Vision Based Bin-Picking of Inserts for Injection Moulding

Master degree in Mechanical Engineering – Industrial Production

Inês Miguel da Silva Empadinhas

Leiria, April of 2026

Vision Based Bin-Picking of Inserts for Injection Moulding

Master degree in Mechanical Engineering – Industrial Production

Inês Miguel da Silva Empadinhas

Project report under the supervision of Professor Carlos Fernando Couceiro de Sousa Neves, and Professor Paulo Jorge Simões Coelho.

Leiria, April of 2026

Originality and Copyright

This project report is original, made only for this purpose, and all authors whose studies and publications were used to complete it are duly acknowledged.

Acknowledgments

First of all, I would like to thank Professor Carlos Neves and Professor Paulo Coelho for their invaluable support and guidance throughout the development of this project. I also want to thank Professor Hugo Costelha for all technical support provided.

I want to thank all my friends and colleagues, in particular Edmar, Leonor and Rafaela, for their continuous encouragement.

I would like to express my gratitude to my family, that have always been my biggest supporters, with special thanks to my parents that have always taught me that we create our own luck.

Lastly, I would like to thank my partner Gonçalo, I would have not been able to conclude this work without his support, patience and for his belief in me throughout every stage of this process.

This work was partially supported by Project nº60 (C645808870-00000067), Produtech R3, co-funded by Recovery and Resilience Plan and NextGeneration EU Funds, recuperarportugal.gov.pt.

Abstract

This project involves identifying computer vision technologies and hardware for the development of a bin-picking system. The developed prototype identifies parts made from a dark, stiff fabric material within a bin and places them in an organized way. The placement of the parts has to be precise and allow a cartesian robot tending an injection machine to pick them in the correct orientation for placement inside the mould.

In the initial stage of this project, research was conducted to identify computer vision technologies and hardware that better adapt to unknown cluttered environments. According to the conclusions taken from the previous stage, a bin-picking prototype was developed, with the aid of a 6 degree of freedom robotic manipulator, an RGBD sensor, classic computer vision technologies, as well as deep learning models.

Initial tests were run to determine areas for optimization of the prototype, and improvements were implemented. The prototype's performance was assessed through a single-fill test and trials with varying part densities in the bin. Multiple metrics were analysed under different conditions to assess the prototype's performance, such as "Average cycle time," "Number of attempts until successful pick" and "Rejection analysis".

Overall, the prototype achieved the desired performance, meeting the required cycle time. The results confirm the feasibility of the developed system; however, further refinements are proposed for the development of the final industrial system under its intended operating conditions.

Keywords: Bin-picking, Robotic manipulation, Vision-guided robotics, Object detection, RGBD sensing

Resumo

Este projeto consiste na identificação de metodologias de visão computacional e hardware para o desenvolvimento de um sistema de *bin-picking*. O protótipo desenvolvido identifica insertos, produzidos num tecido preto e rígido, dentro de uma caixa e posiciona-os de uma forma organizada. O posicionamento dos insertos tem de ser preciso e permitir que um robô cartesiano integrado numa máquina de injeção os apanhe corretamente para posicionamento dentro do molde de injeção.

Numa fase inicial deste projeto, foi realizado um estudo do estado da arte para identificar as metodologias de visão computacional e de hardware que melhor se adaptam a situações em que objetos se encontram de forma desorganizada e com oclusões. Com base nas conclusões deste estudo, um protótipo de um sistema de *bin-picking* foi desenvolvido incluindo um robô com 6 graus de liberdade, um sensor RGBD, métodos clássicos de visão computacional e, também metodologias de aprendizagem automática.

Os primeiros testes experimentais tiveram como objetivo determinar oportunidades de melhoria ao protótipo, e estas melhorias foram implementadas. O desempenho do sistema foi avaliado com base num teste de esvaziamento da caixa e ensaios com diferentes quantidades de peças dentro da caixa. Nesta avaliação foram utilizadas métricas como "Tempo de ciclo médio", "Número máximo de tentativas até apanhar uma peça corretamente" e "Análise de rejeições".

O protótipo atingiu o desempenho desejado, cumprindo o tempo de ciclo. Os resultados obtidos confirmam a viabilidade do sistema; ainda assim, são propostas melhorias para o desenvolvimento do sistema industrial nas suas condições de funcionamento finais.

Palavras-Chave: *Bin-picking*, Manipulação robótica, Robótica guiada por visão, Deteção de objetos, Sensores RGBD

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

AVN	Angle-View Net
CAD	Computer-Aided Design
CCP	Computed Closest Point
CNN	Convolutional Neural Network
DOF	Degrees Of Freedom
DSSD	Deconvolutional Single Shot Detector
FAS	Fast Analytical Searching
FCN	Fully Convolutional Network
GCN	Graph Convolutional Network
GPIO	General Purpose Input/Output
GPM	Geometric Pattern Matching
GSNet	Graspness-based Sampling Network
ICP	Iterative Closest Point
IoU	Intersection Over Union
LiDAR	Light Detection and Ranging
mAP	Mean Average Precision
MT-DSSD	Multi-Shot Deconvolutional Single Shot Detector
PLA	Polylactide
PPF	Point Pair Feature
RANSAC	Random Sample Consensus
RANSAM	Random Sample Matching
RCNN	Region-based Convolutional Neural Networks
SPPF	Semantic Point Pair Feature
TCP	Tool Center Point
TCP/IP	Transmission Control Protocol/Internet Protocol
YOLO	You Only Look Once

1. Introduction

As technology evolves, the number of automated tasks has also increased. The overall goal is to prevent workers from repetitive, monotonous and dangerous tasks that may lead to several health problems. Moreover, the increase in quality and efficiency, and the reduction in costs are very appealing factors to increase investment in these areas (Xu et al., 2024; Martinez et al., 2015).

Industrial automation applications often demand parts to be fed in a precise and organized way into the automatic system. However, these parts do not always arrive in an organized manner: it is very common that they are delivered in bulk, in boxes, bins or bags, and a worker will have to separate and orient the parts to feed them to the automated machine. The automation of these sorting tasks usually requires a considerable amount of effort in their implementation and are commonly very specific for a target situation and part (Martinez et al., 2015; Pochyly et al., 2012).

One of the ways to overcome these challenges is the use of bin picking systems consisting of the pick and placing of objects that are, initially, in a randomly arranged and cluttered scene, inside a bin. These systems perform three tasks: object detection, grasp planning and placement of the object. The advantages associated with this technology are increased efficiency, increased safety and decreased ergonomic problems for workers as well as a better adaptability to different parts and setups when compared to other automated systems (Xu et al., 2024; Pochyly et al., 2012).

1.1. Problem Statement

This work focuses on the pick and place task of a stiff fabric pad, from a bin into a mould cavity, for the molten plastic to be moulded around it, integrating the pad into a final part, as seen in Figure 1, in a process known as overmoulding. The pads must be picked from a bin, oriented and precisely placed in the mould. This process is very time consuming, repetitive and requires a worker to be in close contact with a mould at high temperatures for extended periods of time.

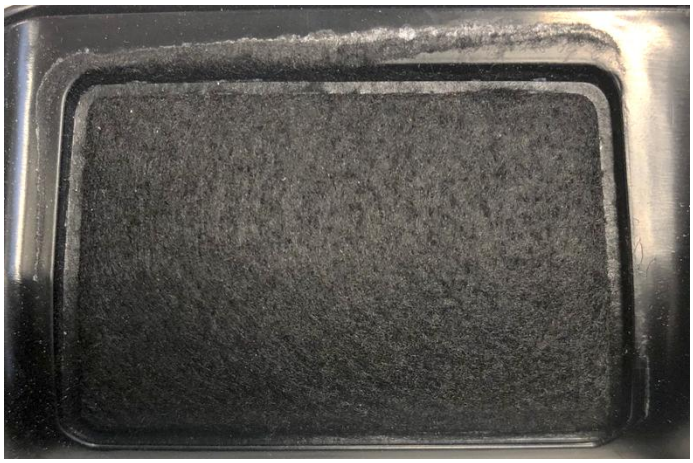

Figure 1: Example of an overmoulded part.

The aim of this project is to develop an automated system that can analyse the environment around it and plan the object pick and place based on the interpretation of the data collected, whilst avoiding collisions with the bin walls and other parts in the scene. The developed prototype is to be installed next to an injection machine equipped with a cartesian robot. A simplified representation is presented in Figure 2.

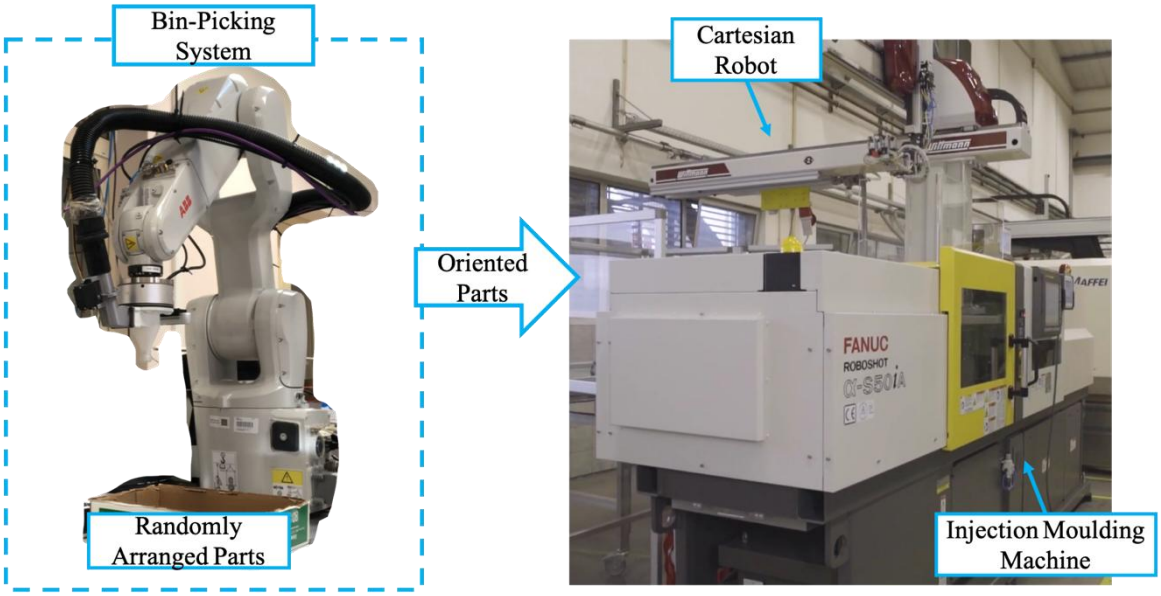

Figure 2: Representation of the complete system, integrating the bin picking system with the injection machine.

In this report, a comprehensive analysis of related work is presented from which key conclusions are taken to design and implement a complete solution capable of handling the stiff fabric pads from the case study at hand.

1.2. Document Structure

This report is divided into two sections. The first section consists of the review of the state of the art on bin picking systems (Chapter 2) and computer vision techniques (Chapter 3).

The second section of this report describes the setup of the developed system (Chapter 4), the tests done along with the results obtained (Chapter 5) and conclusions and improvement opportunities (Chapter 6).

2. State of The Art

This section reviews the literature regarding bin picking applications. The search process for this review is conducted using the search engines: IEEE Xplore, Science Direct and Google Scholar, as well as other available tools, such as ConnectedPapers and ResearchRabbit. The following keywords are used: bin-picking, computer vision and object detection.

The aim of this research is to determine which computer vision methodologies provide good results in heavily cluttered environments with previously known parts, and to determine the hardware configurations that better adapt to the conditions presented, allowing for precise positioning of the parts in their final placement.

2.1. Available Technologies

2.1.1. Machine Vision Methods

Currently, for computer vision methodologies, it is possible to identify two distinct paths: classical computer vision methods and deep learning approaches, the latter being the focus of the most recent research. It is also important to classify the applications analysed regarding the use of prior information about the objects to grasp, therefore the applications are further categorized as model-based or model-free (Cordeiro et al., 2022; Kleeberger et al., 2020).

Traditional computer vision techniques have existed for many decades; these normally focus on identifying distinctive features of the object to detect on a scene. Traditional techniques are implemented through coding and depend on the computer vision engineer's expertise to decide what features are distinctive enough to detect the parts, and to tune the system. Taking into consideration their low complexity, these methods can be used in low-cost microcontrollers, providing computer vision solutions at a lower cost (Mahony et al., 2020).

Deep learning approaches are more recent and a subgroup of machine learning. These methods are based on Artificial Neural Networks which are composed of cells that perform simple operations and together make decisions about the inputted data. These approaches typically require considerable amounts of data to train the models and demand a substantial computational power. Moreover, deep learning depends greatly on the data that is provided

for learning and on the annotations done on this data, which is a time consuming process; in addition, the datasets have to be well constructed, so that the network works as intended for the desired application (Mahony et al., 2020).

Nevertheless, both methodologies can be combined to merge the advantages of both, some examples are: using traditional techniques for augmenting the data used to train the networks, when not enough data is available; to highlight important features of the objects to identify; or to support dataset annotation (Mahony et al., 2020).

Both classical and deep learning approaches can be further subdivided into model-based and model-free. The model-based approaches use prior information about the part to be picked to plan the grasp and, typically, are more object-specific and don't perform as well with unknown objects. The model-free approaches do not rely on prior information about the part to perform the grasp, not allowing for precise placement of the object since its pose on the gripper is unknown (Cordeiro et al., 2022; Kleeberger et al., 2020).

2.1.2. Hardware

A crucial element of a bin-picking system is the acquisition of information about the environment that it is interacting with. Nowadays, a wide variety of sensors are readily available across different price ranges, some examples are RGBD cameras, 2D (RGB and grey level) cameras and depth, tactile and force sensors, all of which provide useful information to the system.

RGBD cameras can provide a full picture of the scene that is being analysed, being able to provide depth and colour information without the need to match data from different sensors; this information can be treated as one, in the form of point clouds, or separately (one RGB image and one depth image). 2D cameras can be used in combination with depth sensors to achieve a similar result, identifying the parts on the 2D image and using the depth information to plan the trajectories of the robot and avoid collisions. 2D cameras can also be used to detect colours to segregate parts by colour or in pairs, as stereo vision, to determine depth information (*RealSense*, 2025; *Industrial 3D Cameras for Robotic Automation - Zivid*, 2025; *One Source. Unlimited Vision. | Teledyne Vision Solutions*, 2025).

The above-mentioned sensors can be installed in different positions, providing different information to the system. They may be in a fixed position, mounted on a frame,

continuously acquiring information from the same point of view, and freeing the robot to perform a different task; or they may be mounted on the wrist of the robot, allowing for more flexibility in the field of view, contributing to the handling of occlusions or blind spots.

Tactile, force and torque sensors can be used to complement the information provided by the cameras and depth sensors, allowing the system to detect collisions, to detect if a grasp is successful or not, and to detect if more than one object is grasped (*Force/Torque Sensors*, 2025; *Force Torque Sensor - Robot Force Sensors | Precise FT Sensor Control*, 2025; *Tactile Sensors*, 2025).

2.2. Related Literature Review

Within the frame of the project, review articles are analysed to understand what the available technologies for bin picking applications are, including the vision methodologies, available hardware, trajectory planning and collision avoidance. From this search, it is observed that recent articles have been mostly focused on deep learning approaches, and a small number of reviews were found for complete bin picking applications.

Mahony et al. (2020) provide a comprehensive overview of traditional computer vision techniques and deep learning approaches, presenting strengths and weaknesses associated with both methods. Additionally, the advantages of combining both methods are highlighted. Some applications for hybrid methods are presented, such as 3D vision, panoramic image stitching, augmented reality, and improving the creation of datasets, by reducing the time needed to annotate them or by augmenting the data when reduced amounts of data are available. One of the key takeaways from this paper is that certain applications will benefit from deep learning approaches, but for others they are simply an overly complex solution. Lastly, some cases are presented where traditional computer vision techniques remain relevant and provide better results than deep learning, such as 3D modelling, video stabilization and motion capture.

Cordeiro et al. (2022) divide the bin-picking procedure in two types of approaches: gripper-oriented and object-oriented, as shown in Figure 3, focusing mainly on deep learning approaches. Gripper-oriented methods don’t aim to identify the different objects in the bin, focusing instead on detecting grasp opportunities. These methods are associated with lower capability to handle occlusions. Object-oriented methods rely on a notion of pose,

identifying first where the object is, and then computing how it will be grasped. These methods are subdivided into model-based/analytical approaches and model-free/data-driven. The first usually require tuning and are very specific to a certain object. The latter identify the surfaces of the object that are viable for grasping. Finally, the authors review a number of relevant articles and conclude on a few key points for the development of the bin picking applications, such as the focus on the development of the processing units and less on the mechanical structure of the systems.

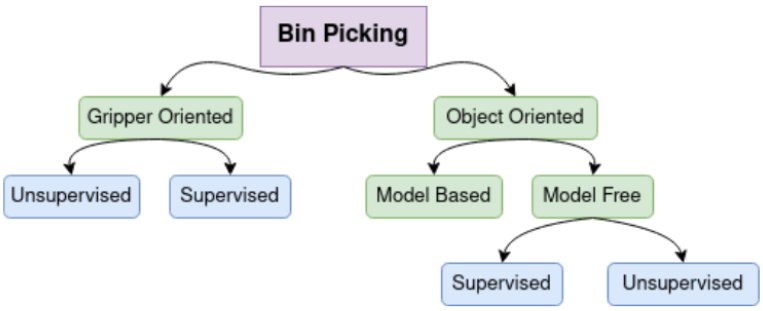

Figure 3: Subdivision of the Bin Picking Technologies according to Cordeiro et al. (2022).

He et al. (2021) provide an extensive overview of 6D pose estimation methods, categorizing them into learning-based approaches, subdivided into keypoint-based, holistic and RGBD-based approaches; and non-learning-based approaches, subdivided into 2D-information-based and 3D-information-based. The authors also provide a comparison between these categories in accuracy, robustness, storage cost, time cost and real time performance, and range of application, as presented in Table 1. Finally, the challenges faced in the development of bin-picking systems are listed, such as: texture-less and reflective objects, foreground occlusion, background clutter and deformable objects.

Table 1: Comparison of the approaches presented in He et al. (2021). (A representing the best performance and C the worst performance).

		Accuracy	Storage Cost	Robustness	Time Cost	Online Performance	Range of Application
Learning-based approaches	Keypoint-based approaches	B	B	B	C	C	B
	Holistic approaches	C	B	B	B	B	B
	RGBD-based approaches	A	C	A	C	C	C
Non-learning-based approaches	2D-information-based approaches	B	A	C	A	A	A
	3D-information-based approaches	A	B	B	B	B	C

Kleeberger et al. (2020) provide a review on robot grasping methodologies, mostly based on learning approaches. The methodologies are categorized into model-based, requiring object specific knowledge; and model-free, being able to generalize its own knowledge to unknown objects. Several articles are analysed on a few different key factors, such as sensor data, if the objects are known or unknown, gripper type, robot type, precise placement of the object and reported grasp success rate. The authors recognize that all model-free approaches analysed executed top-down grasps and presented limited flexibility, not allowing for precise placement of the parts. They also conclude that machine learning and simulation allow for fast and easy implementations of robust bin picking applications for cluttered and non-static scenes.

In Fujita et al. (2019), a comprehensive review of the critical factors to designing a successful and robust bin-picking system are highlighted, such as gripper design, grasp planning and grasp point detection for object grasp, as well as sensors and algorithms for object recognition. Subsequently, the top four ranked systems from the 2017 Amazon Robotics Challenge are presented These systems were designed to handle multi-reference bins, identifying each object, grasping and placing it. Also, the systems are compared using metrics proposed by the authors such as: average time per pick, number of trials per hour, mean time between failures, mean time to repair, availability and mean picks per hour. Different strategies for handling grasp failures allow the robot to quickly recover, improving

performance. Finally, the similarities and differences between the presented bin-picking systems are emphasized and the authors take conclusions about the key technologies to develop robust bin picking systems such as: recovery strategies, re-usability of the system for different parts and autonomous improvement of speed and quality.

2.3. Bin Picking Applications

2.3.1. Classical Methods

2.3.1.1. Model-Based

In Oh et al. (2012), the authors propose a bin picking system based on stereo vision. The proposed setup consists of two analogue SONY cameras, one with a fixed position and one mounted on the robot; a Hyundai HA020 robot arm and a two-finger gripper as the end-effector, as seen in Figure 4. The object detection task relies on Geometric Pattern Matching (GPM) which is applied to the 2D image obtained with the camera on the robot; this method relies on a previously generated database with the specific geometric features of the part. To improve cycle time of the operation, the search of the object is limited to certain regions of interest. The 3D pose of the object is obtained by triangulation, after the determination of the correspondence points between the two images. Additionally, on the parts that present circular features an ellipse fitting technique is applied to perform a more accurate determination of the correspondence points in the stereo image. Lastly, the trajectory is computed, the approximation of the end effector to the detected part being done in the normal direction of the object. However, if the part is near the bin walls, the robot approaches the part vertically. If the system detects an object but cannot determine its pose it will move to a different image acquisition position and recalculate the 3D pose. The system is tested for three different industrial metallic parts in mono-reference bins with an error under 1 mm and 0.3º.

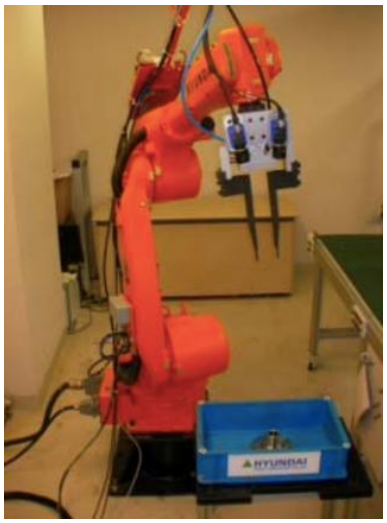

Figure 4: Configuration of the system proposed in Oh et al. (2012), consisting of a robotic arm and a two-finger gripper.

In Buchholz et al. (2013), a bin-picking system is proposed based on an improved Random Sample Consensus (RANSAC) algorithm, denominated Random Sample Matching (RANSAM). The setup consists of a Staubli RX-60 industrial manipulator equipped with a parallel-jaw gripper, and a laser line scanner mounted on a linear axis, as represented in Figure 5. First, the Sobel operator is applied over the image, then the RANSAM algorithm is applied, outputting several valid pose hypotheses which go through pose refinement using the Iterative Closest Point (ICP) algorithm. The grasp position is computed with Key Grasp Frame. This process begins with the definition of a starting pose for the end-effector, followed by the definition of a set of degrees of freedom, each associated with an interval of allowable variation. Finally, the position of the end-effector is simulated in the point cloud to detect collisions. The system is tested with two different metallic objects in mono-reference bins and the average time for the localization of the object and the generation of the pick pose from the moment the scan data is available is 5.5 seconds.

Figure 5: Experimental setup used in Buchholz et al. (2013), consisting of a robotic arm and a laser line scanner.

In Pretto et al. (2013), the authors propose a bin-picking system for planar objects using only one 5 MP grey-level CMOS camera and a Fanuc M-719 robotic manipulator. The proposed setup is developed to handle 7 different planar parts displayed in mono-reference bins. The software is implemented in C++ using the OpenCV library for image processing and is tested in two situations: with the camera mounted on a fixed frame and mounted on the robot, as shown in Figure 6. Object detection is performed using the Line Segment Detector with few alterations to reduce the number of false detections due to image noise, and then a voting Hough-like scheme is applied to get the correct hypothesis for the object pose. The parts inside the bin are picked with a suction cup gripper and the distance of the part to the camera is calculated knowing the placement of the camera and the CAD (Computer-Aided Design) model of the part. The proposed system achieves over 94% of first match successes with a maximum error of 1 mm and 0.5º, for laboratory tests, and over 87% first match successes for real industrial applications.

Figure 6: Experimental setups from Pretto et al. (2013). On the left, camera mounted on the robot and on the right, stationary camera.

In Wu et al. (2015), a picking system for a multi-reference cluttered environment is proposed. First, a database of depth images of three different parts is generated with the assistance of a virtual camera and the CAD models of the objects. Afterwards, the images are converted into point clouds, and a voxel grid filter is applied, reducing the amount of data in the point cloud representation and, consequently, reducing computation time. Next, the surface normals are computed, and Point Pair Feature (PPF) descriptors are stored in a hash table. In the online phase, a Kinect sensor acquires depth data that goes through the same procedure mentioned earlier and the surface normal is computed and matched against the database using a voting-scheme. The matches with low scores are discarded and the accuracy of the pose estimation is increased by applying the ICP algorithm. Finally, to select the object to pick, the centroids of all detected parts are computed and ranked by their z-coordinates, the higher value is chosen, since it represents the object at the topmost position in the pile. The test setup, represented in Figure 7, consists of a two-finger gripper and a Kinect sensor attached to a KUKA 6 degrees-of-freedom (DOF) robotic manipulator, all three parts are displayed on a pile and the system is able to identify, categorize and separate the objects into the respective bins, achieving an 89.7% success rate.

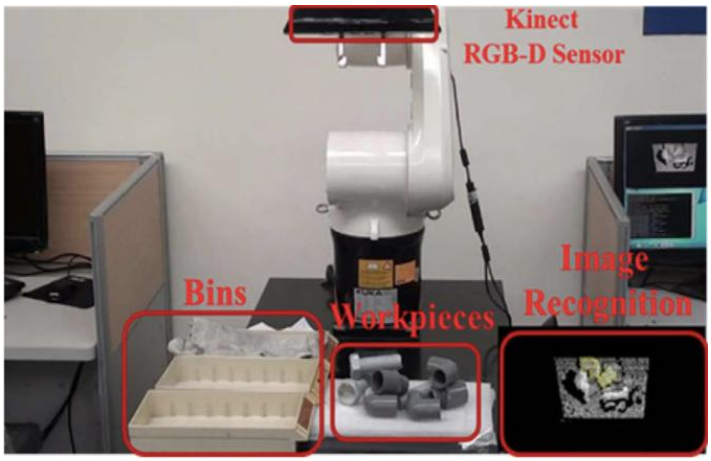

Figure 7: Test setup for Wu et al. (2015), consisting on a two-finger gripper, a Kinect sensor and a KUKA robotic arm.

In Chang & Wu (2016), a bin-picking system based on the combination of a camera and laser projection is presented, where both sensing elements are mounted on the wrist of a Mitsubishi 6DOF robot in addition to a two-finger gripper, as seen in Figure 8. The scene point cloud is computed by knowing the geometric relation between the camera and the laser projector. Next, the topmost point is selected, and 3D pose estimation is performed based on a previously known CAD model of the part. First, Computed Closest Point (CCP) is applied, outputting a rough pose estimation, and then ICP is applied to refine the pose estimation. According to the authors, CCP provides better results than ICP when the deviation between two point clouds is too large and reduces the computation time for the pose estimation task.

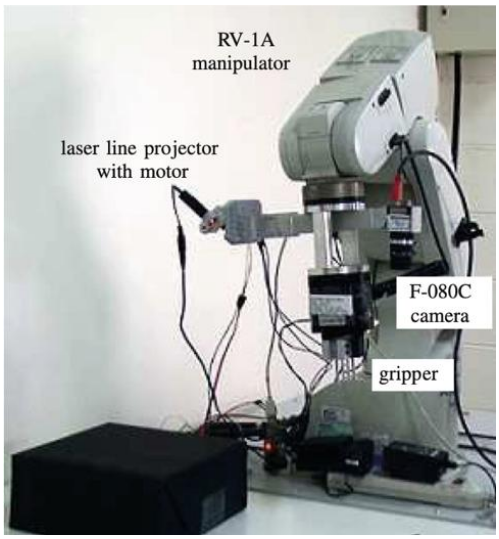

Figure 8: System developed in Chang & Wu (2016), consisting of a robotic manipulator, a 2D camera and a laser.

In Yan et al. (2020), the bin-picking system proposed consists of a 6DOF Kinova robot, equipped with a vacuum gripper, and a 3D camera mounted on top of the bin, as presented in Figure 9. The CAD model of the parts is used to generate offline datasets, which are then matched against the point cloud image obtained from the 3D camera. The matching is done based on PPF descriptors and the accuracy of the pose estimation is increased by applying the ICP algorithm. Since multiple correspondences are obtained, for each reference point obtained, a voting-based scheme is applied, determining the best correspondence between the offline database and the captured scene. To increase the efficiency of the system, a multilayer adaptive threshold is adopted, since the goal is to always pick the part that is closest to the camera. The output depth map can be divided into several layers, based on the part parameters, and the search can be performed only on the top layer. This system is tested with four different parts that are previously known, displayed in mono-reference bins, achieving an average success rate of 91.25% and a maximum error of 1,02 mm in the x-axis direction, 1.0 mm in the y-axis direction and 1.91 mm in the z-axis direction.

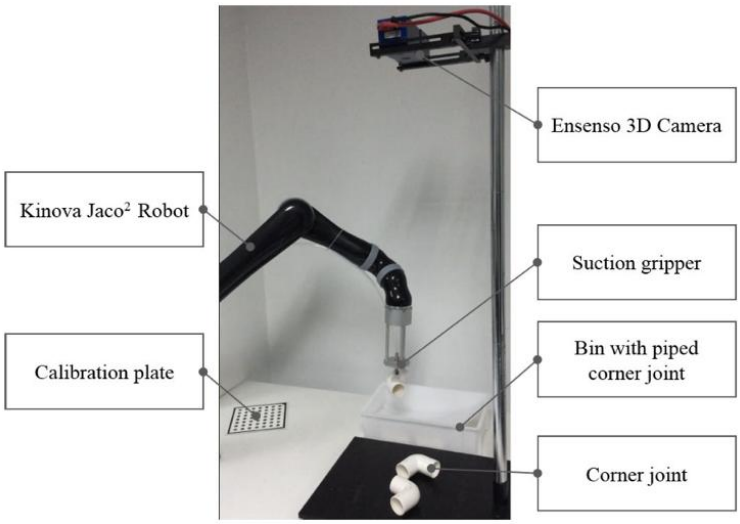

Figure 9: Test setup, consisting of a robotic arm, a suction gripper and a 3D camera (Yan et al., 2020).

2.3.1.2. Model-Free

In Domae et al. (2014), a general-purpose bin picking system is proposed. This system is equipped with a 3D sensor attached to a Mitsubishi Electric RV-6SL robotic arm and is tested with two different gripper types, a two-finger gripper, presented in Figure 10, and a vacuum gripper. First, the depth sensor outputs a depth map which is then segmented to extract the candidate regions. These regions are sorted by height in the bin, and the highest scores are chosen, with the assumption that the objects on top of the bin are more likely to

be successfully grasped. Next, two different types of masks are convolved onto the candidate regions, representing the contact region and the collision region for the gripper being used. Masks with respect to different in-plane rotations are also included in this step. Finally, a graspability map is obtained. The proposed system is tested with five different metallic parts in mono-reference bins achieving an average success rate of 82.47%, with 81.2% for the two-finger gripper and 83.75% for the vacuum gripper.

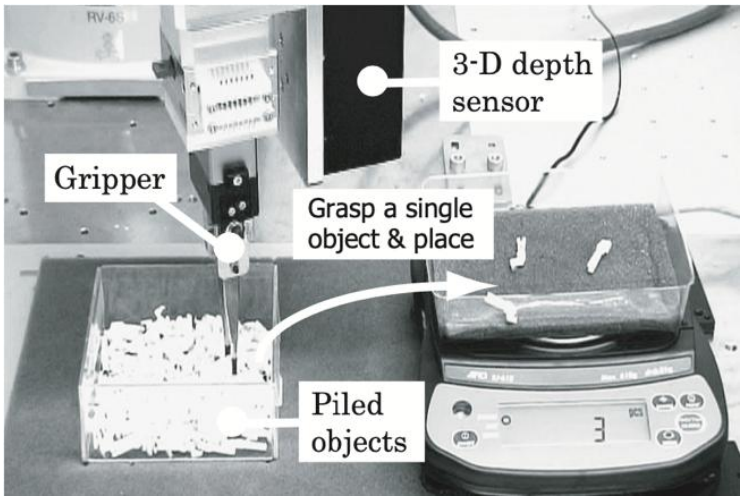

Figure 10: Experimental setup used in Domae et al. (2014), with a two-finger gripper and a 3D sensor.

In Buchholz et al. (2014), a bin picking system designed to handle unknown objects is proposed. The setup proposed consists of a Staubli RX60 robot with a parallel jaw gripper, a Microsoft Kinect mounted above the bin and a wrist-mounted sensor which provides 6D force-torque and 6D acceleration measurements. The depth images generated by the sensor are matched against the mask of the gripper to identify potential grasp points. To select the gripper orientation, a line is fitted into the patches found in the map and the depth of the parts to the camera is determined, then the object is picked. The proposed system is tested for two different conditions: unknown parts, and known parts with precise final placement. The first test is run on four different scenarios: only one type of object in the bin, two types of objects in the bin, and multiple, all different, objects in the bin in two distinct levels of clutter. The information obtained from the sensors mounted on the wrist of the robot is used to compute the centre of mass of the object and used to avoid collisions during the transfer process. The total grasp success rate is 95.4%. The second test is run on known parts, aiming to precisely place these parts. In this situation, the information from the 6D force-torque and 6D acceleration sensors is used to determine the pose of the part on the gripper. The average absolute error obtained is of 2.7º and an increase in cycle time is observed.

2.3.2. Deep Learning Approaches

2.3.2.1. Model-Based

In Le & Lin (2019), a bin picking system for planar-faced objects is proposed. The system performs three tasks: visual perception, 3D pose estimation and target picking. First, the dataset is created using 153 real scenarios obtained with a Kinect 3D camera and using a data augmentation technique to quadruplicate the amount of data in the dataset. Then, Mask Region-based Convolutional Network (RCNN) is trained to perform object segmentation over the RGB image and determining if the object inside the bin is upside up or downside up, outputting the object masks and bounding boxes. Next, the depth information of the scene is combined with the 2D information, and a predicted plane is constructed and used to determine the surface normal. This vector is going to determine the direction of approximation of the vacuum gripper, mounted on the wrist of a Denso 655G6 robot manipulator. The system is trained and tested on the setup previously described, and presented in Figure 11, achieving a 91.18% success rate for the object segmentation task and a 99% success rate for the picking task.

Figure 11: Test setup used in Le & Lin (2019), the robot separates the parts by the side that is facing up in the bins on the bottom part of the image.

In Araki et al. (2020), a complete bin-picking system that performs object detection, object segmentation and grasp detection in one pass is proposed. A Multi Shot - Deconvolutional Single Shot Detector (MT-DSSD) is developed based on a DSSD with an added layer to perform object segmentation and grasp detection. This network is trained on an Amazon

Robotics Challenge dataset that includes 40 objects and is manually labelled for grasp detection, requiring a couple of hours to complete the annotation. The network is compared with other methods, such as SegNet, U-Net and PSPNet, showing good results. Next, the proposed network is tested in a laboratory environment, using a Techman TM5-900 robot equipped with an Ensenso N10 depth sensor, a RGB camera and a vacuum gripper mounted on the robot wrist. The system described is represented in Figure 12. For each scene, several different objects were displayed on top of a table in a pile. When the object segmentation failed, the grasping detection also provided wrong results, which led the authors to conclude that the grasping points were learned by the object class. The system is tested for picking in three different scenes and achieved an average picking success rate of 61.67%, and a global accuracy of 90.49%.

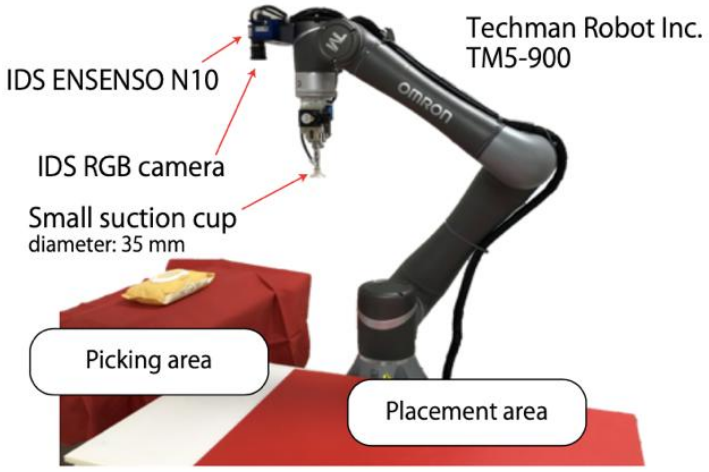

Figure 12: Test setup proposed in Araki et al. (2020), consisting of a robotic arm, a RGB camera, a depth sensor and a vacuum gripper.

In Zhuang et al. (2021), a bin-picking system that performs segmentation of the parts inside the bin is proposed. The object detection task is performed using a Part Mask RCNN which outputs a bounding box and a label from the RGB image input. Next, a Semantic Point Pair Feature (SPPF) gets the semantic point cloud from within the bounding box and determines the difference between the scene and the model coordinates. Two experiments are run with two types of known objects: daily objects and industrial objects. The networks are trained on synthetic datasets for both types of objects. For both experiments a KUKA IIWA robot with a vacuum gripper is used, Figure 13. For the first type of objects an Intel RealSense low-cost RGBD camera is used, and the bin is filled with several different objects; for the latter, a HDI 120 high-precision 3D scanner and a MER 132.30GX RGB camera and the bin

is filled with identical objects. The system achieved an average precision of 83.7% for the industrial part and over 83.6% for the daily objects.

Figure 13: Experimental setup consisting of a robotic manipulator and a vacuum gripper (Zhuang et al., 2021).

In X. Li et al. (2022), the authors propose a Sim-to-Real bin picking approach, where an instance segmentation network and a pose estimation network are trained on a synthetic dataset. The networks used are Mask RCNN and Dense Fusion, respectively. The dataset is created using the physics-simulation module from Blender. A total of 11 objects were trained, however only 4 objects are tested for both object detection and object grasp, the remaining 7 objects are trained from popular datasets for comparison purposes. To ensure precise segmentation on a cluttered environment, the networks are trained on the objects that have the highest visibility score in the scene, aiding in avoiding collisions, since the first objects to be detected and picked are less occluded and higher on the pile. The setup used in the tests consists on a RGBD camera, a Franka Panda robot and a vacuum gripper, represented in Figure 14, and achieved an average success rate of over 88.7%. As mentioned previously, this method is compared with other state of the art methods (PPR-Net and OP-Net) in average precision and achieved a better performance in 5 of the 7 objects used for comparison.

Figure 14: Setup consisting of a robot, a RGBD camera and a vacuum gripper (X. Li et al., 2022).

In G. Li et al. (2023), a bin picking framework combining digital planning and real-world testing is proposed. The CAD model of the part is imported into a digital planner that evaluates the part characteristics such as weight, dimensions and centre of gravity. The grasping area for the part is determined based on this information. A dataset is created with pictures of the part in different angles to train the pattern recognition tool, which is then fed to the TMflow programming environment included with the Omron TM5 collaborative robot. TMflow includes built-in machine learning tools for computer vision applications. The parts detected in the scene are ranked based on their vertical distance to the gripper, as to select the most efficient grasp, and from the closest distance to the top of the bin, to avoid collisions. Then, the system is tested on the RoboDK virtual environment to verify its feasibility and, finally, is tested in the real world using the mentioned collaborative robot, equipped with a suction cup and an FH-SMDA 3D vision system attached to the robot hand and mono-reference bins, as shown in Figure 15. A force/torque sensor is also added to the wrist of the robot to detect collisions and perform safe grasping.

Figure 15: Test configuration from G. Li et al. (2023), consisting of a robot, 3D vision system and a vacuum gripper.

2.3.2.2. Model-Free

In Jiang et al. (2020), a bin-picking system for textureless planar-faced objects is proposed. A deep convolutional neural network determines graspable points in the scene. This network is trained with synthetically generated depth images from three box models. The surfaces are detected using a surface feature descriptor and the centre point and surface normal are calculated for each surface. The images are also labelled for graspability, in two-stages: first at hand-level and then at robot-level. Using the hand-level approach first, the non-graspable points are quickly determined, and then the robot-level labelling only has to be performed for the remaining points, reducing on computation time. The setup used consists of a Toshiba TVL500 robot equipped with two-suction cups, which can be individually activated, and an Intel RealSense SR300, to acquire depth data. The experiments use 12 different boxes obtaining good results, concluding that the network can identify new objects that show similar features to the trained objects, achieving a success rate of over 94.6%. The experimental setup used to test this system is shown in Figure 16.

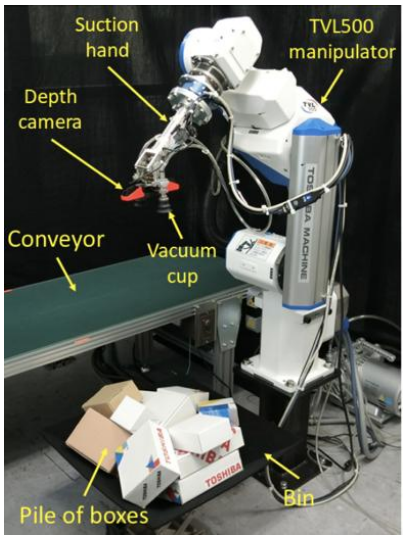

Figure 16: Experimental setup used in Jiang et al. (2020), consisting of a robotic manipulator, a camera and a gripper with two suction cups.

In Iriondo et al. (2021), the authors compare the performance of a Fully Convolutional Network (FCN) with a Graph Convolutional Network (GCN) in bin-picking applications with multi-reference bins. The dataset used to train these networks is developed using 37 parts from the Pick and Place EU Project (*Pick-Place* |, 2025), the data is annotated for each type of gripper, two-finger and vacuum gripper, and the images are captured with a Photoneo Phoxi M Camera. The first test is performed using only objects from the dataset, and both methods presented good performances, with GCN outperforming the precision score of FCN for the vacuum gripper, and FCN performing more precisely than GCN for the two-finger gripper. Next, a second test is run where the networks are tested against 100 new scenes with completely novel parts. In this test, both methods are able to perform good generalization for the vacuum gripper to correctly predict affordances for new parts. However, for the two-finger gripper, the performance of both networks decreased, most significantly the GCN. The results obtained are not tested on real picking applications.

In Gou et al. (2021), the authors propose a convolutional neural network called Angle-View Net (AVN), which generates gripper orientations in different positions of the image identifying five of the seven degrees of freedom of the grasp pose. The remaining two DOF are calculated using Fast Analytic Searching (FAS), consisting of taking the results with higher confidence scores and calculating the widths and distances to the image plane knowing the camera intrinsic parameters. In the experimental phase, the objects are not located inside a bin, but instead on a cluttered environment with some occlusions, the setup used in the trials consists on a UR5 robot equipped with a RGBD Intel RealSense Camera

and a two-finger gripper on the hand of the robot, as shown in Figure 17. The proposed network is trained using GraspNet-1Billion dataset (*Datasets*, 2021) and tested using both objects from the dataset and unknown objects, each scene had 4 to 8 different objects. Finally, the trajectory of the robot is calculated using MoveIt to avoid collisions. This approach presented an average success rate of 91.1%.

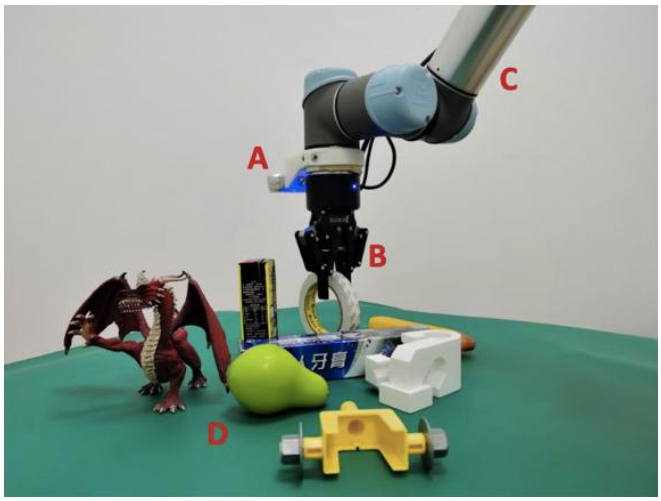

Figure 17: Example of a scene used to test the system developed in Gou et al. (2021), including a manipulator robot, a RGBD camera and a two-finger gripper.

In Wang et al. (2021), a quality measure, “graspness”, is introduced. The ResUNet14 is adopted to detect the objects, extracting both global and local point features. Additionally, an end-to-end network, called Graspness-based Sampling Network (GSNet), that incorporates the "graspness" measure into the grasp pose detection task is proposed to increase efficiency of the object grasp. This network is implemented with PyTorch and is trained with the publicly available dataset GraspNet-1Bilion. The output of the GSNet contains scores and widths for different combinations of in-plane rotation/approach depth, the highest score is picked, and the grasp generation is done using PointNet. The setup proposed consists of a UR5 robotic arm, equipped with an Intel RealSense SR435 Camera and a 2-finger gripper, the objects on the scene are chosen from the dataset.

In Jiang et al. (2022), a bin-picking system based on grasp quality and reachability is proposed. The authors train a SG-U-Net++ network with a synthetic dataset created using PyBullet and CAD models of over 700 different parts. Grasp quality is defined as surfaces that are flat and smooth rather than curved and where the vacuum cup can make full contact with the surface. In the trials, 13 different objects, all novel to the system, are placed inside the bin and the picking task was performed with a TVL500 Shibaura Machine CO., Ltd, with

two vacuum suction cups, and an Intel RealSense SR300 camera mounted on the wrist of the robot, Figure 18. However, the trigger for the camera is always in the same position above the bin. The proposed system achieved a success rate of over 94.6% and a shorter time for grasp planning when compared to Dex-Net.

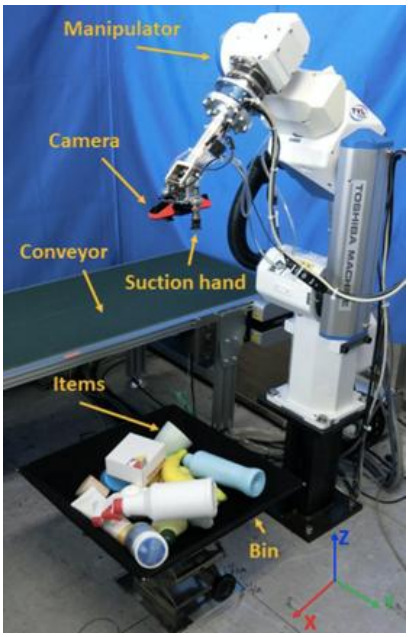

Figure 18: Test configuration from Jiang et al. (2022), consisting of a robotic manipulator, a camera and two suction cups.

In Raj et al. (2024) a bin-picking system for novel objects is proposed. A Convolutional Neural Network (CNN) is trained to predict graspable locations in the scene, then outputs the regions that are more likely to have successful grasp attempts. After that, the grasp poses are sampled uniformly in the image plane in multiple directions and, finally, a grasp evaluator picks the one that is more likely to result in a successful grasp. The network is trained on synthetic data, developed by the authors using PyBullet and an open-source repository. The setup consists of a UR5 robotic manipulator equipped with a two-finger gripper, and a RealSense D435i camera mounted above the bin, as depicted in Figure 19. Then, the bin is filled with different levels of clutter with multi-reference novel objects. To avoid collisions, the grasp approach is done vertically and the force sensors mounted on the wrist of the robot are used to determine the depth that the gripper should go to grip the object successfully and without colliding or damaging the other objects in the scene. The proposed network achieved a grasp success rate of over 87% for high clutter scenes in a time efficient manner.

Figure 19: Experimental setup used in Raj et al. (2024).

2.3.3. Hybrid Approaches

2.3.3.1. Model-Based

Tajima et al. (2020) presents a bin-picking system developed to participate in the World Robot Challenge 2018. The setup proposed is designed to pick 15 different objects in different bins, and consists of a parallel gripper, equipped with tactile sensors on the tips of the fingers, installed on the wrist of a 6DOF robot arm. The tactile sensors are used to confirm if one single object was grasped from the bin. The camera used is an RGBD Intel RealSense SR300 mounted on the wrist of the robot, and each bin was only filled with identical parts. Since the parts are very different from each other, different approaches are taken for each group. For larger parts, LINEMOD pipeline (Hinterstoisser et al., 2011) was used to perform object detection, and Hough Transform for the smaller parts. For the Object Grasp task, the parts were divided into four categories. The position of the gripper is then computed from the grasp patterns detected. After that, the joint positions of the robot are calculated from the gripper position and, finally, all the grasp patterns without collisions between the bin and the gripper are extracted. The proposed system is robust, since it is able of picking each part, but not time-efficient, not being able to complete the task in time, and achieving 4th place in the competition.

In Park et al. (2022), the authors propose a bin-picking system based on a Convolutional Neural Network trained on human-labelled data and self-training. The setup consists of a 6DOF collaborative robot with a parallel gripper and a LiDAR sensor to capture the

information about the environment, as seen in Figure 20. In the proposed approach, 100 depth images of the bin are acquired and labelled by a human operator and the results are used to train and implement the CNN. Next, the ICP algorithm is used to estimate the pose of the detected object. Since no CAD model is available, a partial point cloud of the object is previously obtained from real depth data. This system is only trained and tested on a singular type of object. To avoid collisions, a margin of 30 pixels is added to the object detection. All hardware is controlled by Python scripts and OpenCV is used for data pre-processing. The robot learned each time 100 data was accumulated, increasing its initial 74% of success rate to 81.4% with 2000 data accumulated.

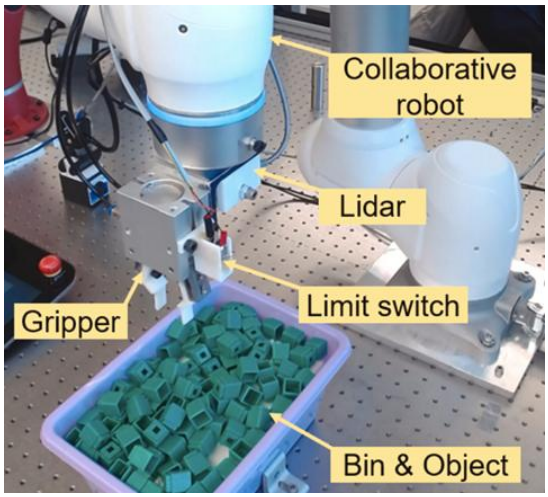

Figure 20: Test configuration for the system proposed. Consists of a robot, a LiDAR sensor and a two-finger gripper (Park et al., 2022).

In Xu et al. (2024), a bin picking system capable of detecting and classifying Lego bricks by colour is proposed. First, a point cloud image of the scene is captured which is then processed using a commercial tool, where the geometrical features of the object are trained, which outputs a 3D bounding box of the position of the object. The pose estimation is done using the ICP algorithm. To remove noise and other non-important features, the best-fit method is applied. Finally, the object is picked, but the authors do not elaborate on the methodologies used for path planning and collision avoidance. After the part is picked, it is positioned in front of the RGB camera to determine the colour of the block and then placed inside the corresponding bin. The proposed setup consists of a stationary 3D camera, an RGB camera to classify the Legos by colour and an UR10 robot with a vacuum gripper, shown in Figure 21.

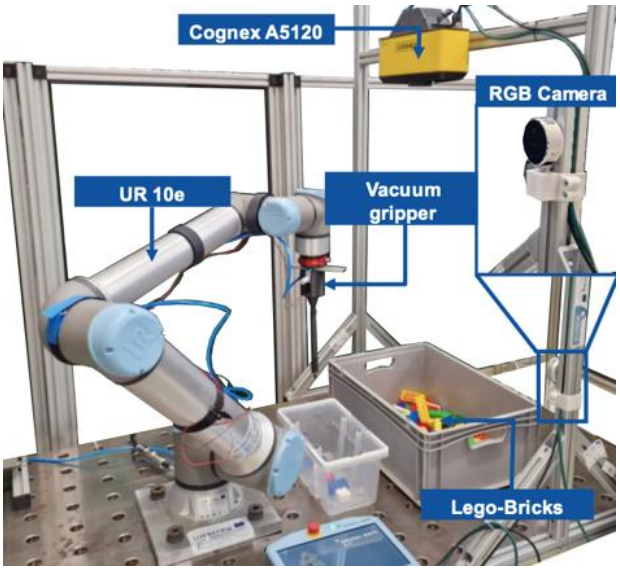

Figure 21: System developed in Xu et al. (2024), consisting of robotic arm, a 3D camera, a RGB camera and a vacuum gripper.

2.3.3.2. Model-Free

Sun et al. (2024) focuses on the development of a bin picking system using a low-cost RGBD camera. The setup used is a 7DoF Franka Panda equipped with a parallel-jaw gripper and an Intel RealSense D435 camera, as seen in Figure 22. A network called MS-ResUNet is designed to predict the local grasp path heatmap and trained using synthetic data. This data is generated using PyBulet and OpenGL, with approximate 3D CAD models of industrial parts. The Canny operator is applied to the input image before feeding it into the MS-ResUNet to increase the performance of the network and mitigate the negative effects of training on synthetic data, such as reducing the domain gap between the synthetic data and real data. First, Mask RCNN is used to detect the object's localization and outputs the bounding box. The experiments are performed on 6 different objects, in bins filled with a single type of object. However, the network is only trained on three of the objects, since the three remaining ones are similar to one of the already trained ones and so the same weight is used.

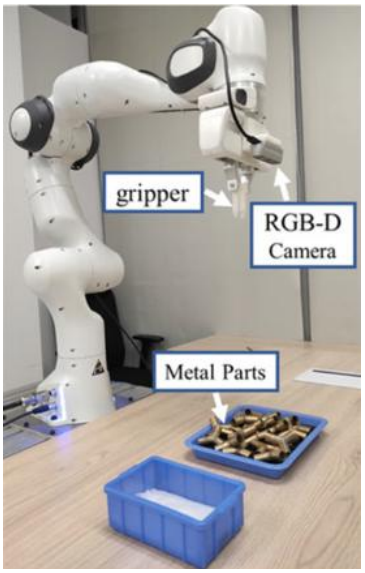

Figure 22: System proposed in (Sun et al., 2024), consisting of a robot, a RGBD camera and a parallel-jaw gripper.

2.4.Insights

This study provided valuable insights on available methods for computer vision, different setup configurations, strategies to ensure that all the parts in the bin can be picked, strategies to avoid collisions and gripper configurations.

Classical computer vision techniques were mostly used in mono-reference bins filled with previously known objects, providing good results, since they are manually tuned for that specific application and parts, and for the environment conditions and changes. A very small number of articles used classical model-free methods, considering that it is much more challenging to determine which features are distinctive enough for parts that are not known. However, model-free approaches generally tend to look for graspable areas more than specific part features, which allows to look for planar surfaces when a vacuum gripper is employed and space between parts when a two-finger gripper is used.

On the contrary, deep learning methods are mostly used as model-free approaches for unknown objects in multi-reference bins since convolutional neural networks can more easily generalize the knowledge obtained from a well-constructed dataset to different scenarios. Deep learning methods are typically used in applications where object segmentation is needed, which means that not only is the object detected in the scene, but its class is identified as well. These approaches are not used as much for model-based

applications since they tend to be simpler, not justifying the computational power, as well as the time and effort needed to acquire and annotate the data for training.

An overview of the methods used, based on the types of objects inside the bin is provided in Table 2. A summary of each presented system along with the type of objects in the bin, type of gripper and sensors used is provided in Appendix A.

Table 2: Quantity of articles analysed based on methodology used and type of parts to handle, mono-reference or multi-reference bins.

Methodology	Mono-reference	Multi-reference	Total
Classical Methods, Model-Based	5	1	6
Classical Methods, Model Free	2	1	3
Deep Learning, Model-Based	4	1	5
Deep Learning, Model Free	0	6	6
Hybrid, Model-Based	3	0	3
Hybrid, Model Free	1	0	1
Total	15	9	24

From the analysed systems, certain similarities in the hardware used were observed, such as: 6 DOF robotic manipulators, allowing for flexibility in the position of the gripper in the movement of the robot inside and out of the bin; vacuum and two-finger grippers; cameras, instead of laser and depth sensors, since they provide more information to the system. To avoid collisions, the most common approach is to use a vertical gripper approach to the part and aiming to grip the topmost object in the pile.

To increase efficiency of the bin picking systems several different approaches are used, such as:

- Placing the camera on the robot, which increases flexibility in the points of view of the scene, allowing for better adaptation to occlusions;
- Performing object detection only in a 2D image and using the point cloud information on the identified bounding boxes to determine the position of the part in the scene, reducing the computational power/complexity;

- Shaking of the bin to move the parts when no more parts can be identified but the bin is still not empty.

Finally, different sensors are used to provide additional information about the environment, like force/torque sensors to detect collisions and avoid damaging the parts inside the bin, tactile sensors to detect the presence of more than one object on the gripper, and secondary cameras to detect colour and to reposition parts when a very precise placement of the part is needed/crucial.

3. Overview of Vision Technologies

In this chapter, an explanation of the computer vision techniques employed, and the deep learning model implemented is presented, providing essential background knowledge on the subject. The system is built using the OpenCV library for implementing classic methods, and the Ultralytics Python framework for the development of the deep learning model used (Yolo11).

3.1. Classic Computer Vision Methods

3.1.1. Threshold

Thresholding is a computer vision technique used to binarize images so that they can be further analysed using other methods. This technique compares the pixel values of an image to a predefined value. This technique consists of comparing each pixel in the input image to the threshold value. If the pixel value is below the given threshold, the corresponding pixel in the output image will be 0 (black); if it is above the threshold, it will be 1 (white) (*OpenCV: Image Thresholding*, 2025).

3.1.2. Adaptive Threshold

In global thresholding, the same threshold value is applied to the entire image, thereby diminishing robustness to lightning changes. Adaptive thresholding is a more appropriate method in these cases, where a different threshold is used for each image region based on the region's overall pixel values. Two modes are available on OpenCV: in one mode the threshold value is determined based on the mean value of the pixels in the defined region, while the other is based on the Gaussian sum. Figure 23 compares basic thresholding, adaptive mean thresholding, and adaptive Gaussian thresholding (*OpenCV: Image Thresholding*, 2025).

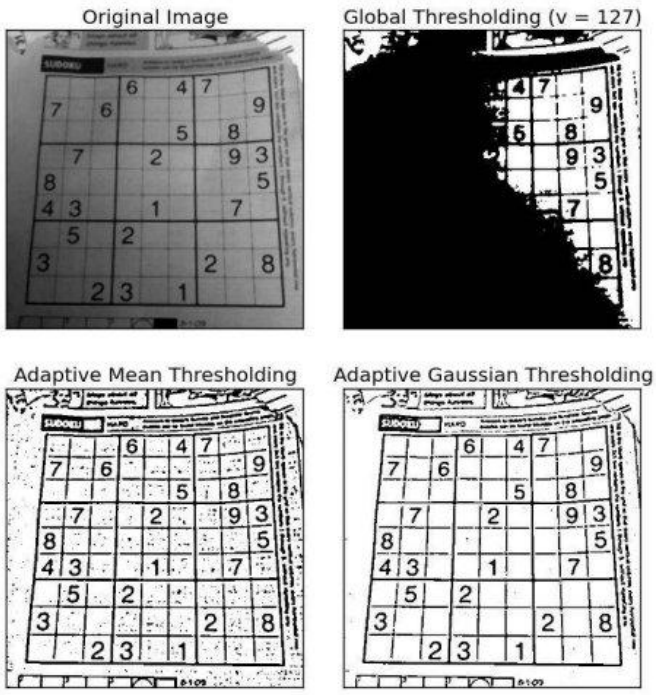

Figure 23: Comparison between basic thresholding and the two adaptive thresholding modes (*OpenCV: Image Thresholding*, 2025).

3.1.3. Canny Edge Detector

The Canny Edge Detector is a multi-stage algorithm that detects edges in an image using gradient and hysteresis thresholding. Essentially, this algorithm applies the gradient to the image to identify areas most likely to be edges, then, based on the minimum and maximum values provided, determines which are strong edges and which are not (*OpenCV: Canny Edge Detection*, 2025).

3.1.4. Contours

A contour is a curve joining all points along an edge and is useful in tasks such as object detection and recognition. The analysis of contours is typically done in binary images, such as those obtained after applying thresholding techniques or edge detectors (*OpenCV: Contours : Getting Started*, 2025).

3.1.5. Template Matching

Template matching is a technique that aims to find the location of a template in an image. This function slides the template over the input image using 2D convolution. It compares the two images and returns a grayscale image that represents the similarity between each region of the input image and the template. Different comparison modes are available to

better adapt to each application: sum of square differences, cross correlation and correlation coefficient (*OpenCV: Template Matching*, 2025; *What Is Template Matching?*, 2024).

3.1.6. Morphological Operators

Dilation and erosion are two morphological operators that can be used to remove noise, isolate individual elements, or join separate elements (*OpenCV: Eroding and Dilating*, 2025).

Essentially, dilating consists of convolving a kernel over an input image. As the kernel scans the image, the maximum pixel value is computed, and this new value replaces the pixel at the centre of the kernel. Erosion is the inverse operation, where the minimum pixel value is computed and the kernel's centre point is replaced with this new value. Figure 24 provides examples for both approaches (*OpenCV: Eroding and Dilating*, 2025).

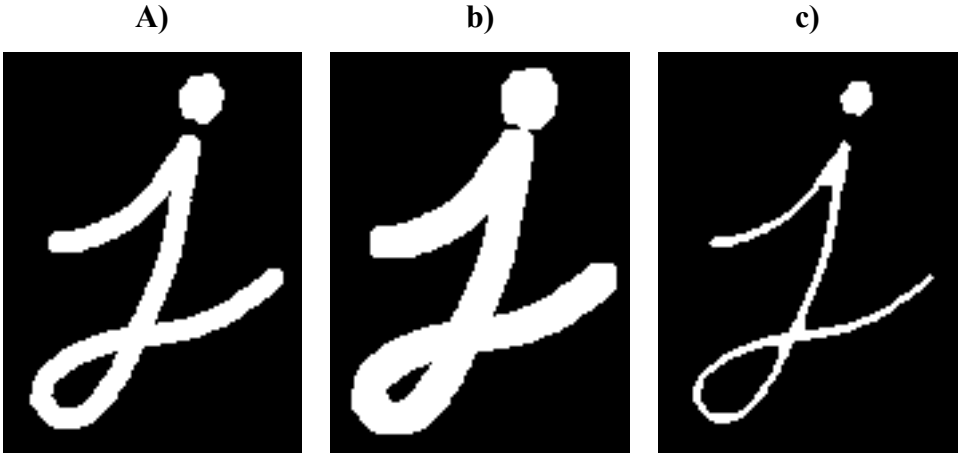

Figure 24: a) Original Image, b) Dilated Image, c) Eroded Image.

3.2.Deep Learning Model

3.2.1. Model

YOLO11 is one of the latest iterations of the Ultralytics YOLO series of real-time object detectors. Compared with previous versions, YOLO11 offers enhanced feature extraction capabilities, optimised speed, greater accuracy, and supports tasks such as object detection, instance segmentation, pose/keypoint estimation, oriented object detection, and image classification (Ultralytics, 2026f).

This model has five variants from nano (n), for small and lightweight tasks, to extra-large (x) for complex tasks and maximum accuracy and performance, with small (s), medium (m) and large (l) in between. It also has six different modes:

- **Train**, to fine tune the model on custom or preloaded datasets;
- **Val**, to validate the model's performance after training;
- **Predict**, to apply the model over the desired image;
- **Export**, to deploy the model in various formats; track, to extend the model to real tracking applications;
- **Benchmark**, to analyse the speed and accuracy of the model in different deployment environments (Ultralytics, 2026f; Ghosh, 2024; Ultralytics, 2026d).

3.2.2. Dataset

To train a custom YOLO11 model for a specific project, it's necessary to develop and annotate a custom dataset. Roboflow is an online platform that allows users to annotate and prepare datasets, but also to train and run not only YOLO but also other computer vision and artificial intelligence models (*Roboflow*, 2025).

Additionally, preprocessing techniques such as grayscale conversion, auto-orientation, resizing, and augmentation (e.g., flip, rotation, shear, noise, and more) can be applied to the dataset directly through Roboflow to improve the model's performance (*Roboflow*, 2025).

After downloading the dataset from Roboflow, the folders should be adjusted to match the Ultralytics YOLO11 requirements. There should be two main folders, "images" and "labels", and inside each, the "train", "val", and "test" folders, as shown in Figure 25. The configuration of the dataset is done with an .yaml file that is included in the downloaded dataset files. This file should be updated with the paths to the training, validation, and test

sets. This file also contains the information of the number of labels used to annotate the dataset and their designation.

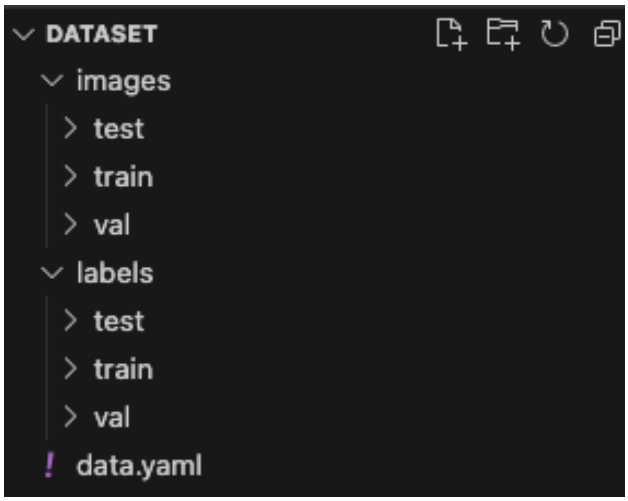

Figure 25: Structure of the dataset folders.

The labels are provided in .txt files with the same filename as the corresponding image. In these files, the object's class is provided, along with a few values for the object's outline. For bounding box annotations, the outline is stored as: x_center, y_center, width, height; for segmentation annotations: x1 y1 … xn yn; and for oriented bounding boxes: x1 y1 x2 y2 x3 y3 x4 y4 (*Oriented Bounding Box (OBB) Datasets Overview - Ultralytics YOLO Docs*, 2025; Ultralytics, 2026c).

3.2.3. Training

The model can be trained on different platforms, such as Roboflow (as mentioned before), Google Colab, or even Visual Studio Code, by defining the model variant to use, the dataset, and the number of epochs. This last argument specifies the number of times the training is run over the entire dataset. Additionally, other arguments can be provided to increase the model's performance (*Model Training with Ultralytics YOLO - Ultralytics YOLO Docs*, 2025).

- **Batch size** defines the number of training samples the model processes before updating its internal parameters. Larger batch sizes provide more accurate parameter estimates at the cost of greater computational complexity. On the other hand, smaller batch sizes lead to less accurate parameter estimation but more frequent updates. This parameter affects the model's learning dynamics, training speed, and performance (*Batch Size in Deep Learning | Ultralytics*, 2025).

- **Patience** parameter defines the number of epochs without improvement in the model before early stopping training, preventing overfitting (*Model Training with Ultralytics YOLO - Ultralytics YOLO Docs*, 2025).
- **Image size** resizes the dataset images to a specified value before presenting them to the model. This parameter affects model accuracy and computational complexity, since downsizing images normally results in some information loss, decreasing both (*Model Training with Ultralytics YOLO - Ultralytics YOLO Docs*, 2025).
- The **pretrained** parameter allows training an already trained model to increase its efficiency and performance without retraining the entire model (*Model Training with Ultralytics YOLO - Ultralytics YOLO Docs*, 2025).
- The **box** parameter defines the significance of accurate bounding-box coordinate predictions. The **class** parameter is similar, determining the importance of the accurate class prediction (*Model Training with Ultralytics YOLO - Ultralytics YOLO Docs*, 2025).

3.2.4. Validation and Metrics

Ultralytics YOLO11 provides a few metrics to assess model performance, such as: confidence score, intersection over union, mean average precision, accuracy, F1-score, precision, recall, confusion matrix and loss (Ultralytics, 2026e; Ultralytics, 2026b).

- The **confidence score** represents how certain the model is that a particular detection belongs to a given class (Ultralytics, 2026b).
- **Intersection over Union** (IoU) measures how well the model's bounding box overlaps with the ground truth bounding box. This metric compares the detection boundaries with the object's actual boundaries (Ultralytics, 2026b).
- **Mean average precision** measures the model's overall performance. In other words, it measures how precisely the model detects each object class. This metric measures not only the correctness of the classification but also the localisation accuracy of the predicted object boundaries. For mAP50, the mean average precision is measured at an IoU threshold of 0.5, indicating looser accuracy requirements. For mAP50-95, it analyses across multiple IoU thresholds from 0.5 to 0.95 (Ultralytics, 2026e; Ultralytics, 2026b; *Accuracy*, 2025) .
- **Precision** measures the proportion of correct positive predictions (*Accuracy*, 2025).

- **Recall**, also known as sensitivity, measures the proportion of positives that the model correctly identified (*Accuracy*, 2025).
- **Accuracy** is a metric that represents the proportion of correct predictions to all predictions; however, it is not a good metric to use with imbalanced datasets, where there is significantly more data for one (or more) labels (*Accuracy*, 2025).
- **F1-score** is a better metric for imbalanced datasets; it is the harmonic mean of precision and recall, balancing both metrics (*Accuracy*, 2025).
- A **confusion matrix** visualises the model's performance by showing counts of true positives, false positives, true negatives, and false negatives. It provides data for calculating accuracy, precision, and recall (*Accuracy*, 2025).
- **Loss** metrics measure the error for the classification task; objectness loss measures the error in detecting whether an object is present in a particular grid cell; and location loss measures the error in localising the object within the grid cell (Ultralytics, 2026a).

4. Experimental Setup

The system developed in this project aims to detect the parts inside a bin, orient and place them precisely in preparation for placement inside an injection mould, three per cycle. The injection process cycle time is 90 seconds; therefore, the maximum allowable cycle time for this system is 30 seconds per part. In the following chapters, the three-part set is denominated by batch.

4.1. Target Parts

As mentioned in Chapter 1.1, the target part to be handled by the system is made of black, stiff fabric, with approximate dimensions of 45 by 22 mm, a large curvature and an indented border, as seen in Figure 26.

Figure 26: Target part.

It was necessary to develop a manufacturing process for the parts, since the supplier could not provide them within the required timeframe.

The stiff fabric is used in the automotive industry primarily for its acoustic properties. According to the information provided by the material supplier "L&L Acoustics", the material processing conditions are as follows:

1. Place the material in an oven at 200ºC for 5 minutes
2. Place the material into the mould with the desired shape and apply pressure until it cools.

Before placing the parts in the oven, the material was cut to the desired size and shape, as shown in Figure 27. Two moulds were manufactured in different materials, aluminium and PLA (Polylactide), both visible in Figure 28, which provided similar results. Despite the good results, the PLA mould suffered significant degradation due to the part's temperature.

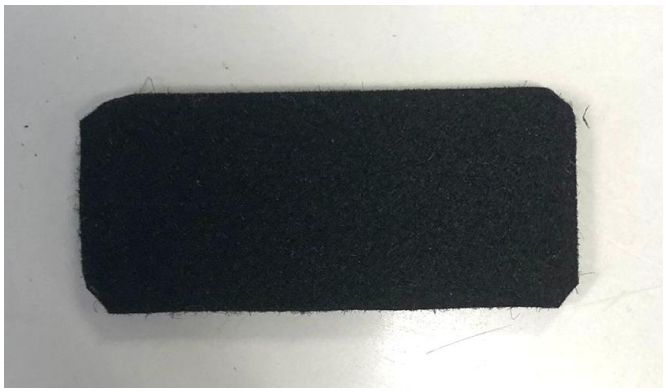

Figure 27: Material cut to the size of the mould.

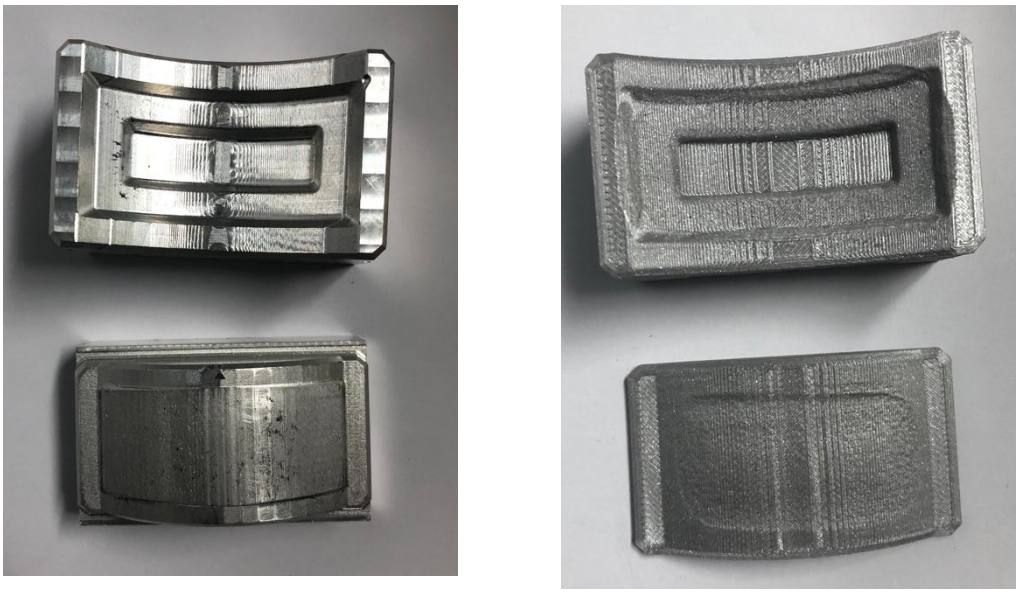

Figure 28: On the left, the aluminium mould and on the right, the PLA 3D printed mould.

Upon heating, the material exhibited a significant increase in thickness and a change from a stiff pad to a soft, malleable material. Within a few seconds of removing the part from the oven, the material started to harden. As a result, the window of time available to place the part in the mould was reduced, leading to significant variability (in thickness, curvature and appearance of the indented border) between parts, seen in Figure 29. These problems are expected to be solved in the industrial application, since the final pads will be systematically produced by the supplier of the base material.

Figure 29: Manufactured parts

4.2.Hardware

Earlier in the project's development, it was determined that, for bin-picking applications, the flexibility of a 6 DOF robot is extremely important for adapting to an unknown environment. Similarly, the ease of integrating an all-in-one sensor, using a 3D camera instead of multiple sensors, is also an advantage. As a result, and given the equipment available in the laboratory, an ABB IRB 1200 robotic manipulator and a RealSense D415 camera were chosen.

Different gripper technologies for this specific material were studied in Vivek Kumar's master's thesis, which concluded that the most suitable option is a high-flow vacuum gripper (Kumar, 2024).

With all of this in mind, the gripper was developed as presented in Figure 30, and is composed of:

- Main support;
- Camera;
- Customized suction cup;
- Valve.

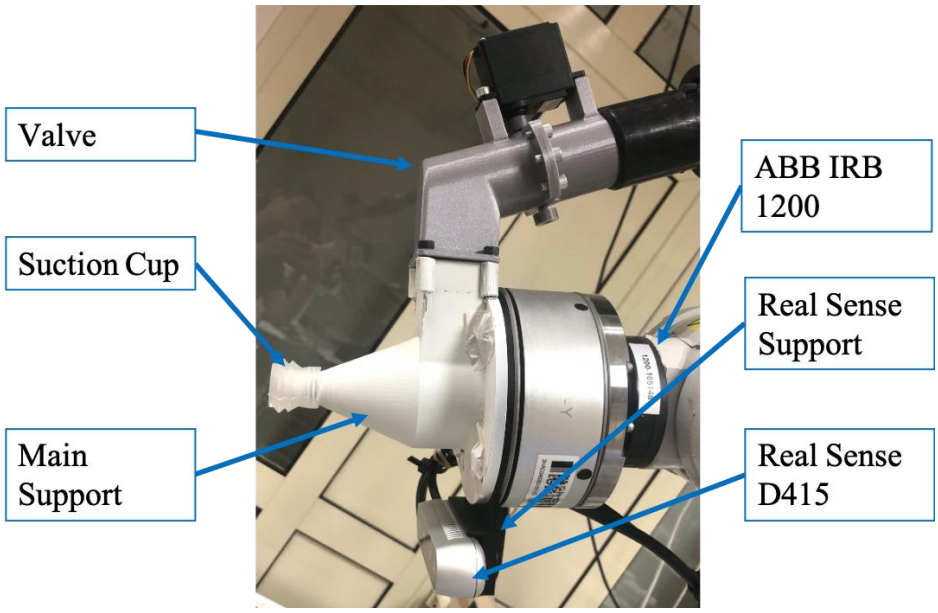

Figure 30: Gripper Configuration.

All the parts developed for this project were designed in a CAD software and 3D printed in PLA.

The main support fulfils a variety of functions. This part is hollow on the inside to allow air to pass from the valve to the suction cup, which were connected to this part as seen in Figure 30. Additionally, it served as support for the 3D camera. This support was fixed to the ABB Force Sensor mounted on the robot flange; however, this sensor was not used in the proposed solution, so a different fixation method should be used when mounting it directly to the robot flange. The main support was also painted in white, the rationale for this will be discussed further in Chapter 4.4.

For this setup, an 8 cm-deep cardboard bin was used to hold the parts. When using a different bin, the gripper length and overall design should be adjusted to ensure that all parts in the bin can be picked without collisions.

The decision to mount the camera on the gripper intended to add more flexibility to the system, allowing it to capture images across different areas of the workstation as needed,

without requiring additional sensors or cameras to gain deeper insight into the working environment.

A tailored suction cup was designed to improve the gripper's performance for the curved shape of the part. The mould used to cast the part is shown in Figure 31, and the final result is presented in Figure 32.

Figure 31: Mould to produce the personalized suction cup.

Figure 32: Suction Cup

The subsequent key component of this system was flow control to correctly pick and place the parts. The high-flow vacuum was produced with the assistance of a vacuum cleaner, and it was necessary to add a mechanism to open and close the flow as needed, resulting in the creation of a personalised valve.

The initial prototype consisted of two parts for the valve body, seen in Figure 33, a disc and a connector to the servo motor, as depicted in Figure 34.

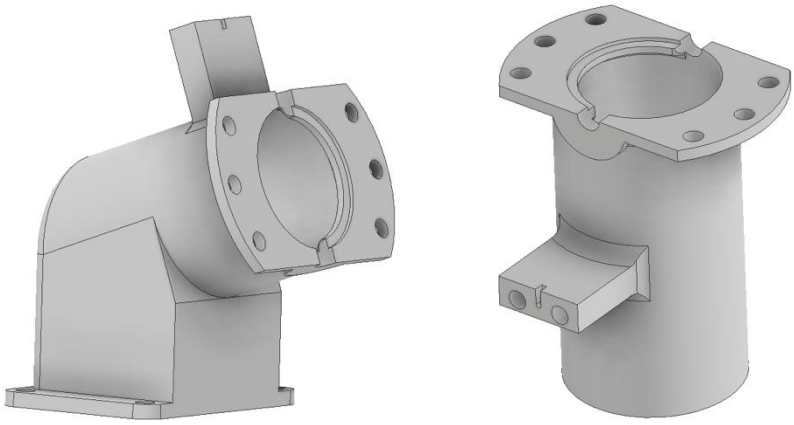

Figure 33: Body of the butterfly valve.

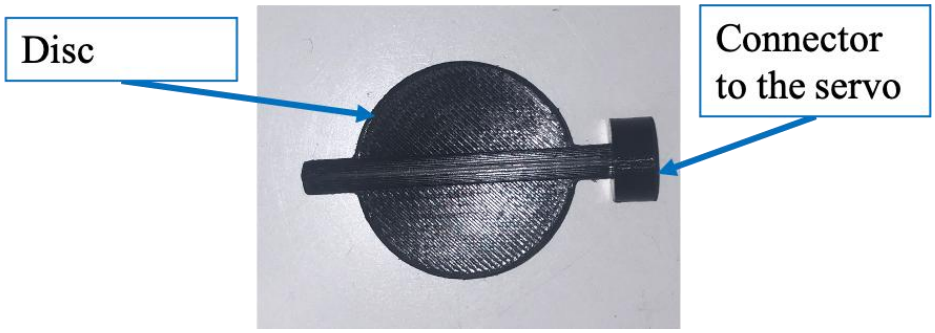

Figure 34: Disc of the butterfly valve and connector to the servo motor.

Following the results of the first tests of this prototype valve, two main issues were discovered. First, with the valve in the closed position the air flow through the gripper was still enough to maintain the part attached. Second, the connection between the connector to the servo and the disc was slightly loose, which resulted in the connector sliding over the disc axis and the disc losing its position as cycles passed.

With this in mind, a few alterations were made. To solve the first issue presented, the disc was separated in two halves and a rubber washer was placed in between, shown in Figure 35 and Figure 36. This made it possible for the disc to seal effectively around the walls of the valve. To solve the second problem, a slot was added to the connector to the servo and to the disc part, to ensure the position of the disc at all times, shown in Figure 37.

Figure 35: Components of the second version of the butterfly valve.

Figure 36: Disc with rubber washer and connector to the servo motor.

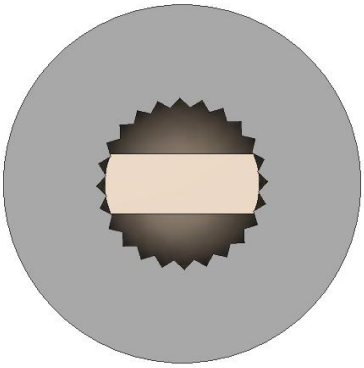

Figure 37: Connector to the servo with a slot.

The implemented changes effectively solved the issues found in the initial prototype of the valve.

The disc's motion was enforced by an HS-322HD servo motor, attached to the connector and fixed to the body of the valve, as seen in Figure 30. The servo was then controlled by a

Seeed Xiao RP2350 microcontroller, programmed in the Arduino IDE to rotate to 0º and 90º via serial communication.

In Figure 38, the complete system is represented with all the connections and communication between the mentioned devices. The robot and the computer communicate through a TCP/IP (Transmission Communication Protocol/Internet Protocol) connection. The Real Sense camera and the microcontroller were connected to the computer and were powered through USB ports. The servo motor was powered using a laboratory power supply and the microcontroller's ground was also connected to this power supply. The microcontroller controlled the servo motor through one GPIO (General Purpose Input/Output) pin.

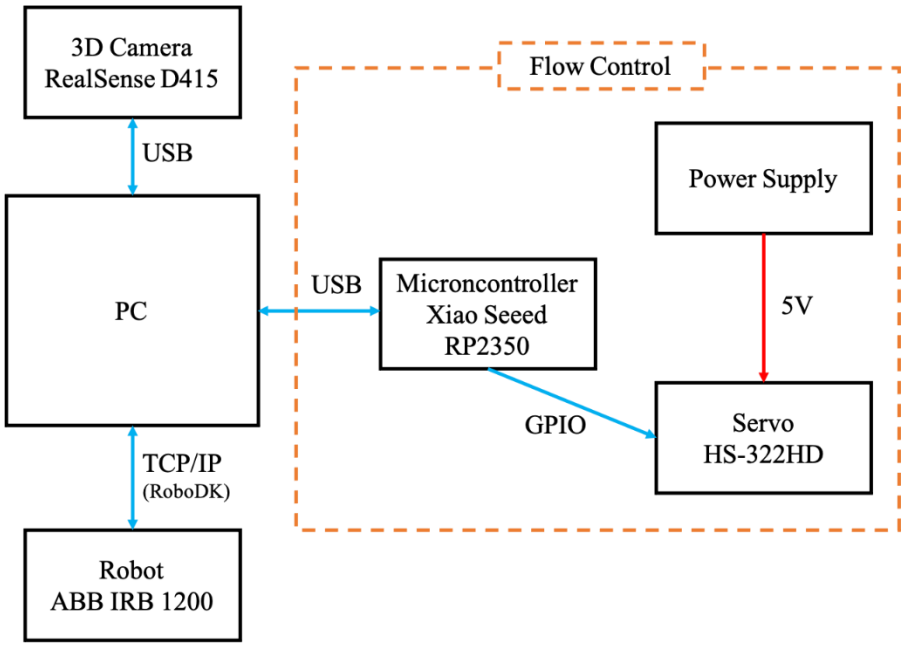

Figure 38: Block diagram of the setup.

For the final stage, the placement of the parts had to be defined. In this case, it was necessary to develop a buffer to accumulate the three parts required per cycle for delivery to the Cartesian robot. Furthermore, the goal was to create a buffer system that allowed the placement of parts independently of which side the robot picked them from, whether upside down or right side up. In Figure 39, the final buffer system is visible.

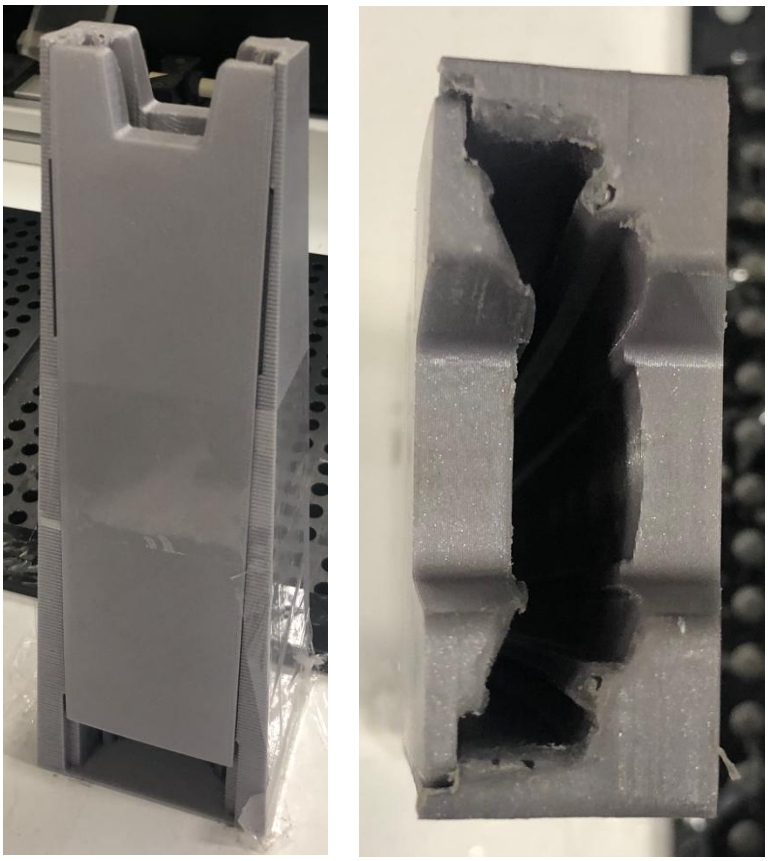

Figure 39:Buffer, front view on the left and top view on the right.

The parts were placed in the buffer horizontally with the curvature perpendicular to the table plane, as demonstrated in Figure 40.

Figure 40: Robot placing the parts in the buffer, according to the side from which the robot picked the part.

As well as the considerations just mentioned, other design choices were made that are important to the correct functioning of the buffer:

- Use of grooves, to reduce the total area of contact between the part and the buffer. Due to the manufacturing process used the surface of the buffer presents high surface friction;
- Wider opening than the dimensions of the part to allow for correction of the placement position after assessing the actual picking position;
- Angled path downwards to better guide the parts and a 90º at the end of the buffer to prevent the parts from exiting the buffer unassisted.

4.3.Dataset Creation and Model Training

A custom dataset was created to train a deep learning model, YOLO11, as detailed further in Chapter 3.2. A total of 1024 images with different backgrounds, lighting conditions, and number of parts per image (from 0 to 20 fully visible parts) were annotated in the Roboflow platform using the Polygon Tool, outlining the parts as precisely as possible and annotating only fully visible parts. Two labels were used to distinguish between parts in the up and down positions, “top” and “bottom”; some examples are shown in Figure 41.

Figure 41: Example of annotated images. Parts labelled as “top” identified in red and “bottom” identified in purple.

Moreover, preprocessing and augmentation techniques are used to increase the quality of the dataset, as suggested by Roboflow: auto-orient and resize for preprocessing, and flip and rotation for augmentation. Since no issues were felt with this approach, no other techniques were tested. As a result, the dataset expanded to 3222 images, divided into a train and a validation set, containing 2864 and 358 images, respectively. This represents 89% of images in the test set and 11% in the validation.

Given that this project aims to develop a complete solution to the identified problem, a test set for the model was not utilized. Instead, the complete system performance was assessed in Chapter 5 through experimental trials.

Conventionally, datasets was split as follows: 70% for the training set, 20% for the validation set, and 10% for the test set. Considering that no test set was used, the validation set was initially set to 30%; however, since the augmentation techniques were applied only to the training set, the final proportion was reduced to 11%.

The model was built using the nano variant because it provides lower latency than the other variants, which was important given cycle-time constraints, and uses less computational power during training. The chosen task was “oriented bounding boxes” since it adds an extra parameter to the detection, the angle of the best-fitted bounding box, when compared to the most commonly used task of “object detection”. This model was trained for 50 epochs.

The obtained confusion matrix, normalized by columns, is depicted in Figure 42, and it is possible to observe that the number of true positives is 0.78 for parts labelled “bottom”, and 0.88 for parts labelled “top”. This difference may be justified by an imbalance in the number of annotations for each label or by a smaller number of key features for the “bottom” label.

Several model detections were identified as background in the ground truth annotations. However, these detections corresponded to partially occluded parts, which, for the purpose of this project, are undesirable, as the accurate determination of the picking and placing positions needs the precise location of the part’s centre.

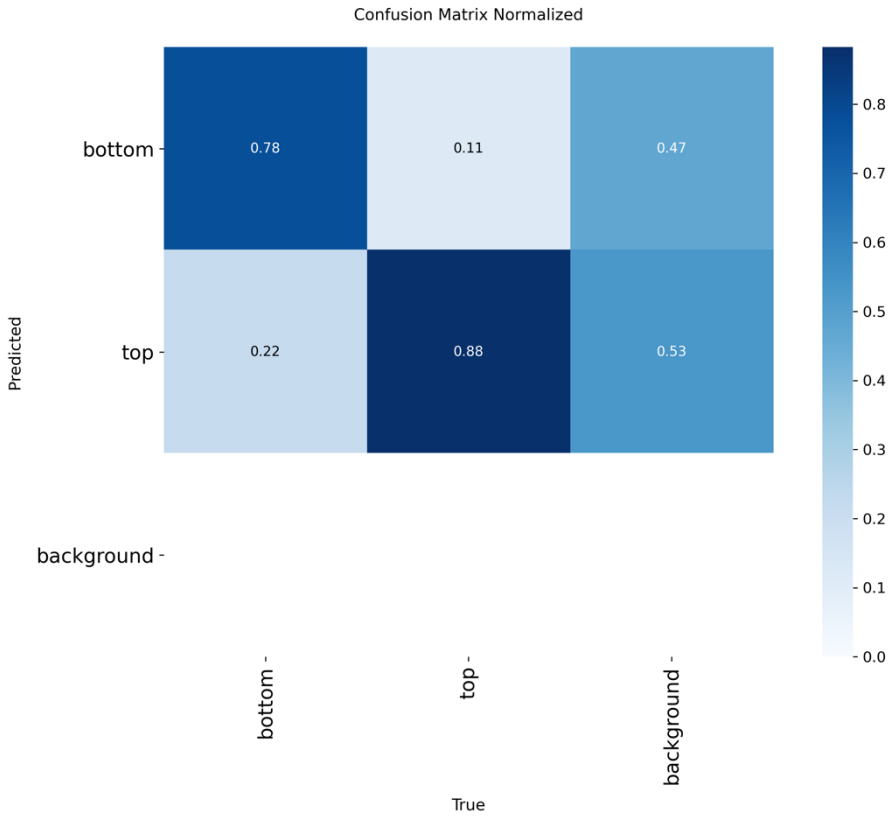

Figure 42: Normalized confusion matrix from the custom trained YOLO11 model.

To guarantee that the detections used in the prototype are accurate, the confidence score was limited to values exceeding 0.7. The confusion matrix, normalized by columns, obtained for this restriction is presented in Figure 43. The number of false positives decreased substantially, as well as the number of detections that were classified as background in the ground truth annotations. Furthermore, several detections were subsequently identified as background rather than being identified with the correct label, as expected.

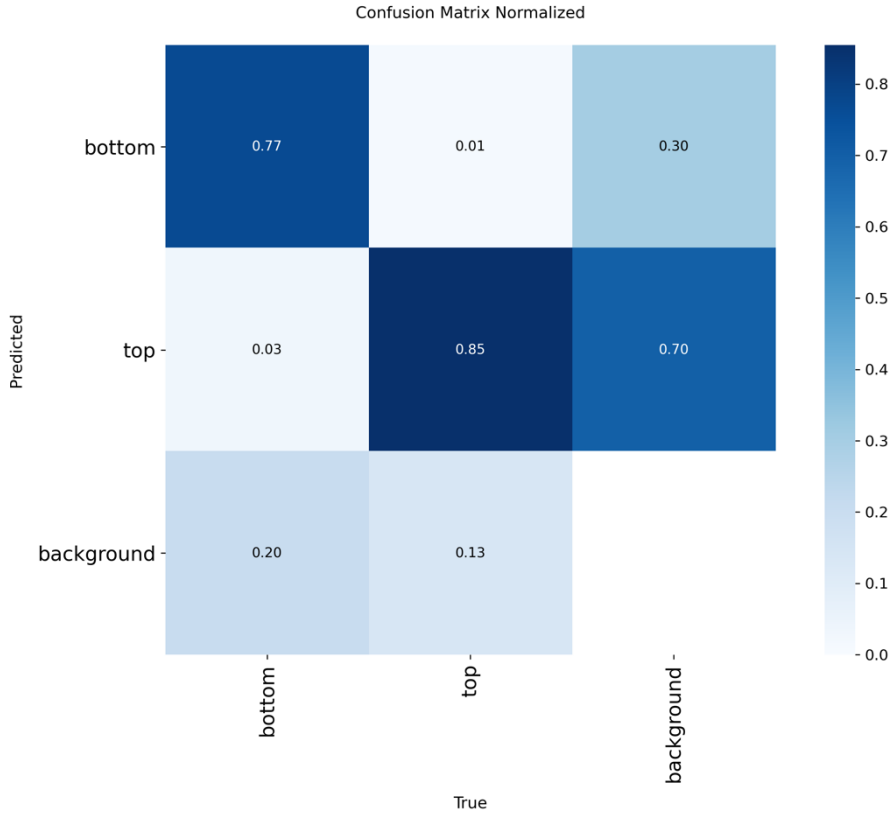

Figure 43: Normalized confusion matrix for a confidence score over 0,7.

From the data obtained from training, the precision, recall, and mean average precision were plotted, as seen in Figure 44, Figure 45 and Figure 46, respectively.

The precision curve converged to a value close to 0.9, indicating that the model learns to distinguish relevant objects in the image, reducing the number of false detections as training progresses. This is compatible with the information obtained from the confusion matrix, where the number of false positives is reduced.

The recall curve converged to 0.9, which implies that the model learned the key features of the part to identify true positives.

The mean average precision curve converged to a value slightly over 0.8, demonstrating that the model gradually learned to adjust the bounding boxes to the ground truth annotations.

Based on these metrics, it is possible to conclude that the training was effective and that the model accurately detects the parts.

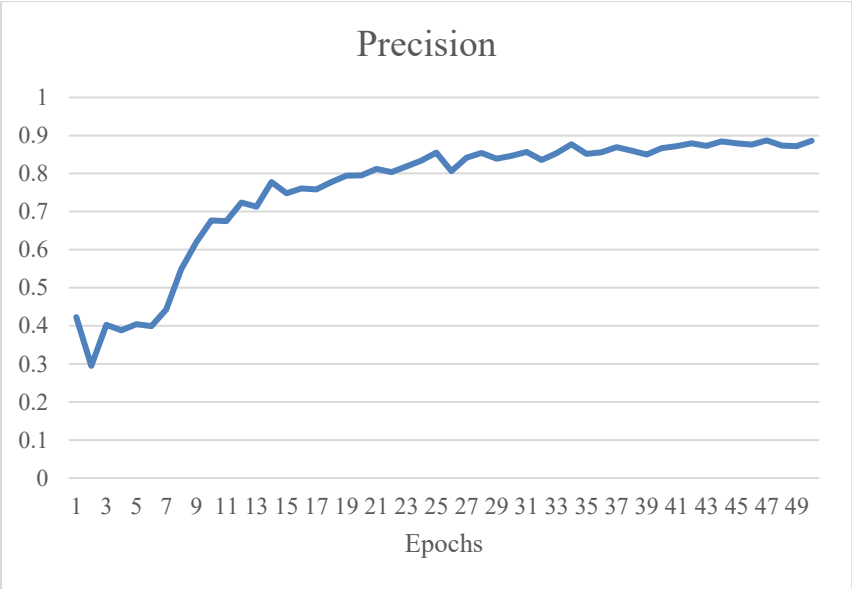

Figure 44: Model's validation precision curve.

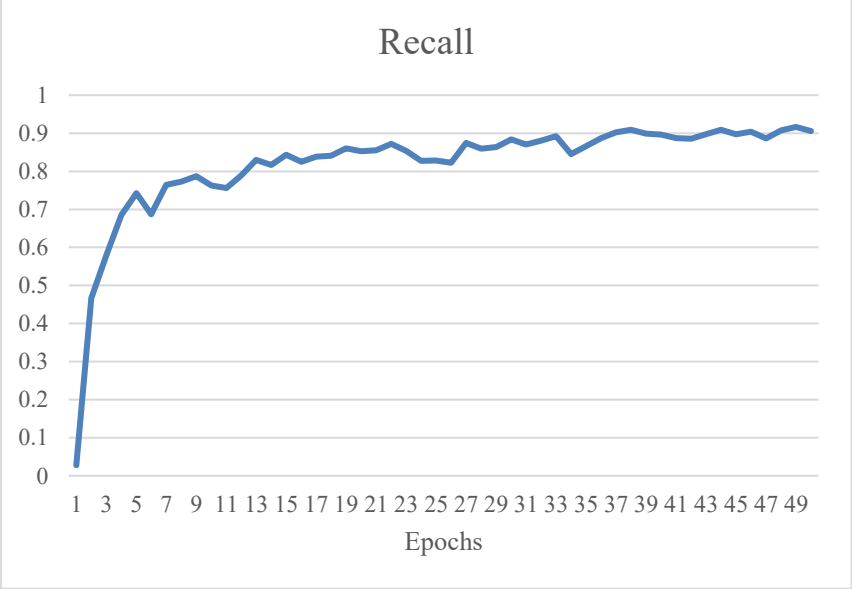

Figure 45: Model's validation recall curve.

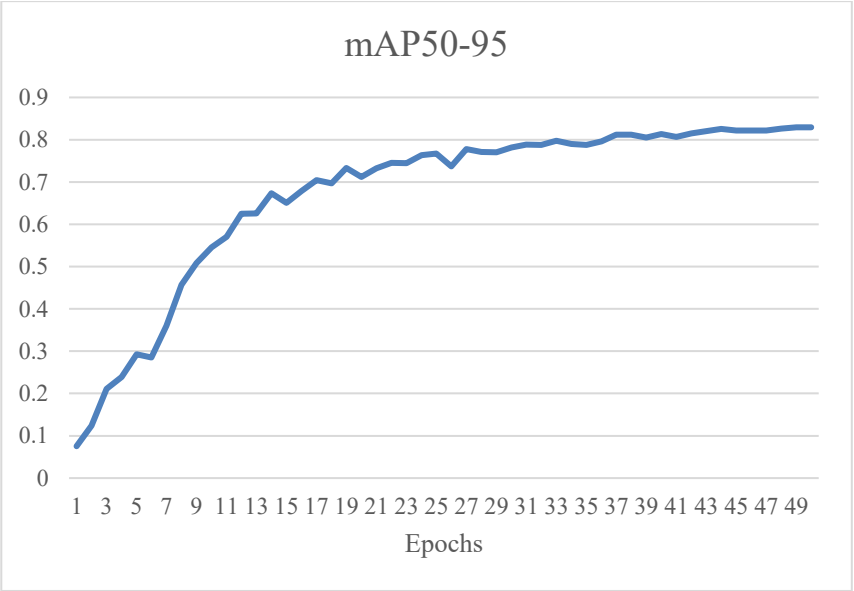

Figure 46: Model's validation mean average precision curve.

4.4.Software

RoboDK was used to program the robot and integrate all the devices used in the prototype. This software allows testing, simulation, and validation of robot programming using a digital twin, preventing collisions and reachability issues, and supporting a wide range of robots and external axes (*Robot Simulation and Programming - RoboDK*, 2025).

The first step in implementing this tool was to create a digital twin of the workstation, placing the robot and table as installed in the lab. Next, the assembly containing all the gripper parts, including the camera, the valve, and the plastic connector from the vacuum tubing, was placed at the TCP of the robot. Finally, the buffer system and the simplified CAD of the bin were placed on top of the table. The workstation's digital twin is shown in Figure 47. The placement of all the parts was extremely important to allow the use of the RoboDK collision prediction feature.

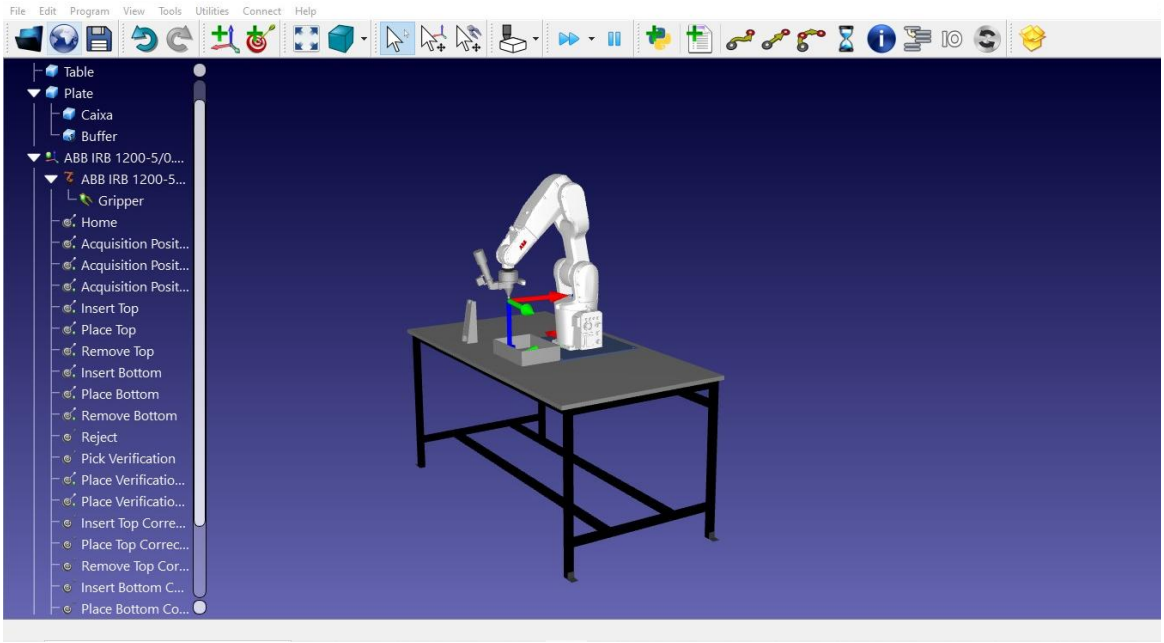

Figure 47: Digital twin of the experimental setup.

Besides the collision prediction feature of RoboDK, another significant advantage of using this software was that it is possible to program the robot through the Python API instead of the RAPID language, allowing for simple and seamless integration with the camera and the microcontroller.

The RoboDK installation comes with a Python interpreter. However, when installing the necessary libraries with pip, they were installed in the native Python installation, so it is necessary to change the Python interpreter and editor in RoboDK. By selecting “Tools” >> “Options”, a new window will appear. In this window, in the Python tab, it is possible to change the path to the Python interpreter and editor, as seen in Figure 48.

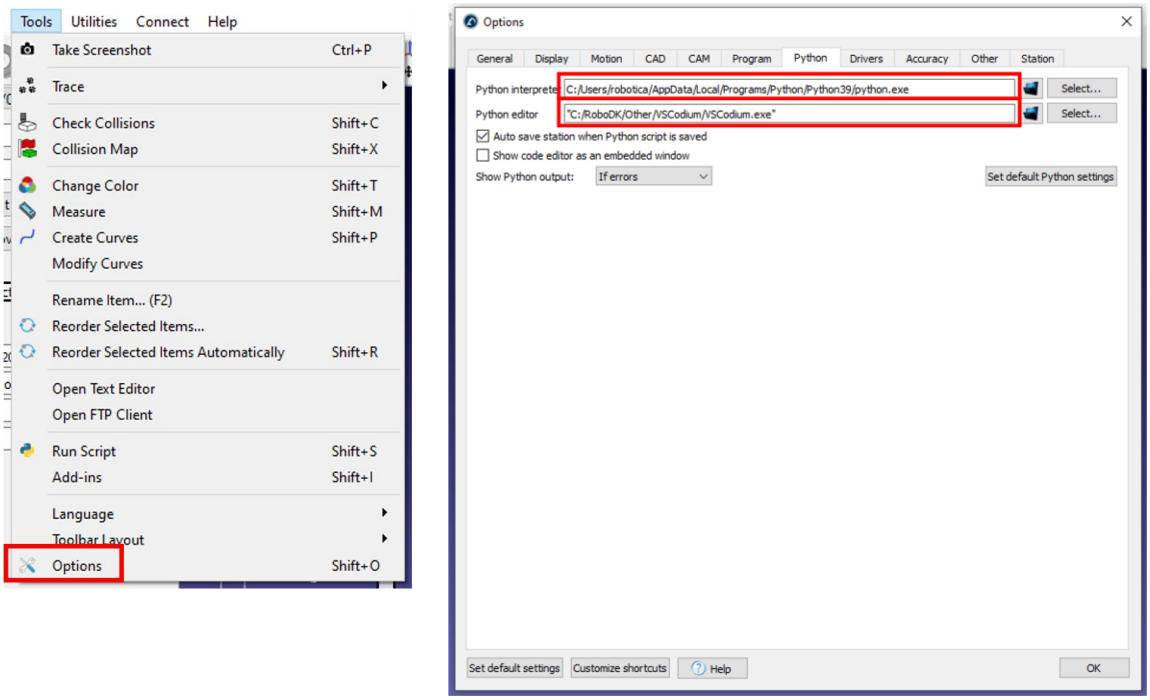

Figure 48: Python interpreter and editor configuration in RoboDK.

The additional libraries used in the development of this system are *opencv-contrib-python*, for the classic computer vision methods and image manipulation, *pyrealsense2*, to control and connect to the camera, *ultralytics*, for the implementation of the model and *pyserial* for the communication with the microcontroller.

To start the communication between the robot and RoboDK it was necessary to upload the file "RDK_DriverSocket.mod" to the robot and to configure the PC's IP to the same network as the robot. Finally, in "Connect" >> "Connect Robot" >> "More options" it was necessary to change the driver to "apiabb", Figure 49, which is the specific software to control and monitor the controller of ABB robots. To run the Python script on the robot, simply click on the Python script on the left side of the screen with the right mouse button and select "Run on Robot".

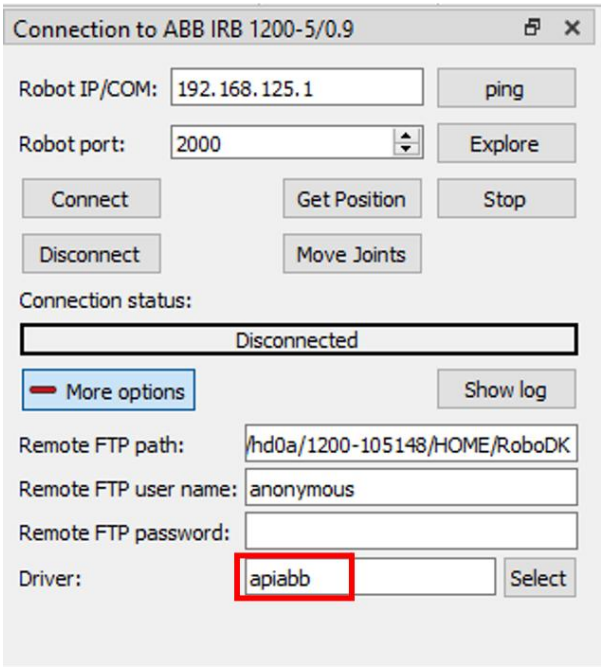

Figure 49: Configurations of the connection to the robot.

The program workflow is as presented in the following flowchart:

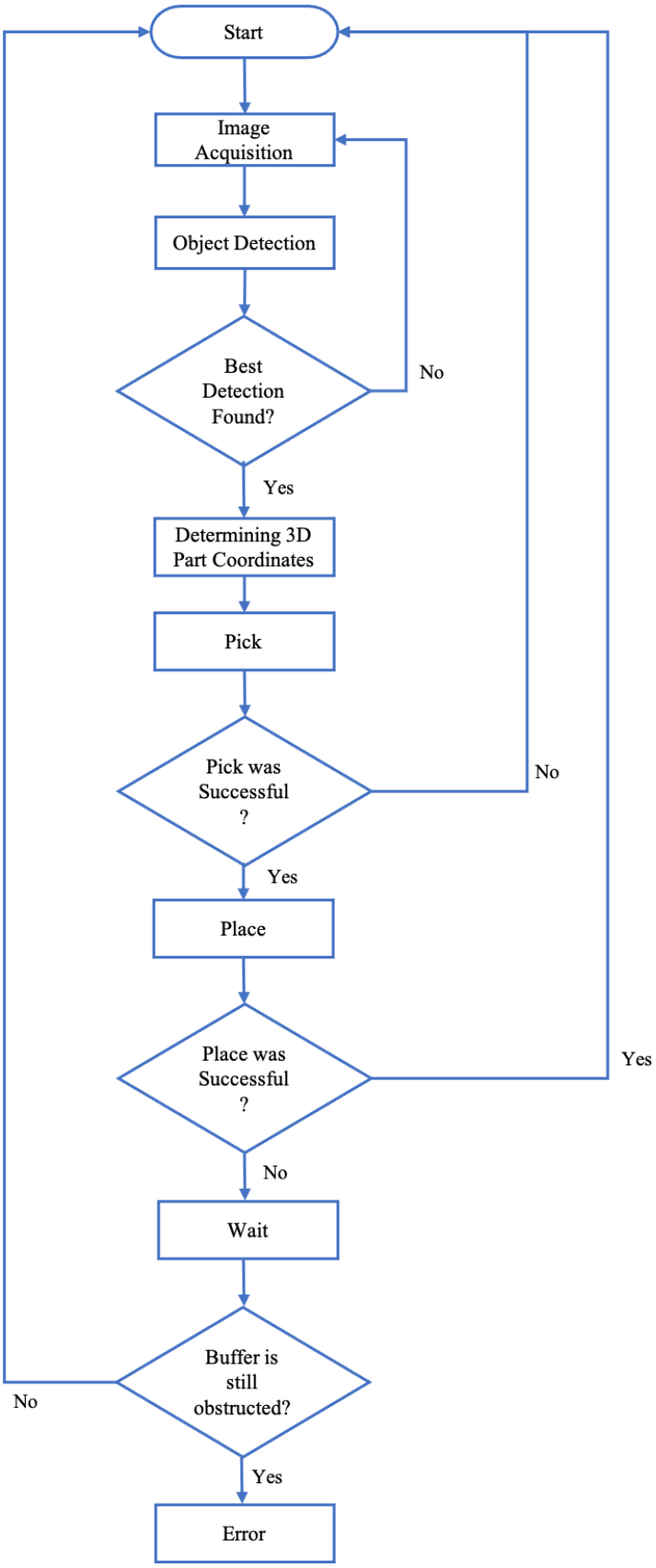

Figure 50: System workflow.

At the start of the program, the robot moves to the image-acquisition position above the bin.[1] The model is applied to the acquired image to detect the parts in the bin, and relevant information is stored for each detection: confidence score, bounding box dimensions, angle, and centre.

According to the stored information, the detections are validated: first, they are ordered by confidence score and all scores under 0.7 are rejected; then, bounding boxes that fall outside of the predefined threshold are filtered out to exclude detections where the part has a large inclination or where two parts are detected as one.

Next, the 3D coordinates of the part are determined based on the camera's intrinsic parameters, the TCP position (since the camera is attached to the robot flange) and the angle determined by the YOLO model.[2] Then, a target is created in the RoboDK environment, and the collision between the gripper and the bin is computed. If no collisions are detected, the robot moves to pick the part. On the other hand, if a collision is detected, the same procedure is repeated for the next best detection until a valid path is obtained. If no valid paths are obtained, the robot will move to a different acquisition position and resume the process from the beginning.

Once the part has been picked, the robot will move to the pick verification position, where it will acquire a new image through the mirror installed in the workstation, parallel to the table. This is why the main part was painted white, as mentioned in Chapter 4.2, with the intention of providing a contrasting background. A threshold is applied to the image, the part's contour perimeter is determined, and the perimeter is compared to a predefined range, rejecting the parts back into the bin when two or more parts are picked or restarting the cycle if no part was picked. Next, the centre of mass of the detected contour is calculated to assess the displacement of the part in the gripper. For values under 4 mm in either direction the

[1] In the actual laboratory setup, the acquisition of the image is delayed one second after the robot reaches the intended position since the table where the robot is fixed lacks stability. The vibration impairs the quality of the image and the performance of the model. This delay may be removed when the robot is installed in a sturdier industrial setting.

[2] For this step, the colour image and the depth image must be aligned using the align function from the pyrealsense2 library; otherwise, the 3D point obtained will not match the part's coordinates.

placement position is adjusted. For values over that, the part is rejected to the bin, because the buffer's opening for the suction cup allows only limited compensation of the part's placement in the horizontal direction, before colliding with the buffer.

Using the mirror for this verification is an extremely low-cost solution that delivers overall good results, with one significant disadvantage: dust accumulation on the surface, which impairs image quality over time. Incorporating a specific angle into the mirror's position may help reduce cleaning frequency. Alternatively, two other options can be employed: either incorporating a second camera, increasing the total cost of the setup, or having the robot place the part and take a new image, at the cost of an increase in cycle time.

After the initial run of tests (presented in section 5.1), another verification was implemented: the robot moves the image acquisition position above the mirror to get a view of the part's curvature, then a template matching is run with two templates, one for each side of the part. The matching mode used was the mean square difference, so the smaller value obtained from both operations was chosen as the correct side.

Subsequently, the part is placed in the buffer according to the last result, and its placement is corrected based on the displacement value obtained in the first mirror-aided test. After positioning the part, a placement verification is performed: the robot acquires an image of the buffer opening, a threshold is applied, and the number of black pixels in the image is calculated. If a part is detected in the buffer, the robot will wait for 90 seconds, one cycle of the injection machine, before outputting an error for the operator to confirm that the system can proceed.

Under the circumstances, classical methods were well suited for pick-and-place verification because the environmental conditions, such as lighting, background and part position, were stable every time and provide a very simple, low-time-consuming implementation. When similar stable environmental conditions cannot be achieved, a viable alternative is to apply the YOLO model to the pick verification image, validating the size and centre of the bounding box as previously, and replace the place verification image with a sensor that detects the presence of the part.

5. Experimental Tests

A range of tests were carried out to validate the developed prototype. The obtained results are presented and discussed in this chapter, along with improvement opportunities.

As previously indicated, an initial test was conducted to understand the overall system's performance, and some improvements were made. The background impact was assessed, and, finally, two tests were performed to evaluate the performance of the final prototype.

5.1. Initial Run – (Test A)

For the initial test, the bin was filled with 160 parts, and the test was terminated after the robot had to move acquisition positions above the bin three times, consecutively. This test was replicated twice, A1 and A2.

Figure 51: Appearance of the bin at the beginning of the test.

5.1.1. Test A - Results

The system placed a total of 137 parts in trial A1 and 136 in trial A2. By analysing Table 3, showing the average cycle time and the standard deviation per part and per batch, it can be concluded that the system complies with the aimed cycle time.

Table 3: Test A - Average cycle time and standard deviation per trial. (Both trials follow the same procedure.)

Trials	Average Cycle Time	Standard Deviation	Average Cycle Time per Batch	Standard Deviation
A1	22.46 s	9.35 s	67.38 s	16.36 s
A2	21.84 s	7.94 s	65.65 s	14.55 s

However, when reviewing Figure 52, displaying the cycle time per batch throughout the duration of the trials, the cycle time exceeded it a few times, 21 for trial A1 and 10 for trial A2. This could present a challenge. If there are no parts in the buffer or if it happens consecutively, the injection machine may have to wait for parts to become available, which is not ideal.

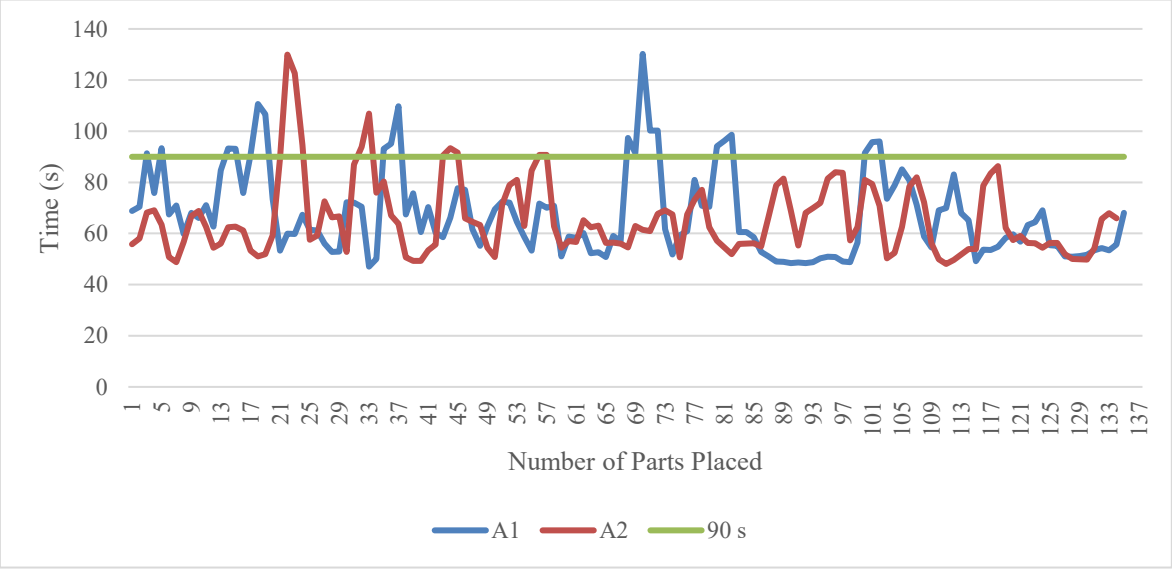

Figure 52: Test A - Cycle time per batch per trial. (Both trials follow the same procedure.)

In Figure 53, the number of attempts until a successful pick is shown. For this test, over 68% of picks were successful on the first try, and the maximum number of attempts before a successful pick was 7. As the number of attempts increases, cycle time increases, and when parts are rejected back into the bin, some may be lost around it.

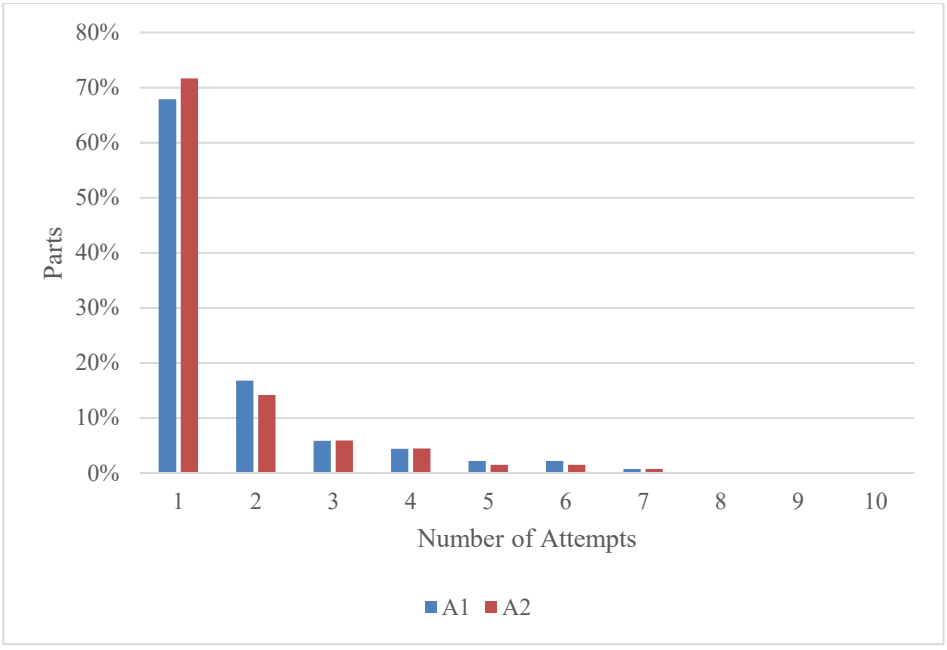

Figure 53: Test A - Attempts until successful pick. (Both trials follow the same procedure.)

The system may reject the parts in three different categories: “Displacement Too Large”, when the robot is not able to compensate for the displacement of the part in the gripper while placing the part in the buffer, “Multiple Parts Picked” and “No Part Picked”. As seen in Figure 54, over 40 % of rejected parts were due to “Displacement Too Large”.

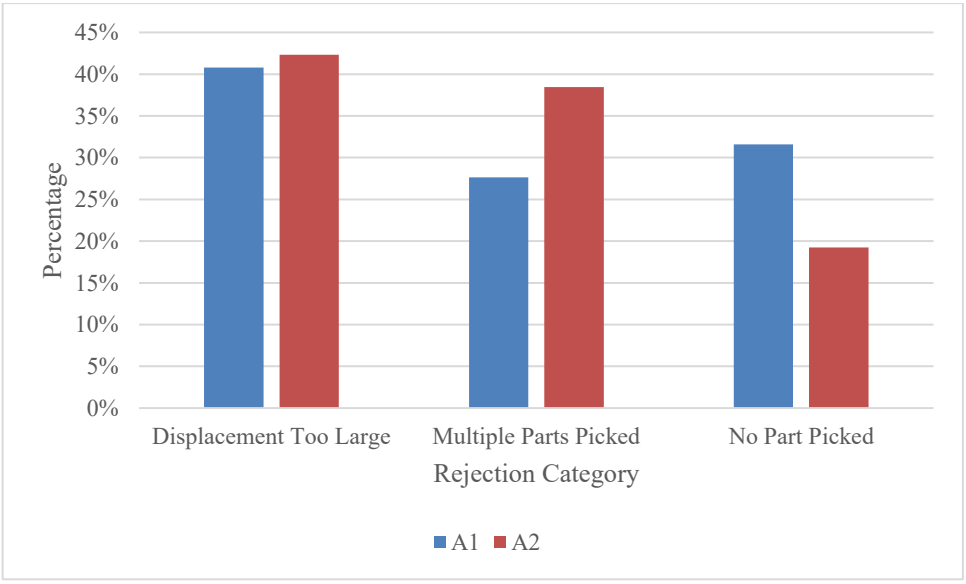

Figure 54: Test A - Rejection analysis. (Both trials follow the same procedure.)

Finally, a certain number of parts were placed on the wrong side of the buffer due to the computer vision model incorrectly identifying which side faced the camera. This was recorded for 8.0% of placed parts in trial A1 and 14.8% in trial A2.

5.1.2. Test A - Discussion

A critical concern raised by the results of this test was the elevated number of parts placed in the buffer with the wrong orientation. The buffer design works as a poka-yoke, meaning that parts placed incorrectly fall around the buffer and do not reach the injection machine's robot, preventing them from entering the injection machine in the wrong direction. However, the percentage is so high that the parts around the buffer would accumulate quickly, and the real cycle time is higher than the one reported in Table 3. This is the reason why the extra verification step presented in section 4.4 is introduced.

To reduce rejections, the conversion from camera to robot coordinates was also revised, revealing an error. The coordinates origin of the Real Sense camera is at the left imager. Initially, it was assumed that the left imager was positioned 17.5 mm from the camera's centre; however this value applies to the D435 model, whereas the system uses a D415 where the left imager is positioned 20 mm from the camera's centre (*Intel® RealSense^{TMTM} D400 Series Product Family Datasheet*, 2019).

The impact of the implemented changes is discussed in section 5.3.3.

5.2. Background Influence – (Test B)

As the system operates, variations in the bin's appearance occur; therefore, the background impact was assessed to achieve optimal performance as the number of parts in the bin decreases. Three different backgrounds were tested, light, medium and dark grey, and the bin was filled with 50 parts evenly distributed as shown in Figure 55, Figure 56 and Figure 57.

Two trials were performed for each background on consecutive days, and the termination condition of the test was set as in test A.

Figure 55: Light Grey Background for tests B1.

Figure 56: Medium Grey Background for tests B2.

Figure 57: Dark Grey Background for tests B3.

5.2.1. Test B - Results

In Figure 58, the total number of parts placed per trial is presented, and it is clear that B1 was able to place the greatest number of parts in both trials. It was also noticeable that the second trial for each test placed fewer parts than the first.

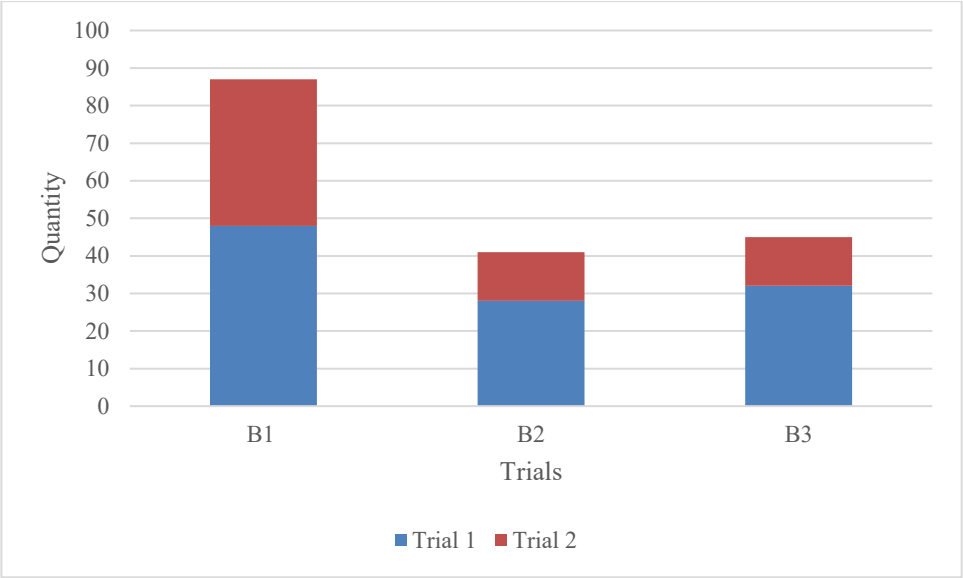

Figure 58: Test B – Total number of parts placed per trial. (B1 trials present the lighter background and B3 the darker background.)

The trials with the light grey background also presented lower average cycle time and standard deviation, as shown in Table 4.

Table 4: Test B - Average cycle time and standard deviation. (B1 trials present the lighter background and B3 the darker background.)

Trials	Average Cycle Time	Standard Deviation	Average Cycle Time per Batch	Standard Deviation
B1 Trial 1	19.47 s	4.43 s	58.70 s	7.11 s
B1 Trial 2	19.67 s	6.50 s	59.45 s	11.01 s
B2 Trial 1	20.50 s	6.52 s	62.84 s	13.44 s
B2 Trial 2	22.95 s	8.39 s	70.40 s	7.64 s
B3 Trial 1	22.16 s	7.23 s	66.72 s	12.54 s
B3 Trial 2	19.31 s	6.11 s	58.99 s	12.74 s

Analysing the attempts until successful pick, displayed in Figure 59, both trials with the light grey background successfully picked over 82% of parts at first pick, and the maximum number of attempts necessary was three. For trials B2 and B3, performance decreased; for B2, 75% of parts were successfully picked on the first try, and for B3, roughly 62%. The maximum number of attempts necessary for both B2 and B3 was four.

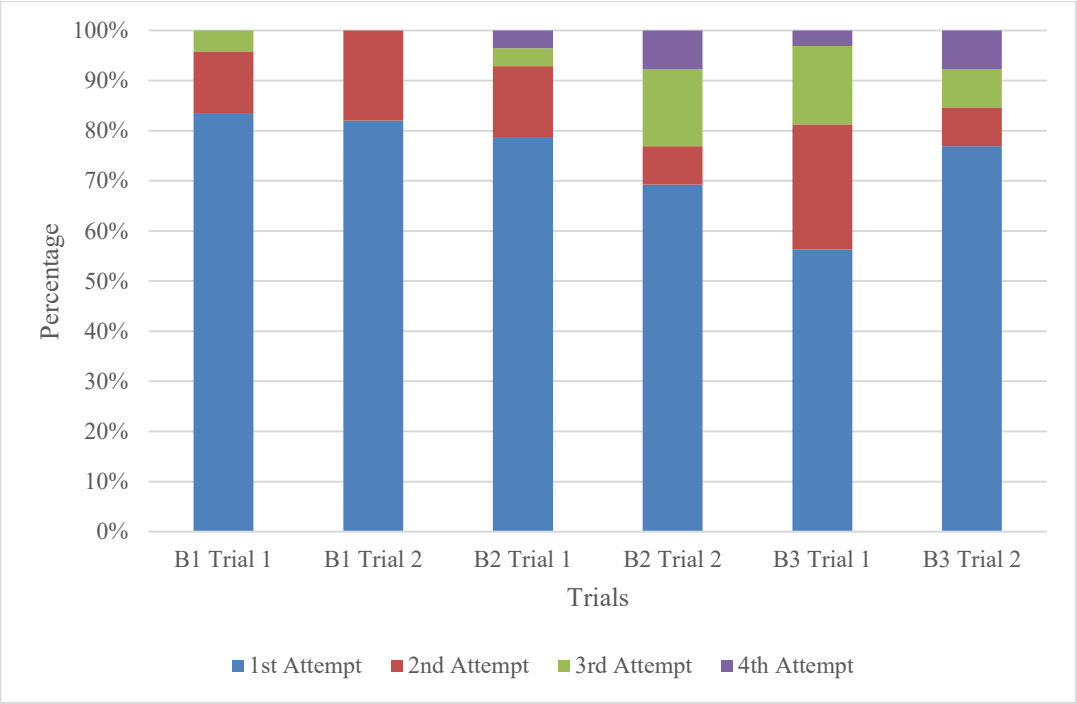

Figure 59: Test B - Attempts until successful pick per trial. (B1 trials present the lighter background and B3 the darker background.)

The median number of "good" detections per trial is presented in Figure 60. A detection is considered good if the confidence score is over 0.7. As expected, the larger number of "good" detections was obtained in trial B1, followed by B2 and finally B3. The trials on the second day achieved a higher median number of "good" detections, which can be attributed to their earlier end.

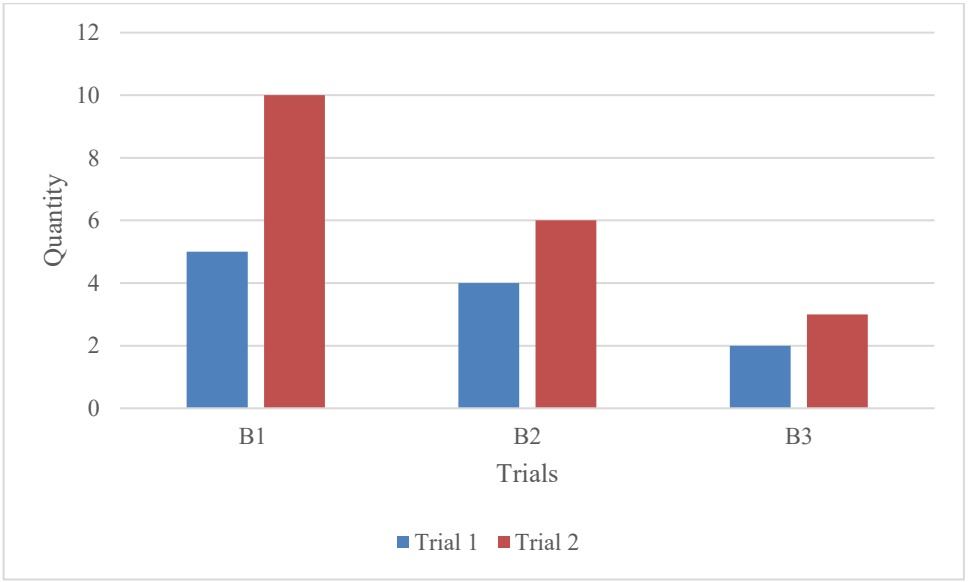

Figure 60: Test B - Median number of "good" detections. (B1 trials present the lighter background and B3 the darker background.)

5.2.2. Test C - Discussion

The light grey background performed better performance in all analysed metrics and was chosen to integrate in the final prototype. In addition, other conclusions can be drawn from the data acquired from this test, such as the model's performance being easily affected by environmental conditions and the opportunity to improve the model, namely by increasing the dataset size and adjusting the training parameters. Even with a decreased performance on the second day, the system can still perform the task at hand successfully.

5.3.Single-Fill Test – (Test C)

After the implementation of the improvements discussed in 5.1.2 and the implementation of the new background, a new test was conducted using the same methodology as test A to assess the performance of the final prototype.

5.3.1. Test C - Results

Four trials were conducted, and the total number of parts placed by the system in each trial is shown in Figure 61.

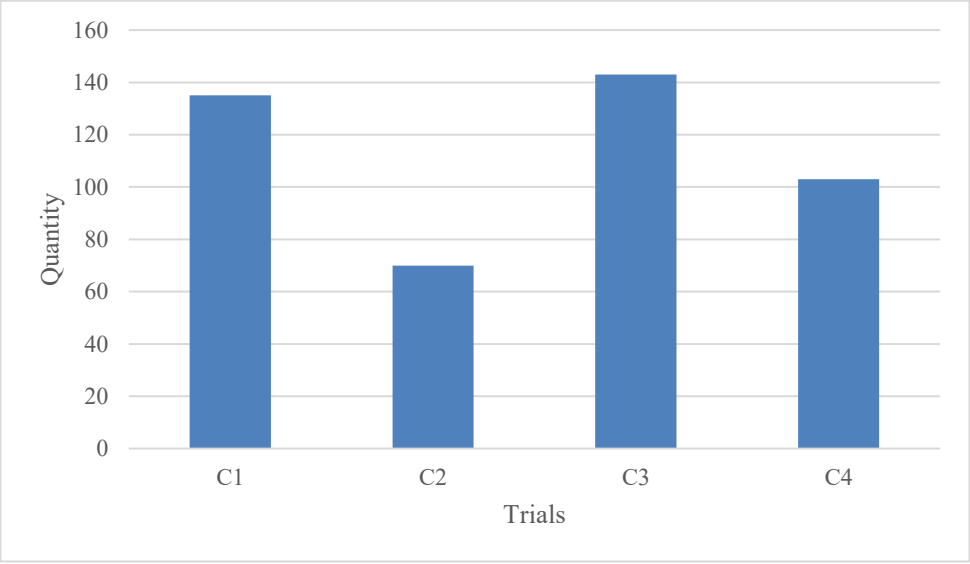

Figure 61: Test C - Total number of parts placed per trial. (All trials follow the same procedure.)

The bin's appearance at the end of each of the trials is shown in Figure 62, Figure 63, Figure 64 and Figure 65. From these images it can be observed that parts are left near the bin's walls and in piles.

Figure 62: Test C - Bin's appearance at the end of trial C1.

Figure 63: Test C - Bin's appearance at the end of trial C2.

Figure 64: Test C - Bin's appearance at the end of trial C3.

Figure 65: Test C - Bin's appearance at the end of trial C4.

The average cycle time and standard deviation per part and per batch are presented in Table 5. The values per trial show a high degree of similarity, with roughly 21.7 seconds per part and 64.5 seconds per batch, meeting the targeted cycle time.

Table 5: Test C - Average cycle time and standard deviation. (All trials follow the same procedure.)

Trials	Average Cycle Time	Standard Deviation	Average Cycle Time per Batch	Standard Deviation
C1	21.87 s	6.68 s	65.65 s	11.76 s
C2	21.71 s	6.21 s	64.77 s	9.74 s
C3	21.59 s	5.99 s	64.56 s	11.82 s
C4	21.50 s	5.57 s	64.46 s	9.19 s

By analysing the cycle time over the course of each trial, depicted in Figure 66, it is still evident that on a number of occasions, the cycle time per batch exceeds 90 seconds. The cycle time per batch is surpassed 3, 0, 10 and 1 times, in trials C1, C2, C3 and C4, respectively.

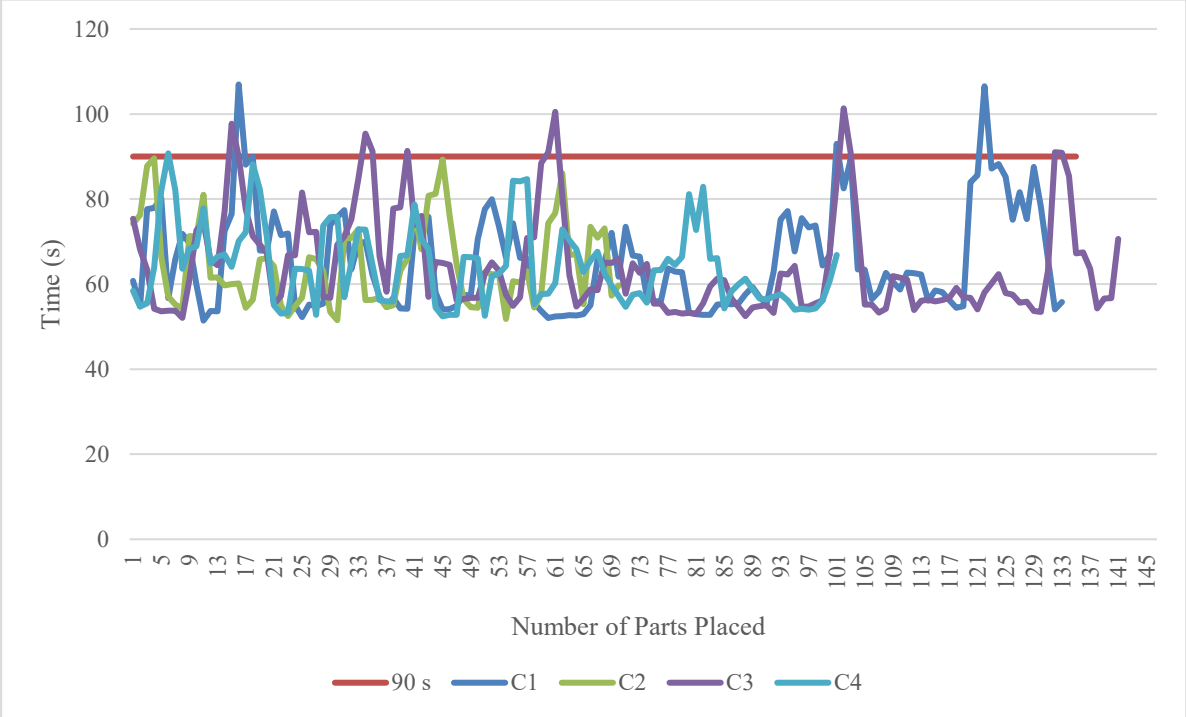

Figure 66: Test C - Cycle time per batch. (All trials follow the same procedure.)

For all tests the percentage of successful picks at first try surpasses 70%, averaging 78% and the maximum number of attempts until a successful pick was four, representing between 1 and 2% of all picks, as seen in Figure 67.

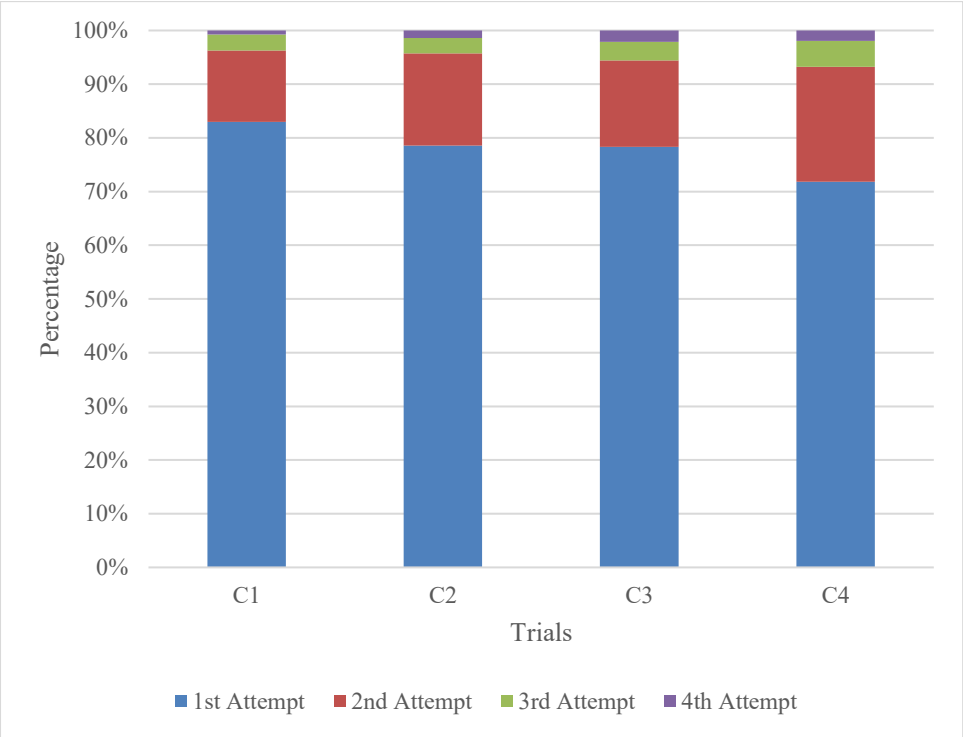

Figure 67: Test C - Attempts until successful pick per trial. (All trials follow the same procedure.)

In Figure 68, the number of rejected picks by category is presented, only slightly over 10% of picks were rejected due to the part displacement in the gripper. Still, “Multiple parts picked" is the most common cause for rejection of the picks.

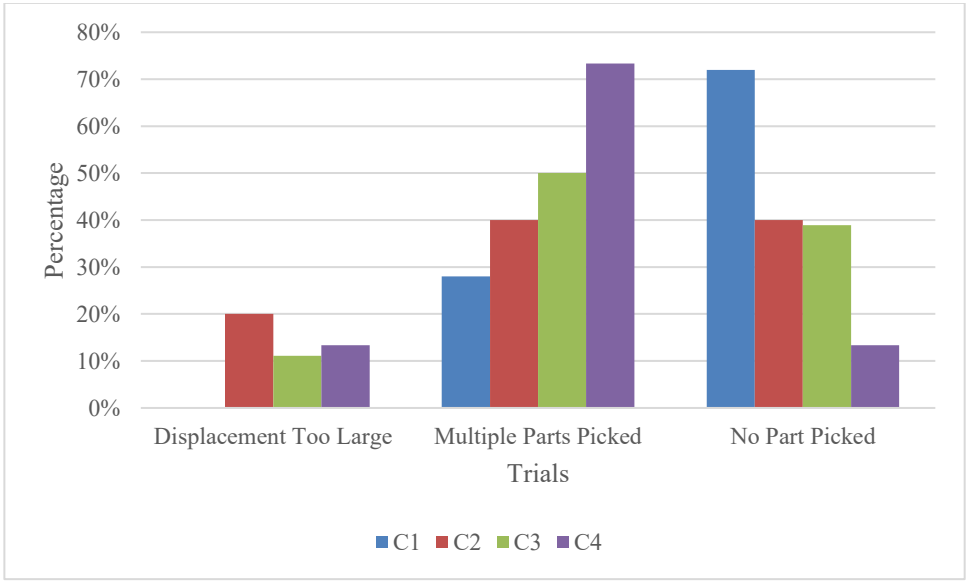

Figure 68: Test C - Rejection Analysis. All trials follow the same procedure. (All trials follow the same procedure.)

In all four trials from test C, only one part was placed on the wrong side of the buffer, for a total of 0.22% across all four trials, as seen in Table 6.

Table 6: Test C - Percentage of parts placed on the wrong side. (All trials follow the same procedure.)

Trials	Placed Wrong Side
C1	0.7%
C2	0.0 %
C3	0.0 %
C4	0.0 %

5.3.2. Comparison Test A vs Test C

The average number of parts placed in test A was 136.5, and in test C was 112.75, representing a 20% decrease, mostly driven by the very low value obtained in trial C2. The comparison between all trials is shown in Figure 69.

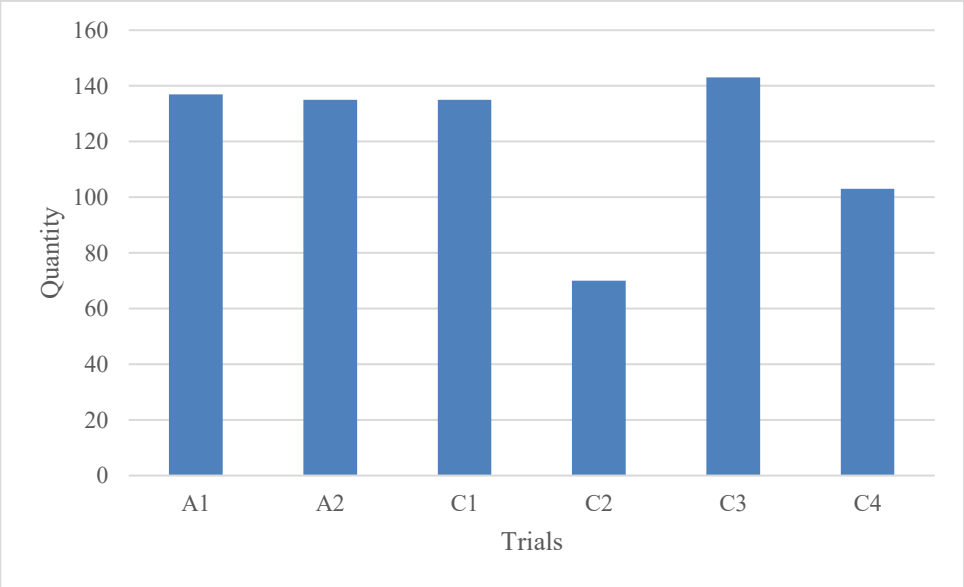

Figure 69: Comparison between total number of parts placed in test A and test C. (All trials follow the same procedure.)

In Table 7, it is possible to verify that cycle time is similar between tests A and C; however, the standard deviation decreases from test A to test C, indicating that cycle time shows less variation in test C.

Table 7: Comparison between the average cycle time between test A and C. (All trials follow the same procedure.)

Trials	Average Cycle Time	Standard Deviation	Average Cycle Time per Batch	Standard Deviation
A1	22.46 s	9.35 s	67.38 s	16.36 s
A2	21.84 s	7.94 s	65.65 s	14.55 s
C1	21.87 s	6.68 s	65.65 s	11.76 s
C2	21.59 s	6.21 s	64.77 s	9.74 s
C3	21.50 s	5.99 s	64.56 s	11.82 s
C4	21.71 s	5.57 s	64.46 s	9.19 s

A clear improvement is seen in the metric “Attempts until a successful pick”, presented in Figure 70, from a maximum number of attempts of 7 in test A to 4 in test C, and an increase in the percentage of successful picks at first try.

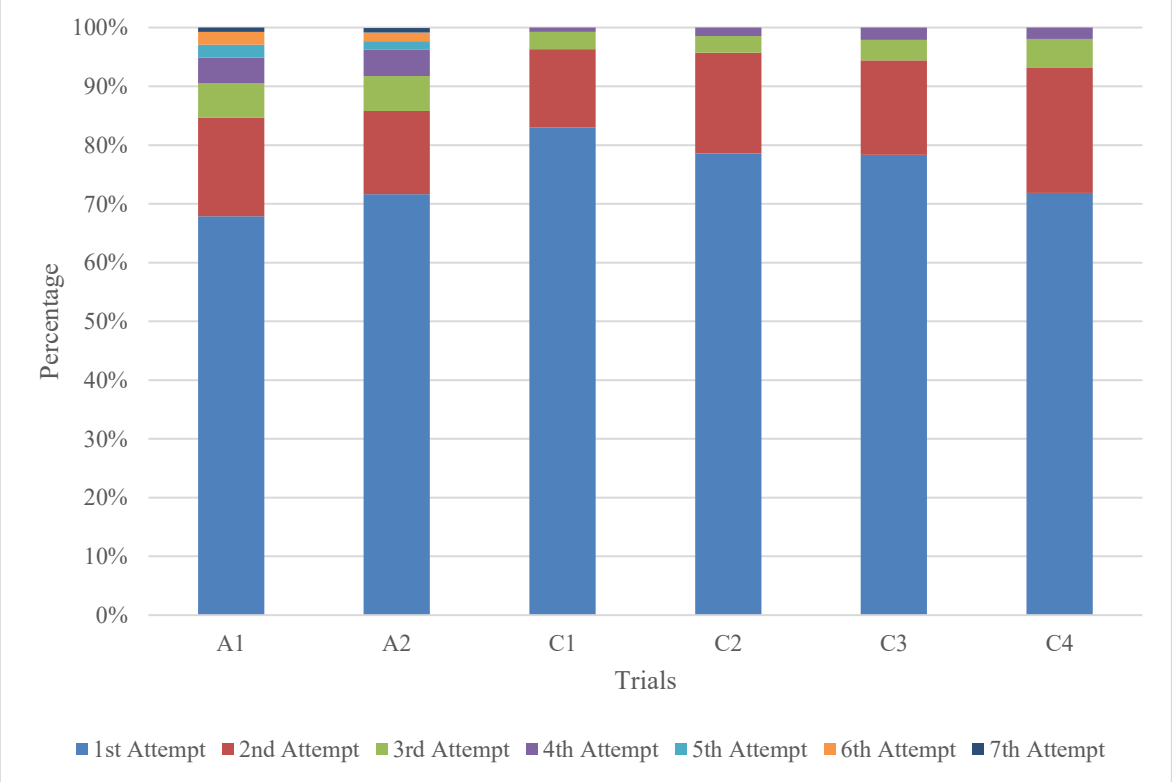

Figure 70: Comparison of attempts until successful pick between test A and C. (All trials follow the same procedure.)

Further improvement was seen in the total number of parts rejected by the displacement of the part in the gripper, as seen in Figure 71. In this graph, the height of the bar represents the percentage of rejections per trial. It is evident that the percentage of rejections reduces significantly from test A to test C, mostly due to the reduction of the rejections in the category “Displacement Too Large”.

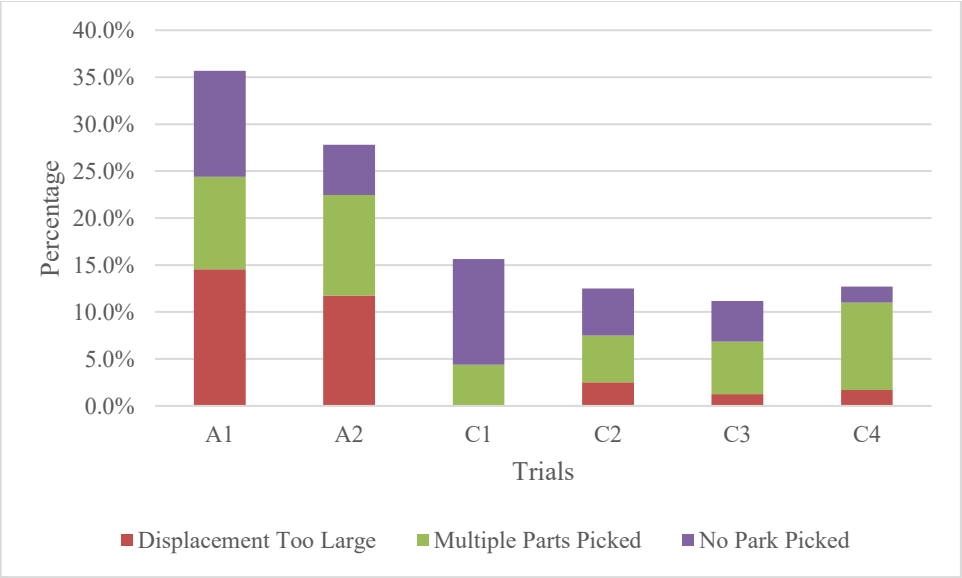

Figure 71: Comparison of the rejection reasons between tests A and C. (All trials follow the same procedure.)

Clearly, the number of parts placed on the wrong side decreased, as reflected in Table 8. The percentage of “Side Misidentified by the Model” is determined by the number of parts that were verified by the side verification step and were to not match the detection made by the model.

Table 8: Placement on the wrong side and side misidentification. (All trials follow the same procedure.)

Trials	Placement on Wrong Side	Side Misidentified by the Model
A1	8.0%	8.0%
A2	14.8%	14.8%
C1	0.7%	11.9%
C2	0.0%	5.7%
C3	0.0%	6.3%
C4	0.0%	10.7%

5.3.3. Test C - Discussion

The results presented clearly show that the changes implemented in the system were successful. The major issue encountered in the initial tests was the placement of the part in the wrong side of the buffer. With the implementation of the side verification, this number was reduced from an average of 11.4% in test A to 0.22% in test C.

Additionally, the percentage of rejected parts, and more specifically related to the displacement of the part in the gripper, was reduced by over 10%.

Furthermore, it can be concluded that the time spent on the additional verification was compensated for by the reduction in time due to fewer rejections. Even though the cycle time per batch exceeds 90 seconds at times, the number of occurrences and the consecutive ones decrease from test A to test C.

The most significant issue observed in test C was the variability in the total number of parts picked. Analysing the bin's appearance at the end of each trial shows that parts are consistently left near the bin walls and accumulate in smaller piles. A likely cause is the reduced visibility of the parts located near the bin walls and the pose of the parts making them impossible to pick.

Additionally, the increase in contrast as the bin was progressively emptied may contribute to an increased difficulty from the model to distinguish between parts and for piles of parts to remain in the bin. The lighter background presented good results, as seen in section 5.2; however, for the tests, the bin was manually filled and the parts were evenly distributed. Considering this observation, a bin emptying strategy should be implemented to improve the total number of parts that the system can pick. This strategy should ensure that the robot will not continuously pick from the same region. The implementation of a vibrating table can also aid in maintaining the parts uniformly distributed in the bin.

It was noted that a few parts per trial were slightly misplaced in the buffer, around 2.88%. The most probable causes are: significant differences between parts, mostly in part's thickness, and a large friction in the buffer. These issues should be solved in the industrial application once the supplied parts are mass produced, presenting less variation, and the buffer is produced in a metallic material.

5.4. Density of Parts Impact – (Test D)

Finally, the system's behaviour in three different stable part densities within the bin was assessed. For trials D1, D2 and D3, the bin was filled with 40, 100 and 160 parts, respectively and resupplied in the same sequence in all three trials. The test was terminated once the system had placed 100 parts; in none of the trials did the system move the acquisition positions above the bin more than three consecutive times.

5.4.1. Test D - Results

In

Table 9, the average cycle time and standard deviation per part and per batch are presented. All trials present similar average cycle time per part and per batch, however trial D3 presents a higher standard deviation.

Table 9: Test D - Average cycle time and standard deviation. (D1-40 parts; D2-100 parts; D3-160 parts).

Trials	Average Cycle Time	Standard Deviation	Average Cycle Time per Batch	Standard Deviation
D1	20.27 s	4.60 s	60.96 s	7.34 s
D2	20.32 s	4.58 s	60.87 s	7.59 s
D3	20.23 s	5.90 s	60.83 s	11.31 s

When analysing the cycle time per batch throughout the duration of the trials, as presented in Figure 72, it is evident that D1 and D2 never exceed the target cycle time of 90 seconds, whereas D3 exceeds it 5 times.

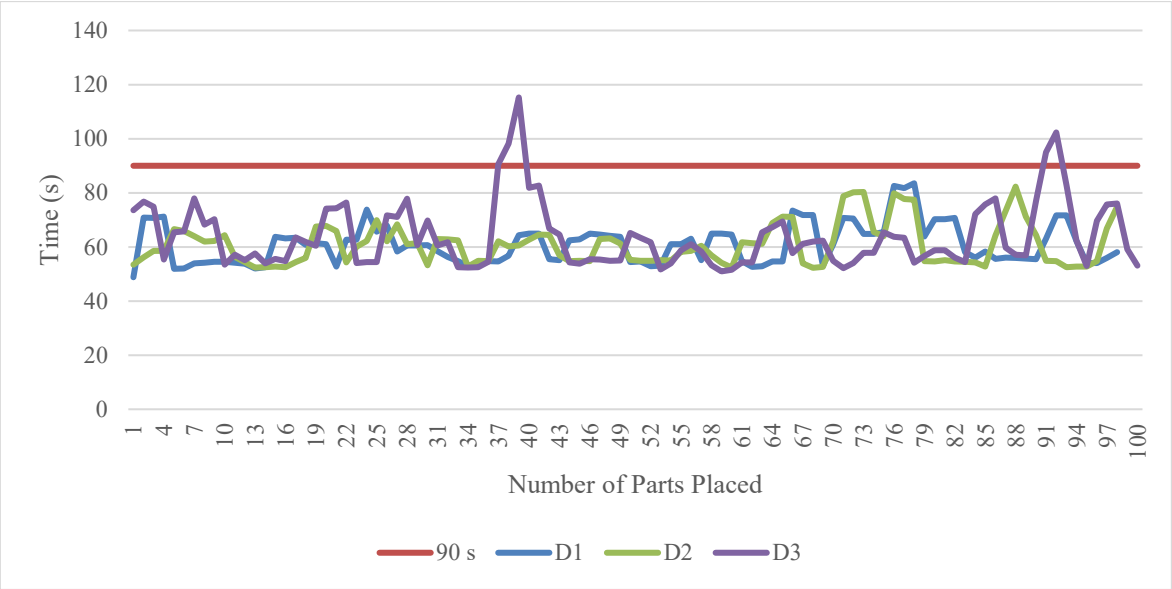

Figure 72: Test D - Cycle time per batch. (D1-40 parts; D2-100 parts; D3-160 parts).

The number of attempts until a successful pick per trial is shown in Figure 73. In trials D1 and D2, over 87% of parts were successfully picked at the first attempt, while in trial D3, that number decreases to 72%. The maximum number of attempts for trial D1 was three, for D2 was two and for D3 five, the highest of the trials.

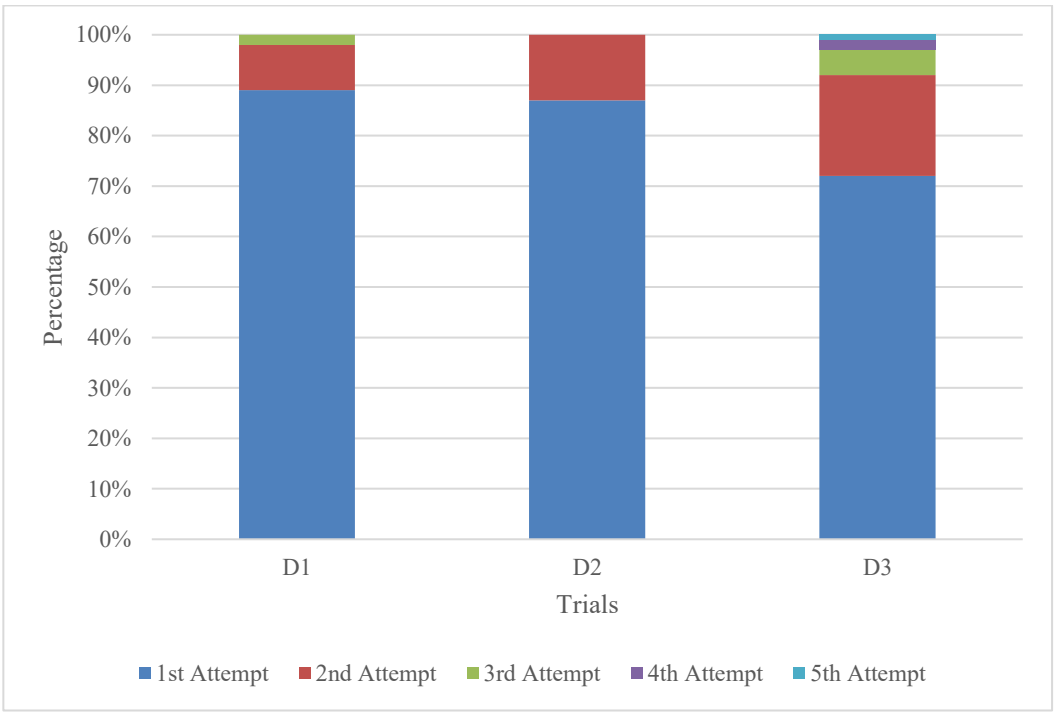

Figure 73: Test D - Attempts until successful pick. (D1-40 parts; D2-100 parts; D3-160 parts).

Looking at the median number of detections and “good” detections for each trial, in Figure 74, it is possible to see that trial D2 presented a higher median for detections. However, trial D1 presented a higher median of “good” detections.

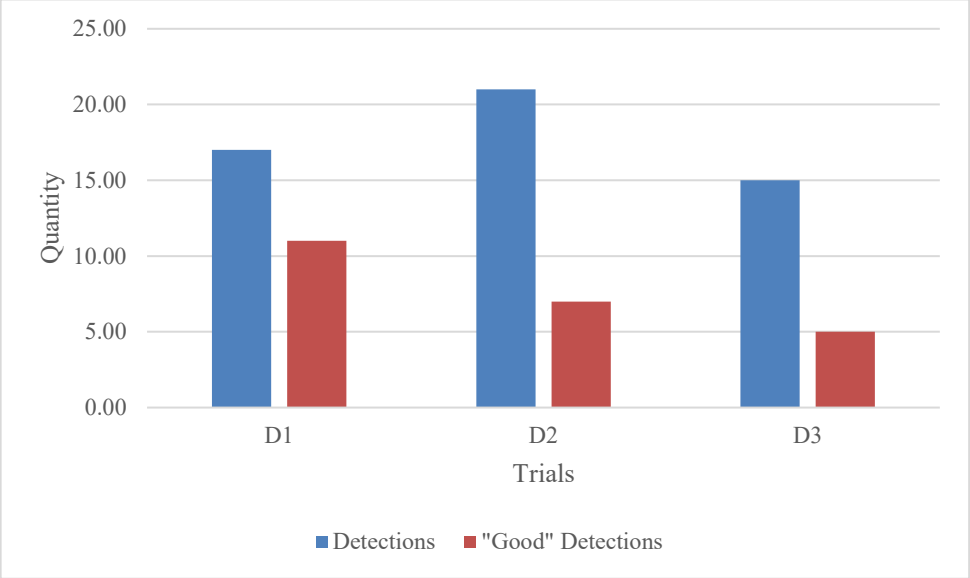

Figure 74: Test D - Median detections and "good" detections per trial. (D1-40 parts; D2-100 parts; D3-160 parts).

The most common rejection cause for trial D1 was “No Part Picked”, and for trials D2 and D3 was “Multiple Parts Picked”, as seen in Figure 75.

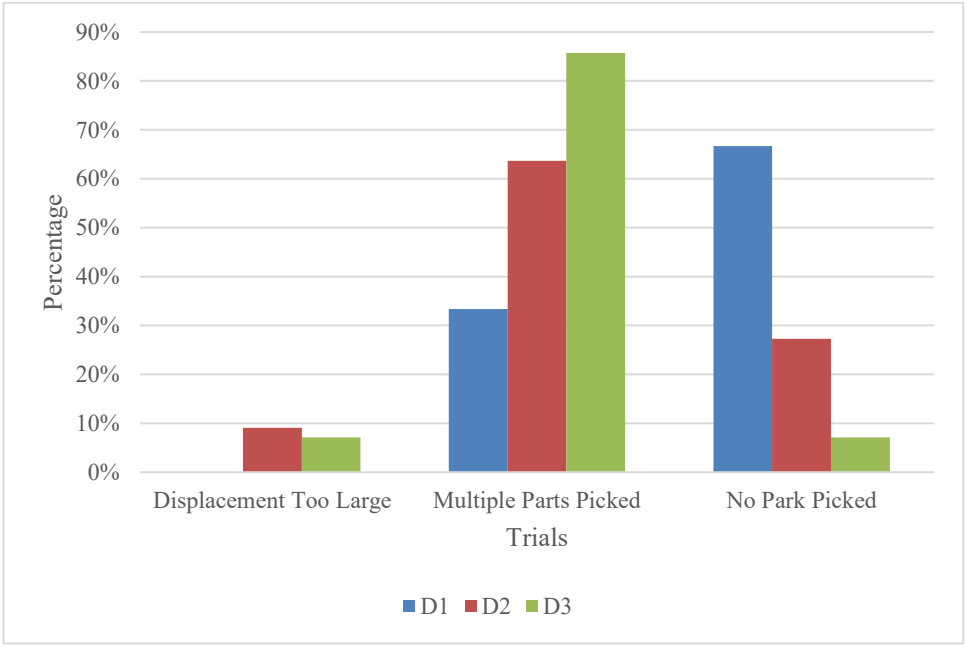

Figure 75: Test D - Rejection analysis. (D1-40 parts; D2-100 parts; D3-160 parts).

In Table 10, an increase in the percentage of parts misidentified by the model is observed as the bin's density of parts increases. Despite this, the total percentage of parts placed incorrectly in test D was 0,67%.

Table 10: Percentage of parts placed on the wrong side of the buffer and misidentified by the model. (D1-40 parts; D2-100 parts; D3-160 parts).

Trials	Placement on Wrong Side	Side Misidentified by the Model
D1	1.0%	7.0%
D2	0.0%	11.0%
D3	1.0%	20.0%

5.4.2. Discussion

According to the analysed outcomes, the test with the worst performance was D3, the bin with the highest part density. The 90-second cycle time per batch was exceeded several times throughout the length of the trial, the number of attempts required until a successful pick and the number of side misidentifications was higher.

The most frequent reason for rejection in test D3 was “Multiple Parts Picked”, due to the minimal part’s weight and the bin being more cluttered.

It is possible to conclude that the average cycle time and standard deviation per part and per batch improve for all trials when comparing with test C, reducing the average cycle time per part by approximately 1 second and the cycle time per batch by approximately 4 seconds.

Additionally, the system presented strong performance in trial D1 where the density of parts in the bin is lower, further supporting the earlier claim regarding the variation in the total number of parts placed in test C.

6. Conclusions

The system developed in this project successfully solves the issue of selecting, picking, orienting, and placing parts from an unorganised setting into an organised, predictable configuration, reducing the strain on the workers who, at this moment, perform this task manually.

The initial stage of this project focused on the computer vision techniques most suited to identify the parts inside the bin, acknowledging that these parts may appear to the camera in different poses. Taking into account existing solutions, this system handles a significantly different part from the majority, which predominantly focus on industrial metallic components or everyday objects. The material's colour posed a real challenge for the vision technology, due to working close to the camera's saturation limit and significant contrast changes in the image as the bin is progressively emptied.

Classical computer vision methods were considered unsuited to adapt to these significant changes in the bin's appearance and the part's pose. On the other hand, these showed good results in the verification stages, since the part's pose and background conditions were much more stable.

A deep learning model was implemented in the system due to its robustness and adaptability. Even though the results showed that the model’s performance is sensitive to environment conditions and changes in the bin’s appearance, the task was still performed within the required timeframe and with precise placement of the parts. One of the bigger disadvantages of using a deep learning model was the time consumption on the development of the dataset (acquiring and annotating a large number of images with varying number of parts and backgrounds).

The next stage of this project focused on developing a functional prototype. From the manufacturing of parts to the development of the gripper, butterfly valve, customised suction cup and buffer technology, as well as the integration of the interaction of all the components through RoboDK.

The proposed solution seeks to minimize cost and prioritize flexibility and adaptability. Two initial investments are necessary: a 6DOF robotic manipulator and a RoboDK permanent

license. RoboDK presents a significant advantage, allowing for anyone to program a robot with no prior knowledge of the RAPID programming language, essential to the technology hand over to the partnering company.

The installation of the camera on the robot's end-effector allowed for increased flexibility of the system, easily adapting to changes in the workstation, such as:

- Moving the bin and buffer into different places according to the available space in the factory
- Implementing of additional features, as in the pick verification steps, without having to recalculate the transformation between the camera's position and the robot.

The pick verifications presented a crucial role to ensure the precise placement of the parts in the buffer for the following processing steps.

Finally, several tests were performed with positive results to validate the developed system. The results showed that the system complies with the aimed cycle time and precise positioning of the parts as intended. The optimal working conditions of the prototype were identified as those in which the parts are evenly distributed in the bin and the bin is not excessively full.

Although the proposed prototype provides an effective solution to this case study, further refinements are possible. Some alternatives to a few of the decisions made were already presented in the report, such as using a second camera, replacing the mirror for pick and side verifications; and using sensors to detect parts in the buffer.

Optimization can be attained by implementing:

- A smaller suction cup, reducing the number of times multiple parts are picked;
- An industrial valve, that completely shuts off the airflow through the gripper, preventing parts from staying attached to the gripper after rejection;
- A buffer in frictionless material, preventing the misplacement of parts and jams in the buffer;
- An improved dataset and refinement of training parameters, improving the robustness of the model, reducing or eliminating the number of misidentifications, allowing for the removal of the side verification step and reducing cycle time;

- A bin emptying strategy, picking parts evenly from the bin or a vibrating table under the bin ensuring a stable bin appearance to the camera, improving the overall performance;
- A stable and robust workstation, allowing for an increase in the acceleration and a decrease in the time necessary before acquiring a new image, reducing cycle time.

The work developed in this report resulted in the publication of an article for the ICARSC 2026 conference and submission of one IEEE journal article.

Bibliography

Accuracy: Machine Learning Metric | Ultralytics. (2025). https://www.ultralytics.com/glossary/accuracy

Araki, R., Onishi, T., Hirakawa, T., Yamashita, T., & Fujiyoshi, H. (2020). MT-DSSD: Deconvolutional Single Shot Detector Using Multi Task Learning for Object Detection, Segmentation, and Grasping Detection. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 10487–10493. https://doi.org/10.1109/ICRA40945.2020.9197251

Batch Size in Deep Learning | Ultralytics. (2025). https://www.ultralytics.com/glossary/batch-size

Buchholz, D., Futterlieb, M., Winkelbach, S., & Wahl, F. M. (2013). Efficient bin-picking and grasp planning based on depth data. *2013 IEEE International Conference on Robotics and Automation*, 3245–3250. https://doi.org/10.1109/ICRA.2013.6631029

Buchholz, D., Kubus, D., Weidauer, I., Scholz, A., & Wahl, F. M. (2014). Combining visual and inertial features for efficient grasping and bin-picking. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 875–882. https://doi.org/10.1109/ICRA.2014.6906957

Chang, W.-C., & Wu, C.-H. (2016). Eye-in-hand vision-based robotic bin-picking with active laser projection. *The International Journal of Advanced Manufacturing Technology*, *85*(9), 2873–2885. https://doi.org/10.1007/s00170-015-8120-0

Cordeiro, A., Rocha, L. F., Costa, C., Costa, P., & Silva, M. F. (2022). Bin Picking Approaches Based on Deep Learning Techniques: A State-of-the-Art Survey. *2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 110–117. https://doi.org/10.1109/ICARSC55462.2022.9784795

Datasets. (2021). https://graspnet.net/datasets.html

Domae, Y., Okuda, H., Taguchi, Y., Sumi, K., & Hirai, T. (2014). Fast graspability evaluation on single depth maps for bin picking with general grippers. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 1997–2004. https://doi.org/10.1109/ICRA.2014.6907124

Force Torque Sensor—Robot Force Sensors | Precise FT Sensor Control. (2025). Fanucamerica. https://www.fanucamerica.com/products/robots/robot-options/force-torque-sensor

Force/torque sensors. (2025). https://schunk.com/de/en/automation-technology/force/torque-sensors/c/PUB_11579

Fujita, M., Domae, Y., Noda, A., Garcia Ricardez, G. A., Nagatani, T., Zeng, A., Song, S., Rodriguez, A., Causo, A., Chen, I. M., & Ogasawara, T. (2019). What are the important technologies for bin picking? Technology analysis of robots in competitions based on a set of performance metrics. *Advanced Robotics*, 1–15. https://doi.org/10.1080/01691864.2019.1698463

Ghosh, A. (2024, October 8). *YOLO11: Redefining Real-Time Object Detection - Tutorial*. https://learnopencv.com/yolo11/

Gou, M., Fang, H.-S., Zhu, Z., Xu, S., Wang, C., & Lu, C. (2021). *RGB Matters: Learning 7-DoF Grasp Poses on Monocular RGBD Images* (arXiv:2103.02184). arXiv. https://doi.org/10.48550/arXiv.2103.02184

He, Z., Feng, W., Zhao, X., & Lv, Y. (2021). 6D Pose Estimation of Objects: Recent Technologies and Challenges. *Applied Sciences*, *11*(1), Article 1. https://doi.org/10.3390/app11010228

Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., & Lepetit, V. (2011). Multimodal templates for real-time detection of texture-less objects in

heavily cluttered scenes. *2011 International Conference on Computer Vision*, 858–865. https://doi.org/10.1109/ICCV.2011.6126326

Industrial 3D cameras for robotic automation—Zivid. (2025). https://www.zivid.com

Intel® RealSense™ D400 Series Product Family Datasheet. (2019). Intel. https://www.intel.com/content/www/us/en/content-details/841984/intel-realsense-d400-series-product-family-datasheet.html

Iriondo, A., Lazkano, E., & Ander Ansuategi, A. A. (2021). *Affordance-Based Grasping Point Detection Using Graph Convolutional Networks for Industrial Bin-Picking Applications*. https://www.mdpi.com/1424-8220/21/3/816

Jiang, P., Ishihara, Y., Sugiyama, N., Oaki, J., Tokura, S., Sugahara, A., & Ogawa, A. (2020). Depth Image–Based Deep Learning of Grasp Planning for Textureless Planar-Faced Objects in Vision-Guided Robotic Bin-Picking. *Sensors*, *20*(3). https://doi.org/10.3390/s20030706

Jiang, P., Oaki, J., Ishihara, Y., Ooga, J., Han, H., Sugahara, A., Tokura, S., Eto, H., Komoda, K., & Ogawa, A. (2022). Learning Suction Graspability Considering Grasp Quality and Robot Reachability for Bin-Picking. *Frontiers in Neurorobotics*, *16*, 806898. https://doi.org/10.3389/fnbot.2022.806898

Kleeberger, K., Bormann, R., Kraus, W., & Huber, M. F. (2020). A Survey on Learning-Based Robotic Grasping. *Current Robotics Reports*, *1*(4), 239–249. https://doi.org/10.1007/s43154-020-00021-6

Kumar, V. (2024). *Robot Gripper Technologies For Stiff Fabric Pads*. Instituto Politécnico de Leiria.

Le, T.-T., & Lin, C.-Y. (2019). Bin-Picking for Planar Objects Based on a Deep Learning Network: A Case Study of USB Packs. *Sensors*, *19*(16), 3602. https://doi.org/10.3390/s19163602

Li, G., Karlsen, B., Ytrøy, V. H., Mork, O. J., & Zhang, H. (2023). Design of Digital Planner and 3D Vision System for Robot Bin Picking. *2023 International Conference on Advanced Robotics and Mechatronics (ICARM)*, 476–481. https://doi.org/10.1109/ICARM58088.2023.10218895

Li, X., Cao, R., Feng, Y., Chen, K., Yang, B., Fu, C.-W., Li, Y., Dou, Q., Liu, Y.-H., & Heng, P.-A. (2022). A Sim-to-Real Object Recognition and Localization Framework for Industrial Robotic Bin Picking. *IEEE Robotics and Automation Letters*, *7*(2), 3961–3968. IEEE Robotics and Automation Letters. https://doi.org/10.1109/LRA.2022.3149026

Mahony, N. O., Campbell, S., Carvalho, A., Harapanahalli, S., Velasco-Hernandez, G., Krpalkova, L., Riordan, D., & Walsh, J. (2020). *Deep Learning vs. Traditional Computer Vision* (Vol. 943). https://doi.org/10.1007/978-3-030-17795-9

Martinez, C., Boca, R., Zhang, B., Chen, H., & Nidamarthi, S. (2015). Automated bin picking system for randomly located industrial parts. *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, 1–6. https://doi.org/10.1109/TePRA.2015.7219656

Model Training with Ultralytics YOLO - Ultralytics YOLO Docs. (2025). https://docs.ultralytics.com/modes/train/

Oh, J.-K., Lee, S., & Lee, C.-H. (2012). Stereo vision based automation for a bin-picking solution. *International Journal of Control, Automation and Systems*, *10*(2), 362–373. https://doi.org/10.1007/s12555-012-0216-9

One Source. Unlimited Vision. | Teledyne Vision Solutions. (2025). https://www.teledynevisionsolutions.com/

OpenCV: Canny Edge Detection. (2025). https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html

OpenCV: Contours: Getting Started. (2025). https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html

OpenCV: Eroding and Dilating. (2025). https://docs.opencv.org/3.4/db/df6/tutorial_erosion_dilatation.html

OpenCV: Image Thresholding. (2025). https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html

OpenCV: Template Matching. (2025). https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html

Oriented Bounding Box (OBB) Datasets Overview—Ultralytics YOLO Docs. (2025). https://docs.ultralytics.com/datasets/obb/

Park, J., Jun, M. B. G., & Yun, H. (2022). Development of robotic bin picking platform with cluttered objects using human guidance and convolutional neural network (CNN). *Journal of Manufacturing Systems*, *63*, 539–549. https://doi.org/10.1016/j.jmsy.2022.05.011

Pick-place |. (2025). https://pick-place.eu/

Pochyly, A., Kubela, T., Singule, V., & Cihak, P. (2012). 3D vision systems for industrial bin-picking applications. *Proceedings of 15th International Conference MECHATRONIKA*, 1–6. https://ieeexplore.ieee.org/document/6415038/?arnumber=6415038

Pretto, A., Tonello, S., & Menegatti, E. (2013). Flexible 3D localization of planar objects for industrial bin-picking with monocamera vision system. *2013 IEEE International Conference on Automation Science and Engineering (CASE)*, 168–175. https://doi.org/10.1109/CoASE.2013.6654067

Raj, P., Behera, L., & Sandhan, T. (2024). Scalable and Time-Efficient Bin-Picking for Unknown Objects in Dense Clutter. *IEEE Transactions on Automation Science and Engineering*, *21*(3), 2289–2301. https://doi.org/10.1109/TASE.2023.3324610

RealSense. (2025). RealSense. https://www.realsenseai.com/

Roboflow: Computer vision tools for developers and enterprises. (2025). https://roboflow.com

Robot Simulation and Programming—RoboDK. (2025). https://robodk.com/simulation

Sun, H., Zhang, Z., Wang, H., Wang, Y., & Cao, Q. (2024). A Novel Robotic Grasp Detection Framework Using Low-Cost RGB-D Camera for Industrial Bin Picking. *IEEE Transactions on Instrumentation and Measurement*, *73*, 1–12. https://doi.org/10.1109/TIM.2023.3346531

Tactile Sensors. (2025). LOOMIA Soft Electronics | E-Textiles. https://www.loomia.com/tactile-sensors

Tajima, S., Wakamatsu, S., Abe, T., Tennomi, M., Morita, K., Ubata, H., Okamura, A., Hirai, Y., Morino, K., Suzuki, Y., Tsuji, T., Yamazaki, K., & Watanabe, T. (2020). Robust bin-picking system using tactile sensor. *Advanced Robotics*. (world). https://www.tandfonline.com/doi/abs/10.1080/01691864.2019.1702897

Ultralytics. (2026a). *Architecture Summary*. https://docs.ultralytics.com/yolov5/tutorials/architecture_description

Ultralytics. (2026b). *Insights on Model Evaluation and Fine-Tuning*. https://docs.ultralytics.com/guides/model-evaluation-insights

Ultralytics. (2026c). *Object Detection Datasets Overview*. https://docs.ultralytics.com/datasets/detect

Ultralytics. (2026d). *Ultralytics YOLO11 Modes*. https://docs.ultralytics.com/modes

Ultralytics. (2026e). *Val*. https://docs.ultralytics.com/modes/val

Ultralytics. (2026f). *YOLO11 🚀 NEW*. https://docs.ultralytics.com/models/yolo11

Wang, C., Fang, H.-S., Gou, M., Fang, H., Gao, J., & Lu, C. (2021). Graspness Discovery in Clutters for Fast and Accurate Grasp Detection. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 15944–15953. https://doi.org/10.1109/ICCV48922.2021.01566

What is Template Matching? An Introduction. (2024, October 8). Roboflow Blog. https://blog.roboflow.com/template-matching/

Wu, C.-H., Jiang, S.-Y., & Song, K.-T. (2015). CAD-based pose estimation for random bin-picking of multiple objects using a RGB-D camera. *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, 1645–1649. https://doi.org/10.1109/ICCAS.2015.7364621

Xu, X., Bashir, A., & Müller, R. (2024). A robot-system for picking parts from unstructured bins – A practical Approach. *Procedia CIRP, 10th CIRP Conference on Assembly Technology and Systems (CIRP CATS 2024)*, *127*, 98–103. https://doi.org/10.1016/j.procir.2024.07.018

Yan, W., Xu, Z., Zhou, X., Su, Q., Li, S., & Wu, H. (2020). Fast Object Pose Estimation Using Adaptive Threshold for Bin-Picking. *IEEE Access*, *8*, 63055–63064. https://doi.org/10.1109/ACCESS.2020.2983173

Zhuang, C., Wang, Z., Zhao, H., & Ding, H. (2021). Semantic part segmentation method based 3D object pose estimation with RGB-D images for bin-picking. *Robotics and Computer-Integrated Manufacturing*, *68*, 102086. https://doi.org/10.1016/j.rcim.2020.102086

Appendices

Appendix A – Summary Table of Bin Picking Applications

Reference	Methodology	Known/Unknown Objects	Mono-reference/ Multi-referece bins	Bin/Cluttered Environment	Gripper	Sensors
Oh et al. (2012)	Classical Methods, Model-Based	Known	Mono-reference	Bin	Two-finger	Two 2D grey-level Cameras (Stereo Vision)
Buchholz et al. (2013)	Classical Methods, Model-Based	Known	Mono-reference	Bin	Two-finger	Laser Line Scanner
Pretto et al. (2013)	Classical Methods, Model-Based	Known	Mono-reference	Bin	Vacuum	2D grey-level Camera
Domae et al. (2014)	Classical Methods, Model Free	Unknown	Mono-reference	Bin	Two-finger and Vacuum	Depth Sensor
Buchholz et al. (2014)	Classical Methods, Model Free	Known and Unknown	Mono-reference and Multi-reference	Bin	Two-finger	Depth Sensor
Wu et al. (2015)	Classical Methods, Model-Based	Known	Multi-reference	Cluttered Environment	Two-finger	Depth Sensor
Chang & Wu (2016)	Classical Methods, Model-Based	Known	Mono-reference	Bin	Two-finger	2D Camera and Laser Line Scanner

Reference	Methodology	Known/Unknown Objects	Mono-reference/ Multi-referece bins	Bin/Cluttered Environment	Gripper	Sensors
Le & Lin (2019)	Deep Learning, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera
Yan et al. (2020)	Classical Methods, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera
Jiang et al. (2020)	Deep Learning, Model Free	Known and Unknown	Multi-reference	Bin	Vacuum	Depth Sensor
Tajima et al. (2020)	Hybrid, Model-Based	Known	Mono-reference	Bin	Two-finger	3D Camera
Araki et al. (2020)	Deep Learning, Model-Based	Known	Multi-reference	Cluttered Environment	Vacuum	Depth Sensor
Iriondo et al. (2021)	Deep Learning, Model Free	Known and Unknown	Multi-reference	Bin	Two-finger and Vacuum	3D Camera
Zhuang et al. (2021)	Deep Learning, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera / Depth Sensor and RGB Camera
Gou et al. (2021)	Deep Learning, Model Free	Known and Unknown	Multi-reference	Cluttered Environment	Two-finger	3D Camera
Wang et al. (2021)	Deep Learning, Model Free	Known	Multi-reference	Cluttered Environment	Two-finger	3D Camera

Reference	Methodology	Known/Unknown Objects	Mono-reference/ Multi-referece bins	Bin/Cluttered Environment	Gripper	Sensors
Park et al. (2022)	Hybrid, Model-Based	Known	Mono-reference	Bin	Two-finger	LIDAR Sensor
X. Li et al. (2022)	Deep Learning, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera
Jiang et al. (2022)	Deep Learning, Model Free	Unknown	Multi-reference	Bin	Vacuum	3D Camera
G. Li et al. (2023)	Deep Learning, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera
Raj et al. (2024)	Deep Learning, Model Free	Unknown	Multi-reference	Bin	Two-finger	3D Camera
Xu et al. (2024),	Hybrid, Model-Based	Known	Mono-reference	Bin	Vacuum	3D Camera and 2D RGB Camera
Sun et al. (2024)	Hybrid, Model Free	Unknown	Mono-reference	Bin	Two-finger	3D Camera