.12: LLM Object Extraction Results by Category

6 Discussion and Conclusion

The performed tests will be used as a base for evaluation of the proposed system. The Image segmentation and pose estimation results will be interpreted as well as the LLM object extraction tests. The achievements of the project will be mentioned and a conclusion to the thesis will then be stated.

6.1 Object Segmentation and Pose Estimation

The developed depalletizing system uses vision in order to segment and estimate the pose of the packages. The system was initially designed to only handle the estimation of poses of different object instances. As seen in figure 6.1, the initial design for the system considered only the scenario where different object instances were available and the highest plane would be chosen for the pose estimation. The test referred is the ”3 boxes” category seen in table 5.1, before the new logic was implemented.

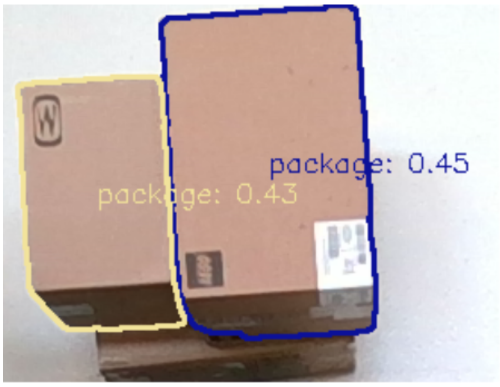


Figure 6.1: Two different segmentation instances in the image, which makes simpler to find the boxes’ top planes.

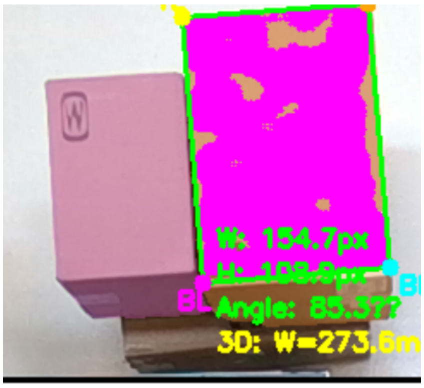


Figure 6.2: Pose estimation of the highest box as the planes are calculated based on the segmentation masks of each instance.

There was though the possibility of different objects sharing the same mask as seen in figure 6.3, this mask is also from the ”3 Boxes” test category on table 5.1. The system was though able to handle these scenarios as seen in figure 6.4.

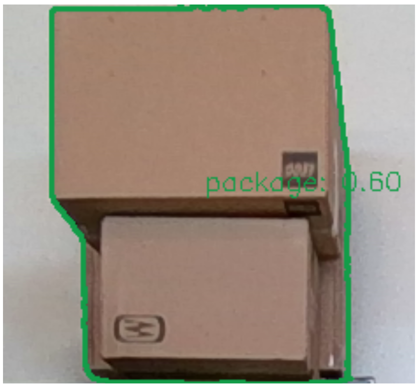


Figure 6.3: Example of test data with a single segmentation mask for three object, two of which are in the foreground.

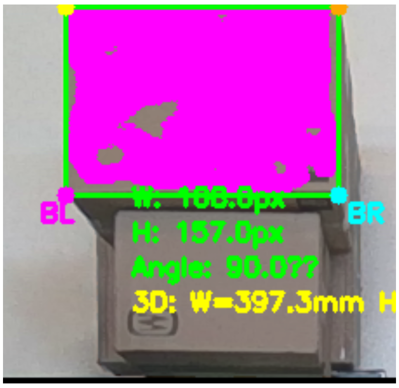


Figure 6.4: Example of test data with a single segmentation mask for three object, two of which are in the foreground, and the pose estimation succeeds in finding the highest plane.

The success seen in figure 6.4 was though not always repeated due to the first system approach selected. The system could discern between objects of different segmentation masks however if they shared the same mask, the biggest plane in terms of inliers would be chosen. This was the reason figure 6.3 showed a successful result, however in cases where to highest plane was not the one with most inliers the boxes would not be correctly chosen. This was though refined after on tests ”3 Boxes (New)” and 4 Boxes (New)”.

A new logic was implemented for the the ”New” tests from table 5.1. An extra layer of decision was implemented in order to take into account not very specific segmentation masks. The simple solution was to choose all planes with more than 3000 inlier points and also increase the number of iterations to 200 to ensure RANSAC on the whole mask. From there, the planes would be ordered by height and the highest one would be set as base of choice, planes with average height ±0.5 cm to the highest plane's height were evaluated and the one with the most inliers between the highest planes was then chosen as the one. As seen in table 5.1, the new approach proved to be robust in the testing conditions.

This approach of choosing the highest plane for each object showed an interesting behaviour once when the effects of tape were being evaluated. Due to the tape on the box making the depth data unreliable a wrong plane was found that involved two different boxes. This exploration of the effects of tape were not a part of the system tests, but they have shown a potential flaw in the system. The shortcoming being the difficulty in discerning between objects that share the same height, since if they share the same segmentation mask, as they might do in some cases, then they would both be seen as one single object by the pose estimation. This could be fixed with a one segmentation mask for each object, a wrist camera on the UR10 could be used to take closer pictures of the boxes and then differentiate each object. Figure 6.5 shows the effects of objects sharing same segmentation mask on the pose estimation if they are detected as being part of the same plane. The object had a height difference of 10 cm and this did not happen the object was not taped, since the depth data was then more reliable.

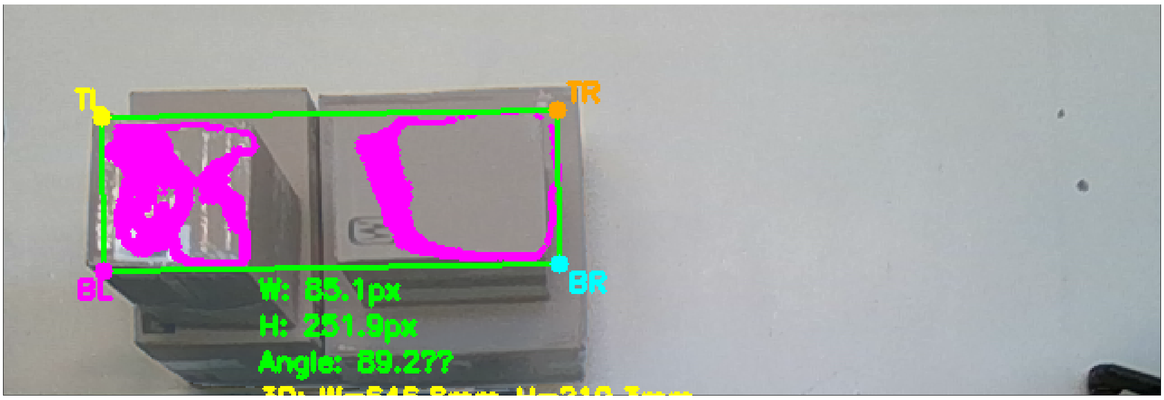


Figure 6.5: Two different boxes sharing the same segmentation mask resulting in a failed planed detection due to the tape making the depth data unreliable. This shows however a short coming in handling boxes of same height in which share a segmentation mask.

It is also important to mention the pose estimation results from table 5.10 and make its interpretation. The pose estimation assumed some conditions in order to retrieve some data for evaluation, however this data is not 100% correspondent to reality as assumptions were made. The results serve as a basis of evaluation and are not perfect representations of reality. The results were however promising under the assumptions taken with 97.5% of results below the threshold of 30 mm deviation, with the only result being 0.01 mm above said threshold. These numbers still do not take into account rotation and misplacement as mentioned in the test description. The numbers are probably lower than 97.5% percentage but are difficult to perfectly measure in the real world, therefore, assumptions were made and results are only an estimate of what the actual number is.

Another interesting observation to make regarding the vision system is on what is necessary to properly segment objects. From the segmentation data in 5.11, there were only 3 instances, classified as fail, and all of them share similar traits. Lack of background contrast and low depth perspective. Figure 6.6 shows the unsuccessful case where only the top box on the left is segmented. However all of the other boxes in the image are not well positioned in relation to the camera and themselves. Their colours are similar and there is not a perspective of an object of significant size they are similar to ”stains” on the floor and not actual boxes. This indicates the importance of camera angle when designing the system, however it also shows that in challenging scenarios the segmentation might fail. A very similar image was taken and can be seen in figure 6.7. For this image the segmentation was though successful in segmenting all objects in the image. The package on the bottom right was segmented as part of the bigger package under it due to the angle of the image making them seem as one.

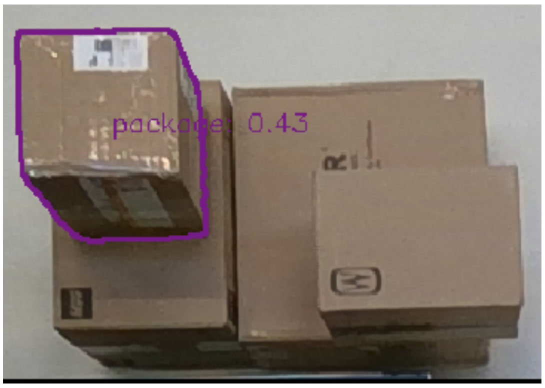


Figure 6.6: Example of failed segmentation due to lack of contrast and almost none depth perspective making the objects not being segmented.

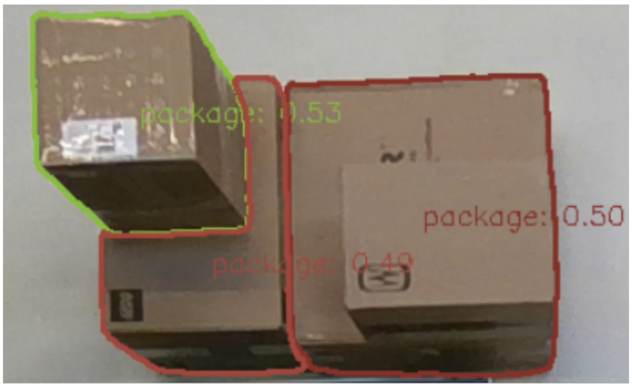


Figure 6.7: Example of successful segmentation even though there is a lack of contrast and almost none depth perspective.

The solution’s shortcoming were presented, however the results from tables 5.1, 5.10, 5.11 show robust system performance in regards of vision.

6.2 LLMs Analysis

A crucial and novel part of the presented depalletizing system is human-robot interaction. The use of the dual LLM structure allows the system to take operator commands and combine them with SAM and GroundingDINO to pick up different objects base don their attributes. As seen in the testing section of the thesis, L-SAM was able to discern objects based on their size and colour, making it flexible for the user without the need of fine-tuning or further training. The object and its attributes have to be extracted from the LLM system and the results from 5.12 have shown a 92% success rate in the object extraction. There were 4 unsuccessful object extractions in a total of 50 prompts. The only categories where success rate was not 100% were the **Multiple Object References** and the **Noise/Extra Clauses.** In all of the unsuccessful object extractions the objects were referenced subtly, which mislead the LLMs. Here is an example:

User Input

```
Can you move both the fragile package and the red one?
```

Extractor LLM Reasoning

Okay, the user is asking if I can move both the fragile package and the red one. Let me break this down.

First, I need to check the items mentioned. The fragile package and the red one. The user wants to move both. But wait, the instructions say to respond with the description of the item(s), using the singular form of the words. So even if there are two items, I should list them as singular.

The fragile package is a single item, so I should mention ”fragile package.” The red one is another single item, so ”red box” if it’s a box, but the user said ”red one,” which might be a package. Wait, the original command was ”move both the fragile package and the red one.” The user might be referring to the red package, but the wording is ”red one,” which could be ambiguous. However, since the example given in the instructions uses ”red box” when there’s a plural, maybe ”red one” here is singular. But the user might be referring to a package. But the assistant should take the user’s words as given. So ”the red one” is singular, so ”red one” would be the item. However, in the examples, when they say ”red box,” it’s singular. So if the user says ”red one,” it’s singular. But the assistant should list the item as described. So the response would be ”fragile package. red one.” But the user might be referring to a package, but the assistant has to follow the exact wording. So the answer is ”fragile package. red one.” But wait, the user might have meant a box, but the instruction says to use the singular form of the word. So if the item is a package, then ”package” is singular. So the answer would be ”fragile package. red one.” But the user might have intended ”red box,” but the instruction says to use the singular form. So the assistant should stick to the user’s exact words. Therefore, the response is ”fragile package. red one.”

Extractor LLM Final Answer

```
fragile package.  red one.
```

Validator LLM Reasoning

`<think>` Okay, let’s see. The user asked to move both the fragile package and the red one. The extracted items are ”fragile package.” and ”red one.”

First, I need to check if the extracted items are in singular form. The examples show that even if the command uses plural, the extracted items should be singular. So ”fragile package” is already singular, and ”red one” is also singular. That’s correct.

Next, the user’s command specifies two objects: ”fragile package” and ”red one.” The extracted items match exactly. So the validation should be ”VALID: true.”

I should make sure there’s no mistake in the singular form. The examples confirm that ”package” becomes ”box” when singular, but here ”package” is already singular. Similarly, ”one” is correct as in the example ”red one.”

No objects were specified in the command, but here both are specified. The response should be ”VALID: true” because the extracted items match what was requested. The user didn’t say ”no objects” but rather specified two, so it’s valid. `</think>`

Validator LLM Final Answer

```
VALID: true Explanation:  The extracted items "fragile package" and "red
one" match the user’s request.  Both are in singular form as required,
and no objects were omitted.
```

This conversation and the reasoning behind the LLMs' answers provide meaningful insights on where the object extraction failed. In the "Extractor LLM", the one whose role is to extract the objects and their attributes ends extracting the wrong object name. It extracts *"red one"* instead of *"red package"*. This happened due to the red package being referenced as red one and the LLM being unable to make the connection to it as a package, it even contemplates that possibility but chooses to extract *"red one"*. The Validator LLM also accepts the extraction as correct without realising the actual meaning behind the word *"one"*. There were, however, a significant number of successful extractions. One example can be seen as the following:

User Input

```
Can you get me the white box and the green one but not actually the green
one, just the white one.
```

Extractor LLM Reasoning

> Okay, let's see. The user is asking for the white box and the green one, but they want only the white one, not the green. So first, I need to identify the objects they're referring to. The white box is clear, and the green one is another box. But since they specified not to take the green one, I should only retrieve the white box. The response should mention the white box and clarify that the green one isn't taken. I need to make sure the answer is concise and follows the formatting rules. Also, check if the plural words require singular forms, but in this case, "box" is singular. So the response should be "white box."

Extractor LLM Final Answer

```
white box.
```

In this case the Extractor LLM is able to understand that the green box is not to be picked and only the white box should be extracted. This conversation also shows the ability to understand object references such as *"the white one"*, which were not fully understood by context in the unsuccessful prompts. This case the reference was probably understood due to the white box being explicitly stated as white box before the reference occurred. The successful and unsuccessful cases show that the system prompts of the dual LLM system has margin for improvement. Greater context for them as to ensure only physical objects are able to be extracted is a possible point of improvement. More examples so references can already be seen in the system prompts are also a possibility to make the system more robust in the challenging scenarios. The dual LLM structure is seen as an effective way to correctly retrieve information from the user and has shown promising results. The characteristics of the objects were extracted in 100% of the cases where the only 4 unsuccessful extractions had the name of the object wrong, for example, *"red one."*. This means that the object attributes can reliably be extracted, however the actual object names need improvement in case of references being used.

6.3 Thesis Achievements

The developed solution is seen as a product that can be further improved on, however the body of work of the thesis is seen as successful taken into account the time constraints. A ROS2 fully integrated system with different working modules was developed. A MoveIt2 program was used to unite and make a sequential logic of all modules together. The solution had many subsystems and different working parts that functioned well together. The LLMs were used in combination with GroundingDINO and SAM to produce flexible object segmentation and allow for a close connection between operator and robotic system, which is a significant step in human-robot interaction. The system also had a reliable object pose estimation and had successful results in the Vision System Tests. The stack optimisation was also used for intelligent box placement in the pallet ensuring they do not fall from the pallet and minimise the height. This is also crucial palletizing operations as it result in intelligent box placement allowing more items to be stacked. The system as a whole is seen as a great body of work due to the time needed to develop each for the subsystems and also test and integrate them in a real robotic system. As far as the main contributions to the literature the following stand out as mentioned in the Introduction of the thesis:

- Use of **RGB-D vision** for environmental perception.
- Integration of **LLMs** for natural language command understanding and flexible object specification.
- Advanced and promptable segmentation models (e.g., **GroundingDINO and SAM)** for identifying diverse objects without prior training on specific instances.
- Robust **pose estimation** techniques
- An **optimisation** approach for **Intelligent stacking** of retrieved boxes.

In terms of practical implementation, there are significant milestones/contributions mentioned in the thesis.

- UR10 robot control through ROS2 in intelligent fashion
- VG10 vacuum gripper integration in ROS2 and successful grasping logic
- Intrinsic Camera Calibration
- Extrinsic Camera Calibration
- Integration of LLMs in ROS2 system
- Use of state-of-the-art models for object segmentation and detection with SAM and GroundingDINO in a ROS2 system
- Robust Pose Estimation for box detection allowing consistent pick and placing of boxes
- Stack Optimisation with Constrained Programming that allowed for objective function definitions and intelligent robot behaviour

6.3.1 Future Work

There are still further development to be done in to improve the body of work presented in the thesis as mentioned in the interpretation of the test results previously in the Discussion section. The points that stand out are:

- Improvement in Object Segmentation by testing extra camera on the wrist and changing placement of the static camera.

- Test of other Pose Estimation logics for different objects that share same segmentation mask
- Change of static camera to evaluate its performance against the current D455 RGB-D camera
- System Prompt Refinement for LLMs
- Use of external server for LLM and L-SAM as these require significant processing power

The first point of possibilities of future work is crucial for a robust system that can reliable segment and estimate the pose of the boxes. The long distance from camera to objects in the image can make it difficult to segment objects consistently, which is also an issue for the pose estimation, since if the segmentation mask is used for pose estimation then objects might be seen as one even if they are different objects. The second point is similar to the first one and is to make the system more robust when estimating the pose of the objects. The depth information might be sufficient in some cases to determine if objects are the same or not so fine-tuning on the present logic and exploring new ideas can be beneficial. As seen in the Implementation section of the thesis, the depth camera presented some undesired results and its change to another model could help in the depalletizing operation, in this case it could be another model of the same D455 RealSense or even one with other specifications. There is also the need to make small adjustments in the system prompt in order to achieve close to 100% success rate. More example prompts and a more detailed description of what the task is and the role of LLM could be sued to increase the accuracy of the responses. The last point is more of a desirable feature to have, since the operation can take time to process the LLMs response while having other modules running on the computer. An external server, in the sense of another computer, could be used to lower the time needed to compute responses and also use bigger versions the Qwen3. As the functionality would be the same, the accuracy in responses would increase while lowering the computation time.

6.4 Conclusion

This thesis presented a flexible and vision-guided robotic system for depalletizing and palletizing operations, with a strong emphasis on enhancing automation through intelligent perception and natural human-robot interaction. The proposed solution addresses a significant gap in current industrial systems and the literature by removing strict assumptions regarding known box dimensions, fixed placements, and extensive pre-training, offering instead a generalizable and adaptable framework.

Through the integration of RGB-D vision, large language models, and state-of-the-art segmentation/detection models (GroundingDINO and SAM), the system demonstrated the ability to retrieve object information from natural language commands, segment and estimate the pose of objects in a real-world testing environment, and intelligent box stacking with robustness.

The practical implementation was successfully achieved in ROS2 with real hardware, including a UR10 robot, VG10 vacuum gripper, and an Intel RealSense D455 RGB-D camera. System tests validated the approach across multiple scenarios, including challenging segmentation and ambiguous language prompts. The results highlight the feasibility and strength of combining robotics with transformer-based models for flexible and efficient logistics automation.

Despite the achievements, challenges remain, particularly in improving segmentation under complex visual conditions and refining object extraction in linguistically ambiguous scenarios. These findings outline clear directions for future work, including exploring alternative camera placements, evaluating different pose estimation strategies, and offloading computationally heavy modules like LLMs and L-SAM to external servers for faster response time.

In conclusion, the thesis makes a meaningful contribution to the field of flexible automation and intelligent robotic depalletizing. It lays a foundation for future systems that are not only robust but also intuitive to operate as a human. This is done to advance and achieve seamless collaboration between humans and robots in dynamic, real-world logistics environments.

Bibliography

[1] Seunghoon Baek, Seung Hyun Lee, and Seung Eel Oh. “Conceptual Design of Simulation-Based Approach for Robotic Automation Systems: A Case Study of Tray Transporting”. eng. In: *Processes* 12.12 (2024), pp. 2791–. ISSN: 2227-9717.

[2] Author. “An Artificial Intelligence enabled Smart Industrial Automation System based on Internet of Things Assistance”. In: *IEEE Conference on Industrial Automation*. 2022. URL: https://ieeexplore.ieee.org/document/9752651/.

[3] Author. “Case studies of robots and automation as health/safety interventions in small manufacturing enterprises”. In: *Human Factors and Ergonomics in Manufacturing* 32.4 (2022), pp. 287–301. URL: https://onlinelibrary.wiley.com/doi/10.1002/hfm.20971.

[4] Author. “Integration of robotics and automation in supply chain: a comprehensive review”. In: *Extrica Journal of Industrial Engineering* 5.2 (2024), pp. 89–104. URL: https://www.extrica.com/article/23349.

[5] Anyware Robotics. *Anyware Robotics Emerges from Stealth Mode to Reveal Its Pixmo Robots for Container and Truck Unloading*. Press release. Feb. 2024. URL: https://anyware-robotics.com/anyware-robotics-emerges-from-stealth-mode/ (visited on 06/01/2025).

[6] Sebastián Valero, Juan Camilo Martinez, Ana María Montes, Cesar Marín, Rubén Bolaños, and David Álvarez. “Machine Vision-Assisted Design of End Effector Pose in Robotic Mixed Depalletizing of Heterogeneous Cargo”. eng. In: *Sensors (Basel, Switzerland)* 25.4 (2025), pp. 1137–. ISSN: 1424-8220.

[7] RiACT. *Success Stories and Cases*. Danish software company specializing in robotic automation solutions. RiACT. URL: https://www.riact.ai/%5C#cases (visited on 05/30/2025).

[8] Hideichi Nakamoto, Haruna Eto, Takafumi Sonoura, Junya Tanaka, and Akihito Ogawa. “High-speed and compact depalletizing robot capable of handling packages stacked complicatedly”. eng ; jpn. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2016-. IEEE, 2016, pp. 344–349. ISBN: 9781509037629.

[9] D.K. Katsoulas and D.I. Kosmopoulos. “An efficient depalletizing system based on 2D range imagery”. eng. In: *Proceedings - IEEE International Conference on Robotics and Automation*. Vol. 1. IEEE, 2001, 305–312 vol.1. ISBN: 0780365763.

[10] D. Katsoulas, L. Bergen, and L. Tassakos. “A versatile depalletizer of boxes based on range imagery”. In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*. Vol. 4. 2002, 4313–4319 vol.4. DOI: 10.1109/ROBOT.2002.1014438.

[11] Haruna ETO, Hideichi NAKAMOTO, Takafumi SONOURA, Junya TANAKA, and Akihito OGAWA. “Development of automated high-speed depalletizing system for complex stacking on roll box pallets”. eng. In: *Journal of Advanced Mechanical Design, Systems, and Manufacturing* 13.3 (2019), JAMDSM0047–JAMDSM0047. ISSN: 1881-3054.

[12] Riccardo Caccavale, Pierluigi Arpenti, Gianmarco Paduano, Andrea Fontanellli, Vincenzo Lippiello, Luigi Villani, and Bruno Siciliano. “A Flexible Robotic Depalletizing System for Supermarket Logistics”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4471–4476. DOI: 10.1109/LRA.2020.3000427.

[13] Qian Li, Shuzhou Dong, Dong Zhang, and Xuelin Wang. “Research on the Lidar-based Recognition and Location Method for Depalletizing Targets”. In: *2020 Chinese Automation Congress (CAC)*. 2020, pp. 683–687. DOI: 10.1109/CAC51589.2020.9327723.

[14] Domenico Buongiorno, Donato Caramia, Luca Di Ruscio, Nicola Longo, Simone Panicucci, Giovanni Di Stefano, Vitoantonio Bevilacqua, and Antonio Brunetti. “Object Detection for Industrial Applications: Training Strategies for AI-Based Depalletizer”. eng. In: *Applied sciences* 12.22 (2022), pp. 11581–. ISSN: 2076-3417.

[15] Pierluigi Arpenti, Riccardo Caccavale, Gianmarco Paduano, Giuseppe Andrea Fontanelli, Vincenzo Lippiello, Luigi Villani, and Bruno Siciliano. “RGB-D Recognition and Localization of Cases for Robotic Depalletizing in Supermarkets”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 6233–6238. DOI: 10.1109/LRA.2020.3013936.

[16] Federico Zaccaria, Alberto Baldassarri, Gianluca Palli, and Marco Carricato. “A Mobile Robotized System for Depalletizing Applications: Design and Experimentation”. eng. In: *IEEE access* 9 (2021), pp. 96682–96691. ISSN: 2169-3536.

[17] Riccardo Monica, Jacopo Aleotti, and Dario Lodi Rizzini. “Detection of Parcel Boxes for Pallet Unloading Using a 3D Time-of-Flight Industrial Sensor”. In: *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*. 2020, pp. 314–318. DOI: 10.1109/IRC.2020.00057.

[18] Santheep Yesudasu. “Enhancing Logistics Automation with AI : Application of Dual-arm Humanoid Torso for AI-powered Depalletizingand Package Handling.” Theses. Normandie Université, Oct. 2024.
URL: https://theses.hal.science/tel-04874770.

[19] Seongje Kim, Van-Doi Truong, Kwang-Hee Lee, and Jonghun Yoon. “Revolutionizing Robotic Depalletizing: AI-Enhanced Parcel Detecting with Adaptive 3D Machine Vision and RGB-D Imaging for Automated Unloading”. eng. In: *Sensors (Basel, Switzerland)* 24.5 (2024), pp. 1473–. ISSN: 1424-8220.

[20] Jonghun Yoon, Jooyeop Han, and Thong Phi Nguyen. “Logistics box recognition in robotic industrial de-palletising procedure with systematic RGB-D image processing supported by multiple deep learning methods”. In: *Engineering Applications of Artificial Intelligence* 123 (2023), p. 106311. ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106311. URL: https://www.sciencedirect.com/science/article/pii/S0952197623004955.

[21] Jinquan Li, Jia Kang, Zhe Chen, Fang Cui, and Zixuan Fan. “A workpiece localization method for robotic de-palletizing based on region growing and PPHT”. In: *IEEE Access* 8 (2020), pp. 166365–166376.

[22] Pierluigi Arpenti, Riccardo Caccavale, Andrea Giuseppe Fontanelli, Vincenzo Lippiello, Gianmarco Paduano, Bruno Siciliano, Luigi Villani, Luigi Villani, Michael Beetz, Bruno Siciliano, and Ciro Natale. “Robots Working in the Backroom: Depalletization of Mixed-Case Pallets”. eng. In: *Robotics for Intralogistics in Supermarkets and Retail Stores*. Vol. 148. Springer Tracts in Advanced Robotics. Switzerland: Springer International Publishing AG, 2022, pp. 81–115. ISBN: 3031060776.

[23] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. “Object Detection With Deep Learning: A Review”. In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), pp. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.

[24] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[25] David G. Lowe. “Distinctive image features from scale-invariant keypoints”. eng. In: *International journal of computer vision* 60.2 (2004), pp. 91–110. ISSN: 0920-5691.

[26] Rainer Lienhart and Jochen Maydt. “An extended set of haar-like features for rapid object detection”. In: *Proceedings. international conference on image processing*. Vol. 1. IEEE. 2002, pp. I–I.

[27] Yoav Freund and Robert E Schapire. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting”. In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139. ISSN: 0022-0000. DOI: https://doi.org/10.1006/jcss.1997.1504.
URL: https://www.sciencedirect.com/science/article/pii/S002200009791504X.

[28] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. “Deep Learning for Generic Object Detection: A Survey”. eng. In: *International journal of computer vision* 128.2 (2020), pp. 261–318. ISSN: 0920-5691.

[29] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. eng. In: *IEEE Conference on Computer Vision and Pattern Recognition. Proceedings*. 2014, pp. 580–587.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *Computer Vision – ECCV 2014*. Springer International Publishing, 2014, pp. 346–361. ISBN: 9783319105789. DOI: 10.1007/978-3-319-10578-9_23.
URL: http://dx.doi.org/10.1007/978-3-319-10578-9_23.

[31] Ross Girshick. “Fast R-CNN”. eng. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Vol. 2015. IEEE, 2015, pp. 1440–1448. ISBN: 1467383910.

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. eng. In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2017), pp. 1137–1149. ISSN: 0162-8828.

[33] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. “Mask R-CNN”. eng. In: *IEEE transactions on pattern analysis and machine intelligence* 42.2 (2020), pp. 386–397. ISSN: 0162-8828.

[34] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. “Deep Neural Networks for Object Detection”. In: *Advances in Neural Information Processing Systems*. Ed. by C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger. Vol. 26. Curran Associates, Inc., 2013.
URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/f7cade80b7cc92b991cf4d2806d6bd78-Paper.pdf.

[35] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks”. eng. In: (2013).

[36] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. eng. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2016-. IEEE, 2016, pp. 779–788. ISBN: 9781467388504.

[37] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, Bastian Leibe, Max Welling, Nicu Sebe, and Jiri Matas. “SSD: Single Shot MultiBox Detector”. eng. In: *Computer Vision - 14th European Conference, ECCV 2016, Proceedings*. Vol. 9905. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 21–37. ISBN: 3319464477.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need". eng. In: (2017).

[39] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. "Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection". In: *arXiv preprint arXiv:2303.05499* (2023).

[40] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. "Segment Anything". In: *arXiv preprint arXiv:2304.02643* (2023).

[41] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. "YOLO-World: Real-Time Open-Vocabulary Object Detection". In: *arXiv preprint arXiv:2401.17270* (2024).

[42] Rafael Padilla, Wesley L. Passos, Thadeu L.B. Dias, Sergio L. Netto, and Eduardo A.B. Da Silva. "A comparative analysis of object detection metrics with a companion open-source toolkit". eng. In: *Electronics (Basel)* 10.3 (2021), pp. 1–28. ISSN: 2079-9292.

[43] Jian Guan, Yingming Hao, Qingxiao Wu, Sicong Li, and Yingjian Fang. "A Survey of 6DoF Object Pose Estimation Methods for Different Application Scenarios". In: *Sensors* 24.4 (2024). ISSN: 1424-8220. DOI: 10.3390/s24041076.
URL: https://www.mdpi.com/1424-8220/24/4/1076.

[44] Alex Kendall, Matthew Grimes, and Roberto Cipolla. "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization". eng. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2015, pp. 2938–2946. ISBN: 1467383910.

[45] Mazin Hamad, Simone Nertinger, Robin Jeanne Kirschner, Luis F. C. Figueredo, Abdeldjallil Naceri, and Sami Haddadin. "A Concise Overview of Safety Aspects in Human-Robot Interaction". In: *International Workshop on Human Friendly Robotics*. 2023.
URL: https://api.semanticscholar.org/CorpusID:262046326.

[46] Ferran Gebellí, Lavinia Hriscu, Raquel Ros, Séverin Lemaignan, Alberto Sanfeliu, and Anaís Garrell. "Personalised Explainable Robots Using LLMs". In: *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2025, pp. 1304–1308. DOI: 10.1109/HRI61500.2025.10974125.

[47] Universal Robots A/S. *UR10 Technical specifications*. Tech. rep. 110110. Copyright © 2009-2016 by Universal Robots A/S. Universal Robots A/S, 2016.
URL: https://www.universal-robots.com/media/50895/ur10_en.pdf.

[48] OnRobot A/S. *VG10 Vacuum Gripper Datasheet*. Tech. rep. Version 1.1.0. Copyright © 2018 by OnRobot A/S. OnRobot A/S, 2018.
URL: https://onrobot.com/sites/default/files/documents/Datasheet_VG10_v2.0_ENGLISH.pdf.

[49] Intel Corporation. *Intel® RealSense™ Camera D455*. May 2025.
URL: https://www.intelrealsense.com/depth-camera-d455/.

[50] Intel Corporation. *Intel® RealSense™ Camera D400 Series Product Family Datasheet*. Tech. rep. 337029-009. Intel Corporation, June 2020.
URL: https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf.

[51] Qwen Team. *Qwen3*. Accessed: 2025-06-02. 2025.
URL: https://qwenlm.github.io/blog/qwen3/.

[52] Zivid. *Hand-Eye Calibration Solution*. Accessed: 2025-05-20. 2024.
URL: https://support.zivid.com/en/latest/academy/applications/hand-eye/hand-eye-calibration-solution.html.

[53] Radu Horaud and Fadi Dornaika. “Hand-Eye Calibration”. In: *arXiv preprint arXiv:2311.12655* (2023). Accessed: 2025-05-21.
URL: https://arxiv.org/pdf/2311.12655.

A Links

Github

Github link: **https://github.com/hjorthchristian/ROB10**

YouTube Videos

Videos of the solution will be uploaded on the author's YouTube channel, as of time of submission a full system test demonstration can be seen in the author's YouTube channel.

YouTube Channel: **https://www.youtube.com/@christianhjorth6047**

Full System Demo video: **https://youtu.be/TlOMOg49Kk8**

B Appendix - Test Data

LLMs Object Extraction Test Prompts

#	Prompt	Expected Output
1	can you get me the packages please?	package
2	Can you pick up the boxes?	box
3	Could you place the boxes on the pallet?	box
4	Please put the package on the pallet.	package
5	Can you get me all those boxes please and put them on the pallet.	box
6	Grab me those boxes please?	box
7	Can you take the packages from the floor?	package
8	Pick up the box for me, please.	box
9	Please get those boxes.	box
10	Can you bring the package here?	package

Table B.1: Prompts and Expected Outputs – Basic Object Identification

#	Prompt	Expected Output
1	can you get me the red box?	red box
2	Could you retrieve the white packages?	white package
3	Can pick me the small package?	small package
4	Place the red package on the pallet please	red package
5	can you get me the big boxes?	big box
6	I want you to get me all of the big packages for me	big package
7	i really love red packages. Can you grab me the blue package?	blue package
8	i was thinking about you picking the red package, can you get me the white box?	white box
9	Please grab the tall box.	tall box
10	Can you pick up the yellow package?	yellow package

Table B.2: Prompts and Expected Outputs – Object + Attribute Extraction

#	**Prompt**	**Expected Output**
1	Can you take the big box and the small box and place them on the pallet?	big box, small box
2	Get the blue package and the red box.	blue package, red box
3	Please grab the large box and the small one.	large box, small box
4	Can you get the green package and the tall box?	green package, tall box
5	I need the heavy box and the light package.	heavy box, light package
6	Pick the short box and the long one.	short box, long box
7	Bring me the black package and the white one.	black package, white package
8	Can you move both the fragile package and the red one?	fragile package, red package
9	Get me the small green box and the medium blue one.	small green box, medium blue box
10	Can you pick up the wide box and the narrow package?	wide box, narrow package

Table B.3: Prompts and Expected Outputs – Multiple Object References

#	**Prompt**	**Expected Output**
1	can you get me the white box and the green one but not actually the green one just the white one.	white box
2	Bring the white package, but not the black one.	white package
3	Get the red box and ignore the green one.	red box
4	Pick up the blue box, not the damaged one.	blue box
5	I want the big package, but not the tall box.	big package
6	Please retrieve the clean package, but skip the dirty one.	clean package
7	Bring the boxes — but leave the crushed ones.	box
8	Take the yellow box, not the orange one next to it.	yellow box
9	Grab the green and white boxes, but not the green one.	white box
10	Collect the large packages, but not the broken ones.	large package

Table B.4: Prompts and Expected Outputs – Negation and Exclusion

#	**Prompt**	**Expected Output**
1	The red box is not needed right now. Please retrieve the blue box.	blue box
2	Ignore the black package for now. When you’re ready, bring the green package.	green package
3	The red package was evaluated earlier. I need the blue package instead.	blue package
4	I initially thought about using the red package. Can you get the white box instead?	white box
5	The green boxes are set aside for another task. Proceed with the white package.	white package
6	The small box is damaged. Please bring the large one.	large box
7	We’re not using the red package for this. Please bring the yellow one.	yellow package
8	Skip the short box. Please retrieve the tall box for the next step.	tall box
9	The fragile package has been excluded. Bring the black one instead.	black package
10	We’ve already used the blue package. Now pick the orange box.	orange box

Table B.5: Prompts and Expected Outputs – Noise / Extra Clauses (with Distractors)