Applications of Object Detection in Industrial Contexts Based on Logistics Robots

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

Christian Poss

Berlin, 21.10.2019

Hauptgutachter: Prof. Dr. Daniel Göhring
Zweitgutachter: Prof. Dr. Vahid Salehi

Tag der Disputation: 14.05.2020

Contents

List of Figures

Abstract

The aim of this doctoral thesis is to automate the logistic flow of material under industrial conditions. For this purpose, the running processes as well as their characteristic features are first analyzed. Based on this, requirements are derived for the robots to be created and technical solutions are sought. This development of the solution is divided into the five sub-areas: Decision, Perception, Action, Interaction and Safety. Depending on the overall task to be examined, three robot concepts are created for this purpose. One for depalletising in incoming goods, one for automating provisioning on the assembly line and one for palletising in empty goods sorting.

Since the logistic processes - in contrast to the production processes in already automated production areas - are mainly characterized by a high variance, dynamic changes and challenging influencing variables from the industrial environment, a very powerful perception module is necessary for the successful use of the robots. This is made up of five modules for all three robots. Depending on the function, a decision is made between function and robustness modules. The former include Detection-, Selection- and Localization-Modules. Initially, all objects are detected by Neural Networks. Then the relevant object for the process execution is selected from the set of detected objects and its gripping point is determined. In order to increase the robustness to values suitable for industrial use, these three modules are supplemented by the Aggregation- and Validation modules. In the Validation-Module, the recognized classes are examined on the basis of predefined rules for their plausibility in occurrence itself as well as in appearance among each other. Finally, this process is executed several times. The results are evaluated in the Aggregation-Module. This means that outliers, for example, caused by background processes in the plant, can be excluded.

Finally, the designed robots are evaluated. Depending on the respective hardware status, this was done in series production in the factory, in test trials in the factory as well as in test trials on the development areas. The industrial suitability of the developed solution can be demonstrated by this holistic test.

Zusammenfassung

Ziel dieser Dissertation ist es, den logistischen Materialfluss unter industriellen Bedingungen zu automatisieren. Zu diesem Zweck werden zunächst die laufenden Prozesse sowie deren charakteristische Merkmale analysiert. Daraus werden Anforderungen an die zu erstellenden Roboter abgeleitet und technische Lösungen erarbeitet. Diese Entwicklung der Lösung gliedert sich in die fünf Teilbereiche: Entscheidungsfindung, Wahrnehmung, Ausführung, Interaktion und Sicherheit. Abhängig von der zu untersuchenden Gesamtaufgabe werden zu diesem Zweck drei Roboterkonzepte erstellt. Eines für die Depalettierung im Wareneingang, eines für die Automatisierung der Bereitstellung am Montageband und eines für die Palettierung in der Leergutsortierung.

Da sich die logistischen Prozesse - im Gegensatz zu den Produktionsprozessen in bereits automatisierten Produktionsbereichen - vor allem durch eine hohe Varianz, dynamische Veränderungen und anspruchsvolle Einflussgrößen aus dem industriellen Umfeld auszeichnen, ist für den erfolgreichen Einsatz der Roboter ein sehr leistungsfähiger Wahrnehmungsalgorithmus erforderlich. Dieser besteht roboterübergreifend aus fünf Modulen. Je nach Funktion wird zwischen Funktions- und Robustheitsmodulen entschieden. Erstere beinhalten die Module zur Erkennung, Auswahl und Lokalisierung. Zunächst werden alle Objekte von Neuronalen Netzen erkannt. Anschließend wird aus der Menge der erfassten Objekte das für die Prozessausführung relevante Objekt ausgewählt und die Greifpunkte bestimmt. Um die Robustheit mit Blick auf Industrietauglichkeit zu erhöhen, werden diese drei Module durch die Module Aggregation und Validierung ergänzt. Im Modul Validierung werden die erkannten Klassen nach vordefinierten Regeln auf ihre Plausibilität sowohl im Auftreten selbst als auch im Erscheinungsbild untereinander untersucht. Schließlich wird dieser Prozess iterativ ausgeführt. Diese Ergebnisse werden im Aggregationsmodul ausgewertet. Dadurch können Ausreißer, z.B. durch Hintergrundprozesse im Werk oder verrauschte Sensorinputdaten, ausgeschlossen werden.

Abschließend wird der Wahrnehmungsalgorithmus sowie die konzeptionierten Roboter evaluiert. Dies erfolgt in Abhängigkeit des Hardware-Status in der Serienproduktion im

Werk, in Testversuchen im Werk sowie in Testversuchen auf den Entwicklungsbereichen. Die industrielle Eignung der entwickelten Lösung konnte durch diesen ganzheitlichen Test nachgewiesen werden.

List of Abbreviations

BMW	Bayerische Motoren Werke
CAD	Computer Aided Design
CAN	Controlled Area Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
GPU	Graphics Processing Unit
HOG	Histogram of Oriented Gradients
HRC	Human Robot Collaboration
IoU	Intersection over Union
IT	Information Technology
LED	Light-Emitting Diode
mAP	Mean Average Precision
OEM	Original Equipment Manufacturer
PC	Personal Computer
PLC	Programmable Logic Controller
QR	Quick Response
RGB	Red Green Blue
RGB-D	Red Green Blue Depth
ROI	Region of Interest
ROS	Robot Operating System
SCARA	Selective Compliance Assembly Robot
SGD	Stochastic Gradient Descent
SSD	Single Shot Detector
SVM	Support Vector Machine
TCP	Tool Center Point
UR	Universal Robot

1 Introduction

1.1 Thesis Motivation

1.1.1 Logistics Challenges

In recent years, logistics has developed more and more away from a real enabler to vehicle production towards a core component of it [KKS18]. This is reflected, for example, in the share of logistics costs per part. Logistics activities cause almost 25% of the total cost of manufacturer purchase prices. The remaining 75% are mainly composed of material costs (66%) and production costs (15%) [Gün11].

This increase in costs is primarily due to the increased complexity of the entire logistics system [Bau14]. The essential contributing factors are the increase of the variety of offered vehicle models as well as the respective individualization possibilities of these vehicle derivatives [Gün11]. For example, the number of vehicle models in the Bayerische Motoren Werke (BMW) Group rose from six (1990) to 29 within 25 years. Due to the personalization possibilities of these vehicles demanded by customers, there are now 10 to the power of 32 possible variants when ordering a vehicle [Die14]. Together with the reduced vertical range of manufacture due to advancing globalization and increased cost pressure, these developments have led to a significant increase in the number of components to be delivered to the assembly line in the vehicle plants. While in 2000 more than half (55%) of the body components were still produced by the Original Equipment Manufacturers (OEMs) themselves, this figure fell to 29 % by 2015 [Gün11].

The increase in the number of parts to be provided leads directly to an increase in the work steps required in the plants, which ultimately results in additional personnel requirements. Due to the growth efforts of the automotive OEMs as such, the number of vehicles to be produced in the individual plants was continuously increased in parallel [Die14]. This leads to a further strain on the personnel situation, especially in those areas where the relaxed labor market situation has already led to almost full employment [Goo19].

One practical approach that encourages productivity while reducing the costs is the holistic automation of the logistics material flow in the plants [Bau14]. In addition to the economic gains, it reduces the need for human employment in monotonous and unergonomic handling tasks and promotes more demanding working areas such as final vehicle assembly or quality assurance [Gra17].

1.1.2 Evolving Automatisation

The high-dynamic production processes in the modern industry increase the complexity of logistics and thus encourages the growing adoption of automation solutions. Therefore, robotic solutions and intelligent machines are transforming manufacturing industries to a higher level of automation. The number of industrial robots worldwide will increase from 1.828.000 (2016) to over 3 million by 2020. The main markets are Asian countries such as China, South Korea, and Japan as well as America and Germany [TZW16].

With falling costs and innovative business models such as robot leasing, this trend is affecting not only the production halls of large OEMs but also those of small and medium-sized enterprises.

The highest use can be observed in the electronics industry in particular in the automation of the thoroughly repetitive production steps [Lit+18]. The stagnating figures in the automotive industry, on the other hand, show that the number of processes to be automated by classic industrial robotics has been exhausted [TZW16]. For the automation of logistics processes, which are characterized by flexibility and dynamic changes with a high variance, autonomous and intelligent systems with a high level of adaptability instead of standard automation technology are therefore necessary [Arn+17].

1.1.3 Artificial Intelligence

Artificial intelligence is an attempt to automate complex problems by simulating specific human decision-making structures.

While the basic idea behind the concept of artificial intelligence in its most original form dates back to 1748, it was not until the 1960s that more considerable attention was paid to it. The discussions at that time were characterized by euphoric expectations regarding the ability of computers to solve tasks that require intelligence when performed by humans [RN05] [WL93].

Initially, areas of application with initial success were rule-based strategy games such as chess or checkers as well as areas with formally represented knowledge contexts, so-called expert systems. The MYCIN developed at Stanford University at the beginning of the 1970s, which was used for diagnosis and therapy of blood infections, became particularly famous. Scientific evaluations confirmed that his decisions were at eye level with those of human colleagues. However, such systems had only limited success because it was very time-consuming to convert the required correlations into formal rules manually.

In response, the field of machine learning has evolved. The aim is to circumvent this weakness by learning such interrelationships independently from the computer. Artificial neural networks have proven to be the most successful structure in recent years. Initially, these were linear models, such as the McCulloch-Pitts cell (1943) or the Adaline model (1959), which showed weaknesses in the acquisition of even simple logical functions. Only progress in the development of non-linear, multilayer, folding neural networks, and the hardware required for training made it possible to outperform classical methods. In 2009, the so-called Deep Learning approaches were able to beat classic approaches with manual feature extraction in numerous pattern recognition competitions such as the popular ImageNet.

Further fields of application are the processing of natural language, sensor technology, cybernetics, and robotics. In some sub-areas, moreover, scenarios are repeatedly successful in which AI beats people, even if the use case is not based exclusively on rule-based, learnable behavior patterns. For example, the AlphaGo system defeated the world's best Go player. Likewise, successes could be achieved in the most sophisticated computer game Dota 2 with which it depends with humans also on abilities such as intuition [Boh15] [Dee19].

Despite the current euphoria in this area, however, there are also critical limitations. Thus, error susceptibility as well as algorithms that exhibit unpredictable behavior can cause severe damage in the wrong applications such as autonomous vehicles [Dyk94] [Spy17].

1.2 Thesis Contributions

The thesis on which this doctoral thesis is based is that the innovative use of intelligent robots can be used to automate complex processes in logistics under industrial conditions that have not yet been automated.

In addition to the respective handling task to be automated, it is necessary to satisfy certain requirements in order to enable the systems to be integrated into the existing structures and processes of the individual plants. Particular focus is also placed on the consideration of security measures and the guarantee of robustness in process execution. The specific requirements are dealt with in Chapter 3.

By prooving the stated thesis above, the doctoral thesis will achieve the following contributions:

- Generic hardware concept for material handling in industrial environments
  - Conceptualization of a modular robot construction kit for the automation of the four logistics relevant tasks: Depalletizing, picking, staging and palletizing
  - Generic gripping concept for handling different objects of different geometries made of different materials
  - Integration of a safety system to enable the robots to be used in areas that are not empty of people
  - Construction of a modular robot state machine to enable an individual robot reaction to different events in the real process flow
- Modular Perceptual Algorithm for achieving fully autonomous systems
  - Detection: Detection of various objects under industrial influences
  - Selection: Selection of process-relevant objects from the set of recognized objects
  - Localization: Determination of a gripping point for stable process execution
  - Validation and Aggregation: Validating the calculated quantities to ensure robust and repeatable processes

- Evaluation of the perception algorithm under real operating conditions
  - Simplified validation procedure by mirroring the algorithms on mobile devices
  - Comparative tests of the overall performance on compiled data sets as well as the use in the real environment
  - Comparison tests of different module combinations within the framework of the Perception Algorithm to achieve the best performance variant

1.3 Thesis Outline

Because of the already mentioned challenges, a systematic approach is necessary to achieve the goals set. This is shown in Figure 1.1 in the form of a flow chart and is explained in more detail below.

Following on from the introduction (Chapter 1), the knowledge necessary for a general understanding of this work will first be conveyed in the basics chapter (Chapter 2). Based on an analysis of robots used in industry and their fields of application, the logistics, the internal material flow will be presented.

In Chapter 3, a generic robot concept is developed and presented. This will be explained later in this Chapter, divided into the essential generic function blocks of intelligent robots: Decision, Action, Perception, Interaction, and Safety.

In order to be able to benefit from the use of automation technology despite the inherent challenges in logistics, the most crucial functional robot component, the perception of environment and objects, is presented in more detail in Chapter 4. Based on an analysis of existing methods in this area, the underlying architecture underlying the Perception Algorithm is presented. This Chapter outlines the design and the implementation of the considered perceptual system, which includes three main modules, namely detection, selection, and localization, towards reaching an object target. The finalized algorithm is then validated for compliance with the initial requirements. In order to overcome the weak points of the Perception Algorithm and to improve its accuracy, two additional modules, namely validation, and aggregation, are proposed in Chapter 5.

After the final evaluation (Chapter 6), the main findings of this work are summarized and discussed (Chapter 7).

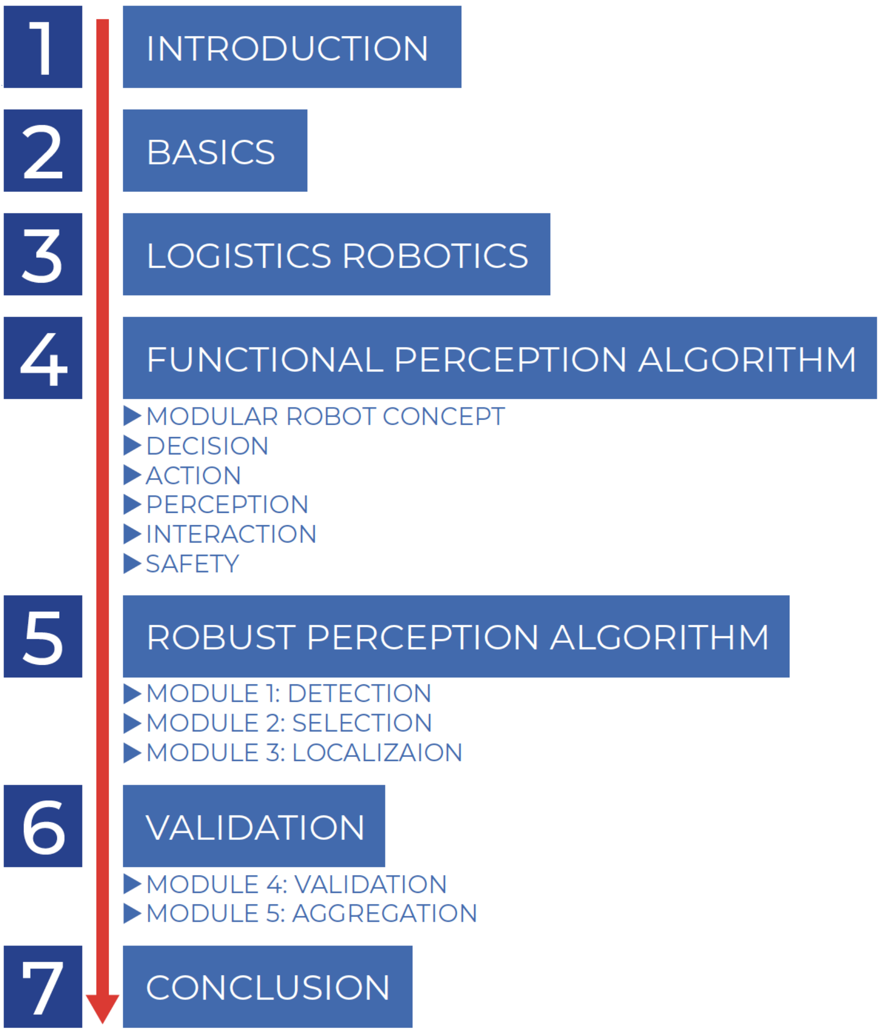


Figure 1.1: Thesis outline (own illustration)

2 Fundamentals and Literature Survey

In this chapter, relevant terms of the doctoral thesis are first introduced in more detail in order to create a shared understanding of the further course of the work. Based on these definitions, an overview of robot applications already used in the production industry and the environmental requirements necessary for their use is given. Moreover, the logistics material flow in high-variant assemblies, as well as its general conditions, including the objects occurring during these processes, are explained.

2.1 Definitions

To allow a common understanding, the following definitions briefly define critical terms for the work.

High Variant Assembly

As a high variant assembly, the "mixed model assembly" line in the context of this work refers to the assembly of an end product which is available in various variants and variant combinations [Küb+16]. Since such a diversity of variants has considerable effects on the overall logistics system, it is treated as a differentiation criterion. An example of this is the automotive industry, which is taken up in this doctoral thesis for better clarity.

Inbound Logistics

The considered system in this dissertation comprises all the elements required to perform logistical tasks between the supplier's goods issue and the assembly site on the assembly line. In the literature, this system is generally referred to as inbound logistics to supply production and assembly locations. [MH11]

Autonomous versus Automated

Automated systems serve to fulfill a recurring task. This is done by an initially programmed, fixed sequence of predefined movements. In contrast, autonomous systems can react independently to changing environmental conditions and adapt their behavior accordingly. This definition is followed in the context of the present work by describing intelligent systems “Autonomous”. [MH10]

Robot

A robot is a programmable multi-purpose handling device for moving material, workpieces, tools or special equipment. The freely programmable motion sequence makes it suitable for a wide variety of tasks. One of the most popular collective works on the subject of robotics is by Matthias Haun. As it provides a good overview of the topic explained here, it will be used regularly in the following as an overall reference. [Hau13]

Industrial Robot

An industrial robot is a universal, programmable machine for handling, assembling or machining workpieces. These robots are designed for use in industrial environments (e.g., automotive manufacturing). [Hau13]

Collaborative Robot

A special type of robot is the collaborative robot, which is designed in such a way that it can work with humans in a room without a separating protective device. This opens up completely new application possibilities, but also brings new requirements for the safety concept with it, which can lead to restrictions, for example, concerning the payload, and cycle time. In the literature, these robot types are also referred to as Human Robot Collaboration (HRC) robots or lightweight robots. [Hau13] [RD01]

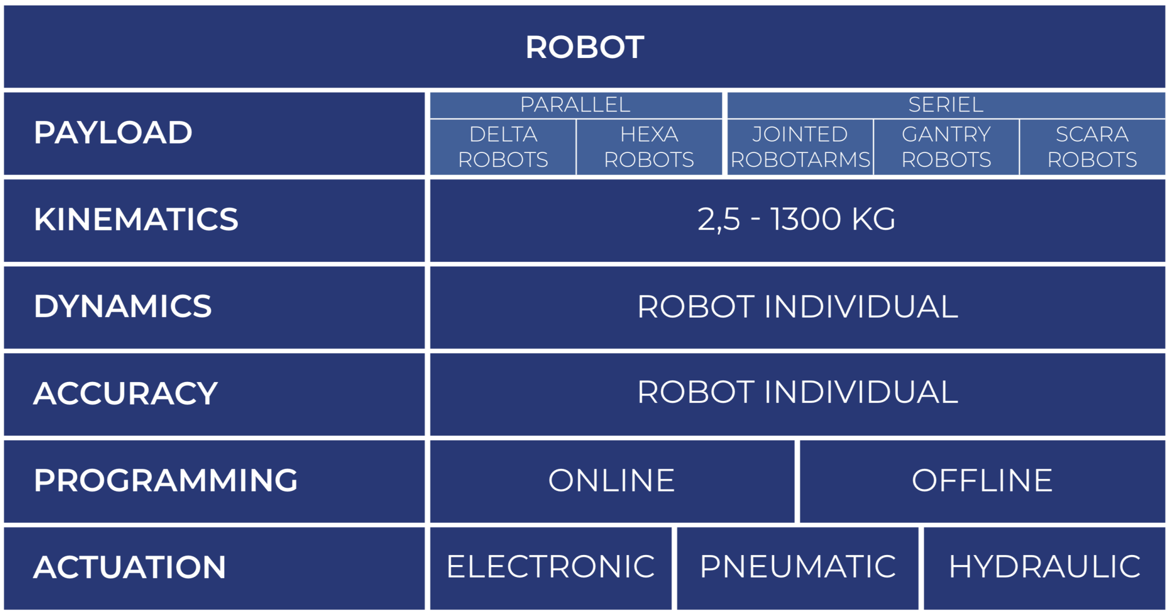
2.2 Robots in Industry

As already explained in the previous subchapter (Chapter 2.1), in the literature, an industrial robot is introduced to be a universal, programmable machine [Hau13]. Numerous different industrial fields of application can be assigned to it. After a short technical description, these fields of application will be examined in more detail.

Industrial robots are purchased as a standardized primary product and then individualized for their respective applications based on individualized peripherals [ABB19] [KUK19]. For this reason, the following section first deals with differentiation criteria and their possible forms of occurrence for robots and grippers before focussing the on the respective applications.

2.2.1 Industrial Robots

Significant distinguishing features for the classification of industrial robots are kinematics, payload, dynamics, accuracy, programming, and actuation. The payload, i.e., the maximum mass that can be attached to the end of the manipulator, is to be classified in a range between 2.5 and 1300 kg, depending on the model [Hau13]. This is determined by the robot kinematics and is shown in Figure 2.1 together with the possible characteristic forms. A distinction can be made between parallel and serial kinematics. Examples of the former are Delta and Hexapod robots. The latter can be further divided into classic jointed-arm robots and gantry robots. The number and arrangement of the axes are decisive for the movement possibilities of robot arms. In most versions between five and seven are used. Special cases are dual-arm robots, palletizing robots, and Selective Compliance Assembly Robots (SCARAs). Dual-arm robots have 15 axes thanks to two 7-axis arms and an additional rotation axis. Palletizing robots usually have two or four driven rotary axes and one linear axis. Robots with three parallel rotation axes and one linear axis are called SCARAs. The axes are controlled by the robot controller, in which the movements of the robot are defined. These programming efforts can be made either online or offline. The movement specifications are executed by electric, hydraulic, or pneumatic drives and monitored by the internal sensors of the robot. For this purpose, incremental encoders, interference patterns, and light barriers can be used. Since in most cases, the robots are not used in isolation in the factories but are dependent on upstream and downstream process steps, they also frequently have external sensors. Image processing systems, triangulation sensors, light barriers, or ultrasonic sensors can thus be used to react to predefined events. [Hau13]



ROBOT					
PAYLOAD	PARALLEL		SERIEL		
	DELTA ROBOTS	HEXA ROBOTS	JOINTED ROBOTARMS	GANTRY ROBOTS	SCARA ROBOTS
KINEMATICS	2,5 - 1300 KG				
DYNAMICS	ROBOT INDIVIDUAL				
ACCURACY	ROBOT INDIVIDUAL				
PROGRAMMING	ONLINE		OFFLINE		
ACTUATION	ELECTRONIC	PNEUMATIC	HYDRAULIC		

Figure 2.1: Robot characteristics (own illustration after [Hau13])

2.2.2 Effectors

The actual execution of the automation task is carried out by individual manipulators attached to the robot arm. These include grippers and devices such as cameras or inspection equipment [Hau13]. While the latter does not differ from the devices for manual operation, the following explanations focus on grippers. A gripper is a subsystem of a handling device that provides temporary contact with a gripping object. It secures position and orientation for pick and place operations. Holding is achieved with force-generating, form-fitting, or material-pair components. The grippers can also be divided into different classes. Depending on their physical operating principle, they can be divided into mechanical, pneumatic, electromagnetic, or form-fit grippers. This subdivision is shown in Figure 2.2. [Sch19] [MAD04]

2.2.3 Applications

The number of industrial robots used has significantly increased in recent years, particularly in high-wage countries [Unk18]. In the course of the advancing automation as well as the continuous improvement of the sensor, control, and actuator technology of

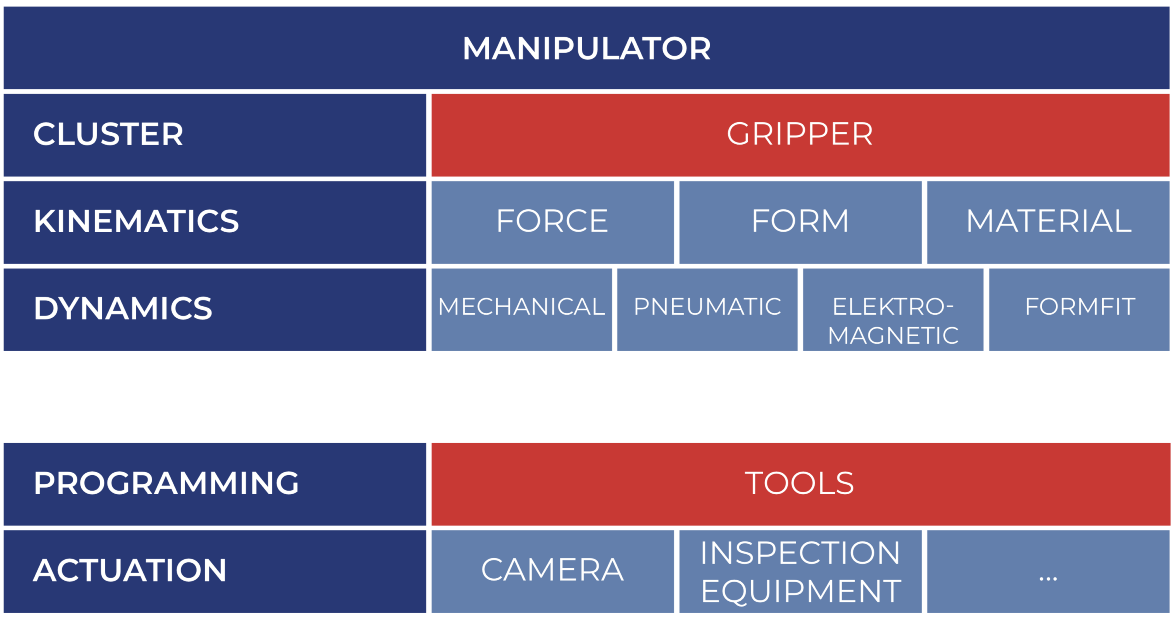


Figure 2.2: Effector characteristics (own illustration after [Hau13])

the robots, these can be found in an ever broader application spectrum. These include primarily unergonomic, hazardous, and monotonous processes or processes with very high accuracy requirements. This results in the following main fields of application:

- Welding
- Painting
- Measuring
- Handling. [Hau13] [KUK19]

Industrial robots are complex technical systems that require certain framework conditions in order to perform their intended tasks. These framework conditions can be further divided into robot-specific components, first and second peripherals. The former comprises, as already mentioned, the robot, the controller, internal protection and safety devices, internal sensors and the end effector or, in the case of broader applications, the effector exchange system. The first periphery includes safety devices, measuring and testing equipment, effector magazines, clamping and positioning systems, workpiece storage, feeding devices as well as position and object recognition devices. The second periphery consists of transfer equipment, technological equipment and storage, and transport systems. Storage devices are used for the temporary storage of handling goods near the workplace. Through feeding devices, these goods can be brought into the working area of the robot to a predefined position. There they are fixed by clamping and positioning systems to ensure trouble-free process execution. After the task has been completed, the result can be checked using measuring and testing equipment. This is particularly relevant for interlinked robots since compliance with the necessary framework conditions must also be ensured for the subsequent process steps. An important aspect in the application of robotics in industry, in addition to the functional components, is fulfilling the safety requirements. This is predominantly achieved by separating the movement spaces of man and robot in the form of fences. Depending on the process, technological auxiliary equipment may also be necessary. These include, for example, devices for reworking electrode tips in spot welding guns. As the end effector, the respective periphery is strongly dependent on the selected application field in order to provide a deeper insight. These are examined in more detail in the following subchapters and presented with a view to the necessary framework conditions for successful process execution. [Hau13]

Welding

Welding usually occurs in the industry when sheet metal components are joined together. This implies high component weights, unwieldy component dimensions as well as very high accuracy requirements for each of the often several thousand welding spots. In recent decades, for example, this has led to almost complete automation of body assembly. In robot lanes, several hundred industrial robots assemble individual parts and assemblies from them into a finished chassis. [Hu97]

The components to be joined are inserted manually into defined workpiece carriers. This enables a robust process execution under the high accuracy requirements. Auxiliary services such as the refilling of rivets or screws and the maintenance of the welding guns are also carried out manually. The initial programming of such a robot network for the introduction of new vehicles is associated with a very high expenditure of time since all activities must be perfectly coordinated. If one sub-process fails, the entire component is damaged. For this reason, greater attention is also being paid to ensuring the best possible framework conditions. For example, cleanliness or the provision of faultless components is critical. As a rule, only one vehicle derivative is manufactured in such lanes. Due to the modularization of the vehicles, however, it is now common to produce a small number of variants in one production line. An example of such a production line can be seen in Figure 2.3. [Pel+14] [KUK19]

7-axis robot arms are used due to the high weights of up to 500 kg to be handled and the necessary room for maneuver for manipulating large components. Grippers and tools, in this case, welding tongs, are required as end effectors for manipulating the components during welding. In order to ensure that all components were placed at the intended positions before the welding process, various sensors are installed at their pick-up points. The robots are initially programmed offline. In addition to the robot controller, which controls the sequence of an individual robot, a higher-level controller is also integrated, which organizes the interaction of the individual robot cells. In order to guarantee human safety despite the delicate process, the robots are fenced with rigid fences. If a door inside is opened for maintenance purposes, all robots in the entered area are immediately shut down. [Che+05]

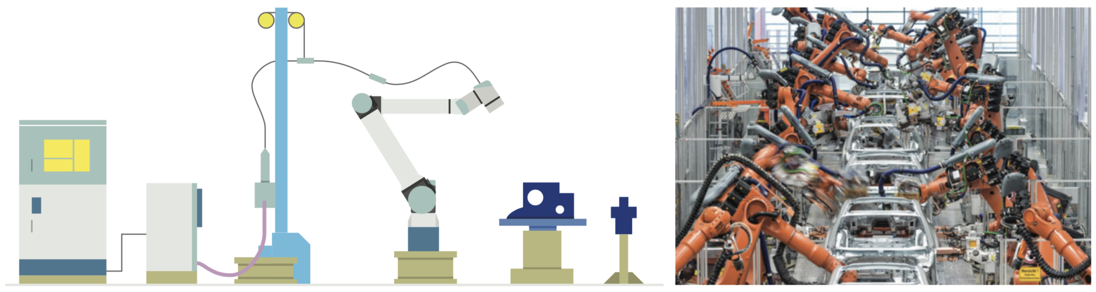


Figure 2.3: Welding robots in bodyshop production lanes (own illustration, imagesources: [Hau13], faps.fau.de)

Painting

An equally complex process is painting. In addition to the high requirements for uniformity and repeatability of this activity, the toxic environmental conditions, in particular, have led to full automation of these activities.

The chassis are conveyed to the painting station on a conveyor belt. No other sensors are installed on the painting robots, and the bodywork must be placed at precisely the right target points. There they are painted by two to five robots. In addition to the painting itself, the robots perform additional manipulation tasks, for example opening doors or flaps. Additionally, to the painting nozzles, mechanical pins are attached to the end effectors. Since the robots do not have to take weights, criteria such as mobility and the accessibility of all relevant points play a dominant role in robot selection [AC15]. As the painting takes place in closed chambers due to the nitrous vapors (compare Figure 2.4), no additional safety measures on the robots are necessary. The vehicle-dependent paths that the robots have to follow are usually initially pre-set offline. In the meantime, however, there are already first approaches to automatically generate these trajectories online based on Computer Aided Design (CAD) data. This would allow a specific variance of vehicles to be painted [CZ13]. A further influencing factor on the variance to be processed in the painting stations is the color selection of the vehicles. It is indeed possible to fill different paint into the supply lines to the robot after each vehicle. However, since such a procedure requires cleaning processes between vehicles, which costs paint, time, and money, this degree of freedom is primarily dispensed with. This means that the same color is always applied to the bulk of vehicles. [Muz+12]

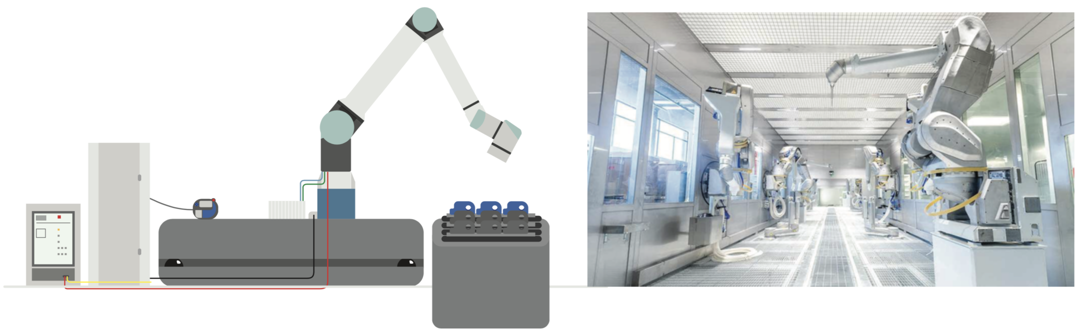


Figure 2.4: Painting robots for different industries (own illustration, imagesources: industr.com/de/, sames-kremlin.com)

Quality Inspection

Automated measurement and quality control also represent an important process step which is pictured in Figure 2.5. Manual tests are always subject to human influences. For example, measurement errors or inaccuracies in the placement of cameras do not guarantee the completely correct measurement of product quality. In automation, the repeatability and precision of robots can be used. Besides, potentials about measuring speed can be seized in this way. [KPL10]

Similar to painting, it is of high importance that the product to be inspected is conveyed to an exactly predefined position during measurement. Otherwise, the robot will not be able to perform the desired measurements while running its offline pre-programmed trajectories. Also, in this process step, a small variance of the products to be measured is conceivable. However, the robot must be informed before each measurement which product is to be checked and which predefined path should be followed. [KT81] [BS04]

In this application, it should also be noted that robots are mainly used to acquire images

Figure 2.5: Robot working in measuring tasks (own illustration, imagesources: businesskorea.co.kr/news/)

or measurement data. The actual evaluation of this data takes place in downstream computer stations [BGW03].

Manipulation

Handling processes, especially for large and heavy components, are also predestined for automation for ergonomic reasons. Examples for those can be supply processes during production, such as the component handling for welding mentioned above, or isolated handling processes, like those found in logistics processes. Two examples, therefore, can be seen in Figure 2.6. [KET07]

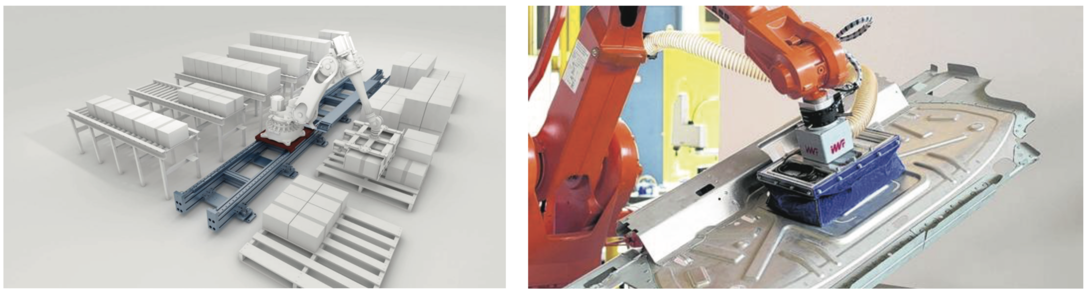


Figure 2.6: Robotic Usage for industrial material handling (own illustration, imagesources: vdi-nachrichten.com/Fokus, i-need.de)

A particularly unusual activity here is the depalletizing of unmixed pallets. This can often be found in goods receipts in industries that do not have a significant product variance. The beverage industry should be mentioned here as an example. A comparison of the various depalletizing systems implemented also illustrates the high importance of low variance and a high degree of standardization. [ZS09]

Further Applications

In addition to these described industrial applications, niche applications have been added in recent years. Today, for example, industrial robots can be found in fairground rides or other entertainment applications like it is shown in Figure 2.7 [Kuk16]. However, as these do not meet the requirements of industrial automation pursued in the context of this doctoral thesis, they will not be considered further here.

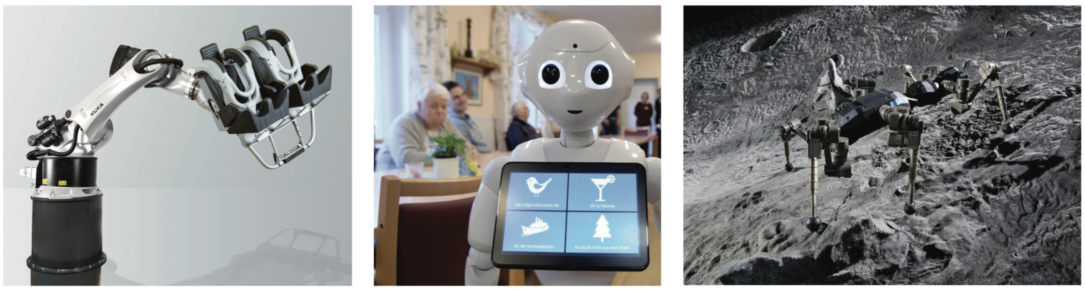


Figure 2.7: Further possible robot applications (own illustration, imagesources: robotik.dfki-bremen.de, kuka.com/de-de/, welt.de/wirtschaft)

2.3 Logistics

There are numerous definitions of the term logistics, which can be reduced to a few common core elements [Arn+17]. These include goods (materials, substances), persons, information, energy, material flow means, means of production, information flow means as well as the underlying infrastructure in the form of buildings, areas, and paths [Ihm06] [FD04].

Automotive logistics is divided into procurement, production, distribution, and disposal logistics. These include transport and warehouse logistics as sub-functions [Sch08]. The three main areas are also synonymous with inbound, in-house, and outbound logistics [GB13]. As it is not possible to separate these areas from each other, the inbound logistics how it is understood at the center of this work covers all processes and material flows from the outgoing goods department of the first-tier supplier to the provision at the assembly line in the production plants of the OEMs.

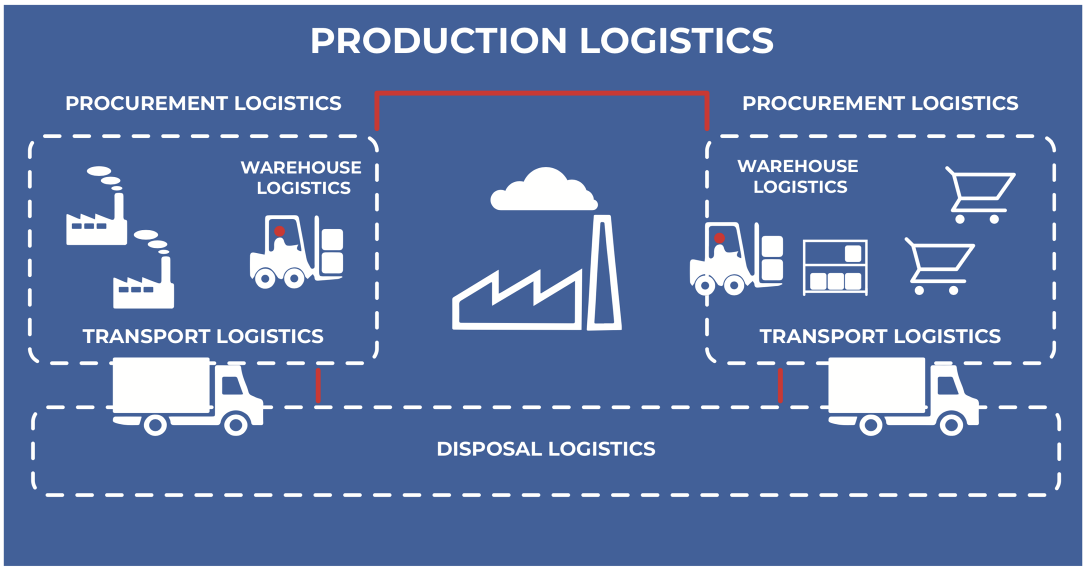


Figure 2.8: Overview and differentiation of the different logistics areas (own illustration after [Sch08])

The overall goal of logistics is the fulfillment of the so-called 6Rs: the provision of

- the right amount
- the right objects
- to the right destination
- at the right time
- in the right quality
- and for the right price.

For this purpose, all internal and cross-company flows of goods and information are planned, controlled, coordinated, carried out, and monitored. [Ihm06]

2.3.1 Intralogistics

Intralogistics comprises the organization, control, implementation and optimization of internal material flows, information flows, and goods handling in industry, trade, and public institutions. [MH11]

In order to achieve the goals mentioned in the previous chapter, the conveying and storage technology and complete systems used must be adapted to the product properties, such as size, shape, weight, and sensitivity of the material to be handled. This is achieved in logistics planning. Also of importance is the creation of a high degree of flexibility to adapt quickly to changing frameworks and environmental conditions. Nevertheless, costs must be constantly reduced, and processes must be carried out with a high degree of reliability. Process errors in intralogistics can lead to a shutdown of the assembly line and thus to high costs. [Sch08]

In addition to the production-synchronous provisioning processes, which have already been increasing over the last years in particular for large assemblies such as seats or the instrument panel, the majority of the required components are still delivered to the assembly lines in asynchronous production [Arn+17]. This material flow - which is shown in Figure 2.9 - is the focus of this work and will, therefore, be examined in more detail below.

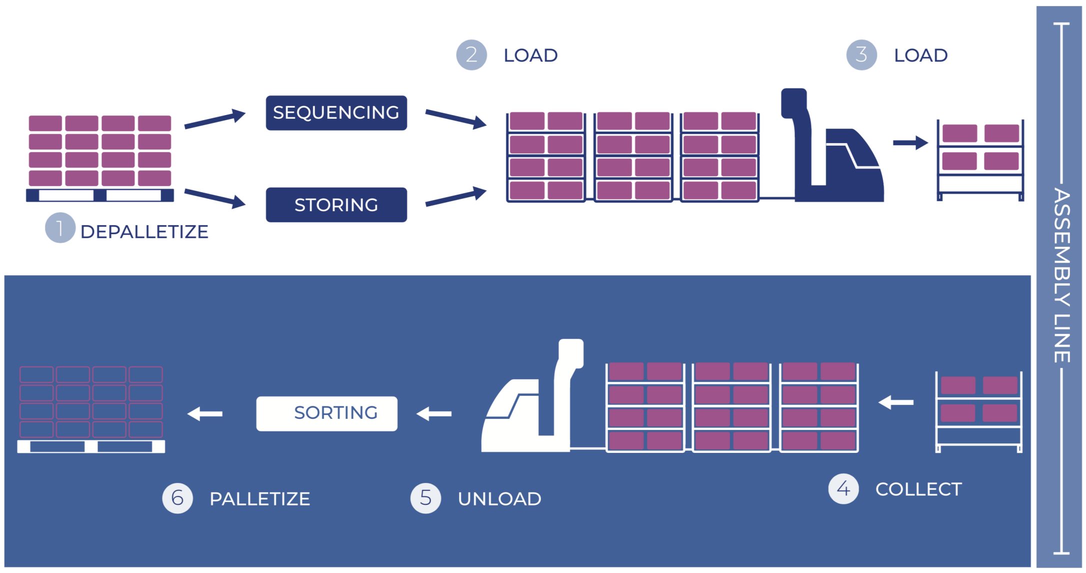


Figure 2.9: Necessary handlingsteps in the intralogistics processchain (own illustration)

The components to be assembled are delivered in containers on pallets by the suppliers on trucks. After the trucks have been unloaded, the pallets are transported to the goods receiving area. There the individual containers are depalletized (1) and stored in an automated small parts warehouse. As soon as the components contained in the container are needed, they are removed from storage and delivered to the assembly line (3). Since the space available on the assembly line is limited, not all containers can be provided there. Therefore some components have to go through sequencing steps. The components there are getting pre-sorted for the next vehicles to be assembled and then transported separately to the provision areas. Transport - direct or indirect - to the assembly line is currently mainly carried out by tugger trains. If the containers are directly from the small parts warehouse, the shelves on the tugger train trailers are filled automatically. Arrived at the assembly line, the containers are removed by the tugger train driver and transferred to the staging racks. In return, the empty containers are collected and reloaded onto the tugger train (4). The empties collected in this way are then sorted on conveyor systems in the empties sorter and palletized by type (6). Finally, these pallets are prepared for transport and sent back to the suppliers. Since the explanations in this doctoral thesis concentrate on the handling of containers, the following handling steps are the focus of the following chapters:

- Depalletizing of full containers
- Removal and provision of containers from one shelf to another
- Palletizing of empties. [Arn+17] [Sch08]

These process steps are analyzed in more detail below. The following explanations are based on process analyses in several car factories. General explanations can be found in [HW07], [Kro+15] and [La 94].

Depalletizing

During depalletizing, the pallets with the stacked containers are first brought onto a conveyor system by forklifts. This conveyor transports the pallets to the depalletizing stations as it can be seen in Figure 2.10. First of all, the banding and the pallet covers are removed. The barcodes of the pallet can then be scanned. As soon as the pallet has been initialized in the system, the containers can be removed individually from the pallet and placed on the next conveyor system. There, the barcodes on the single labels of each container are scanned. Based on this identification, the container can be assigned a storage location in the automated small-parts warehouse.

The full containers can weigh up to 15 kg. Not only the weight to be handled is particularly challenging in this process step, but also the weight distributions that can occur. These make container handling more difficult, especially with bulk materials such as screws. It should also be noted at this point that both the positioning by the conveyor system and the position of the containers on the pallets are subject to an individual tolerance of several centimeters.

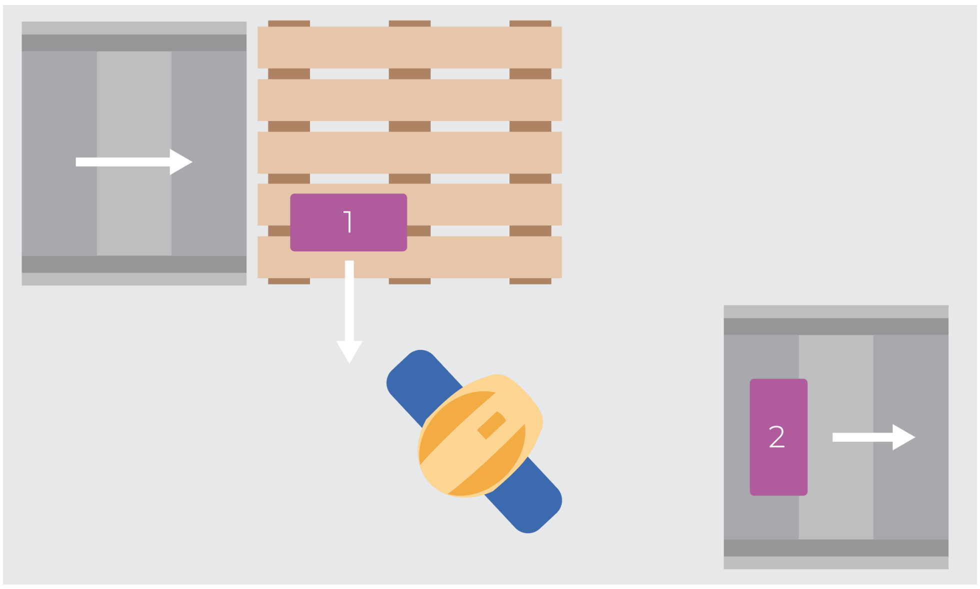


Figure 2.10: Visualization of process and environment: depalletizing (own illustration)

Supply of the Assembly Line

In the case of the assembly line supply (Figure 2.11), the containers are placed on shelves on tugger train trailers at the intended assembly locations on the assembly line. The containers are located in these gravity-driven shelves on three levels. There, the containers appear next to each other and one behind the other. Since the shelf rows are significantly longer than the containers, the horizontal container position can vary up to 10 cm. Using the labels affixed to the containers, the components in the containers can be identified and assigned to the corresponding target positions on the provisioning shelf. These racks are also gravity driven roller racks. Before the container can be transferred to the shelf, the barcode is scanned and posted in the internal goods management system. The employee then removes the empty containers from the staging racks and places them on the tugger train racks.

In assembly, the supply locations are sometimes rearranged daily depending on the product variants to be produced. Therefore, all provisioning racks are on castors to allow flexible adaptation of the infrastructure. However, this also leads to a considerable variance, since the shelves can be found in other positions with precise tolerances and twists.

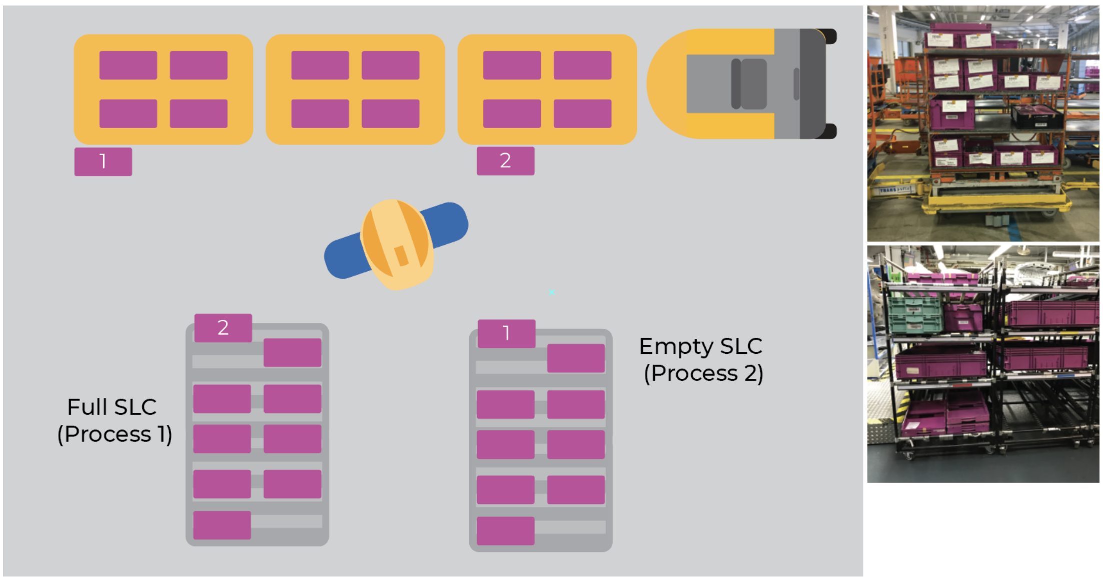


Figure 2.11: Visualization of process and environment: supply of the assembly line (own illustration)

Palletizing

Before the empty containers can be returned to the suppliers, they must be sorted and palletized. For this purpose, the containers are removed again from the tugger train racks, labels, and other contaminants are removed and placed on a conveyor system. This sorts the containers and ejects the respective types at the delivery stitches assigned to them. The containers arriving there are dirty after they have already gone through multiple handling and storage steps. The position of how the containers arrive at the ends of the delivery stitches is not defined. It can vary in length, width, and rotation. Once there, the containers are gripped, turned, and stacked on pallets by other employees. When a pallet is filled, a lid is placed on it. It is then picked up by a forklift driver, retailed and transported to the goods issue area. This process is depicted in Figure 2.12

2.3.2 Objects

This chapter describes the objects required for material flow in more detail. Those are Containers, Pallets as well as shelves.

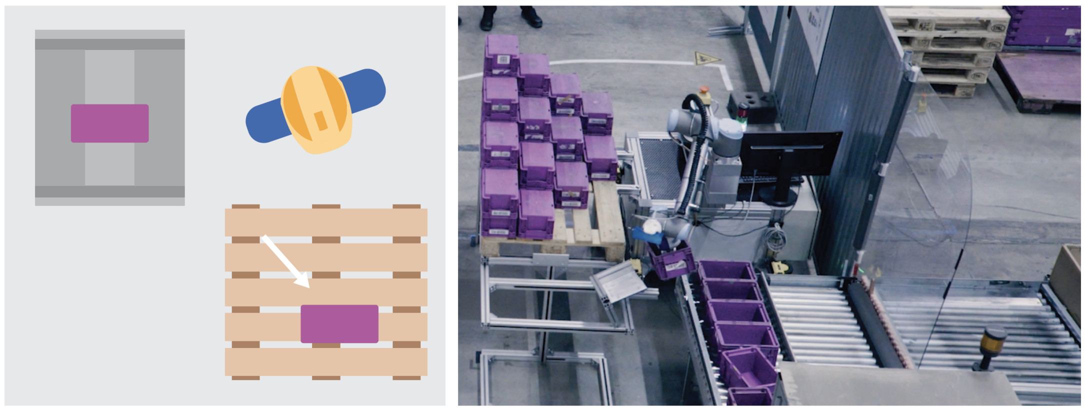


Figure 2.12: Visualization of process and environment: palletizing (own illustration)

Containers

As already mentioned, the components to be assemblied are delivered to the production facilities by the suppliers as complete loading units. A loading unit is the combination of several "load carriers" that are revert to as "containers" in the following, with one loading aid. These can be differentiated by size (small load carriers (container) and large load carriers) and degree of specialization (standard load carriers special load carriers). The main functions of load carriers and the formation of loading units are the protection of materials and the optimization of transport, handling, and storage processes in logistics [Mar11]. The BMW Group currently uses more than 3,000 different containers due to the specific requirements of the components and the material flow [BMW18]. A large number of different special containers reflects the trend towards increasing model diversity in the automotive industry. Special containers are specially developed and constructed for the transport and storage of components with complex part geometry or specific requirements [Klu18]. BMW uses approximately two hundred fifty different special containers per vehicle derivative.

By standardizing container properties, standard containers can be used in all plants and technologies and throughout the supply chain. The specified dimensions are often based on DIN or VDA standards [Klu18]. Unlike special containers, standard containers are not assigned to any specific components and can, therefore, be used flexibly.

The main requirement for the containers from logistics is the damage-free delivery of faultless parts to the installation site [Klu18]. Not only the protective function of the containers plays a decisive role, but also the quality of the load carriers. Containers must absorb static and dynamic forces along the logistics chain [Klu18]). In almost all handling processes, this is reflected in the quality of the containers. Furthermore, the components to be transported have an immense influence on the quality of the container. If oily parts have to be delivered in a container, residues remain inside the container [BMW18] after removal at the production site. If a container passes through the container circuit several times, heavy soiling and wear can occur. These described phenomena lead to some properties which are of particular importance in connection with the automation of these processes. They are addressed in the following.

Optically, the high visual variance is particularly relevant. Depending on the vehicle derivatives to be produced, up to 400 different container types can be observed per plant. These differ mainly in terms of color, dimensions, and the material used. Figure 2.13 shows a small part of this planned variance. Due to the harsh industrial conditions prevailing in logistics, such as dust or other contamination, the effect of high physical forces due to frequent handling steps and the continuous loading and unloading of the containers so that humans can identify them, an unplanned optical variance can be observed in the plants in addition to the planned variance. This is shown in Figure 2.14.

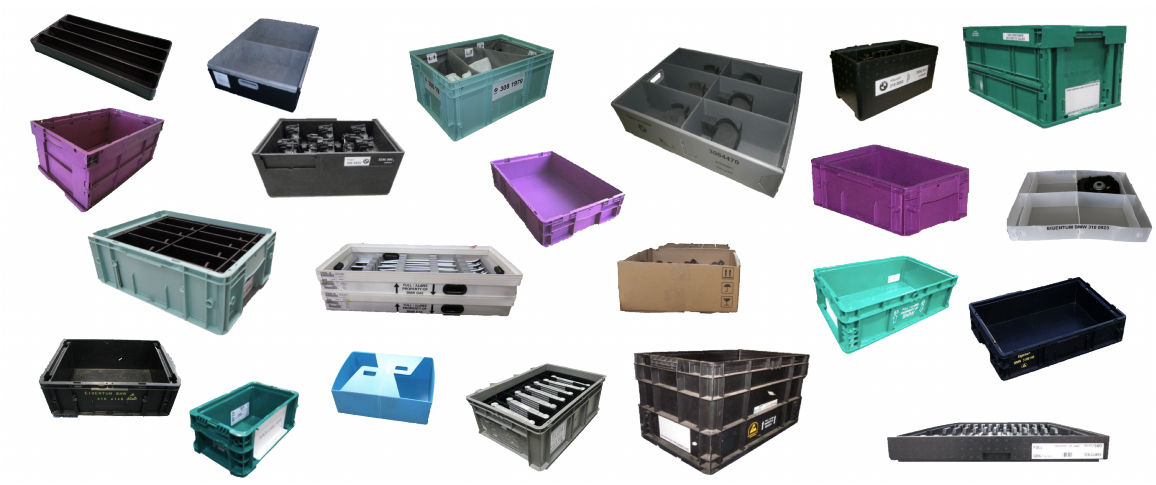


Figure 2.13: Visualization of the planned container variety (own illustration)

Another characteristic feature of containers is the different distribution of containers in the material flow caused by standardization efforts. This is shown in Figure 2.15. This illustration also shows the plant dependency of the containers used. This is actively caused by the derivatives to be produced. Geographical aspects also play a role if, for example, in overseas plants, a large proportion of the materials have to be delivered by sea freight.

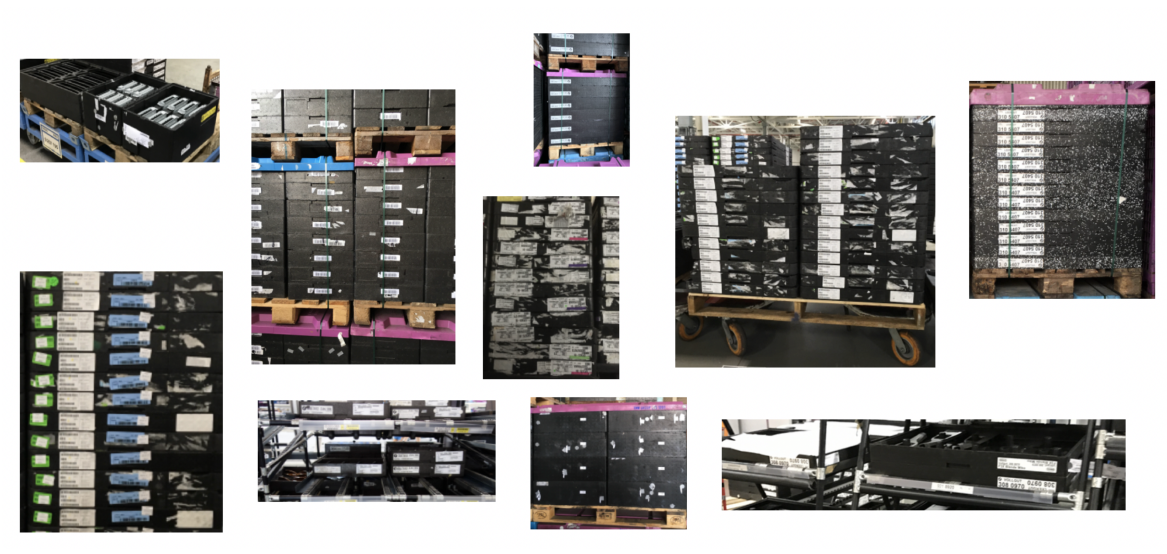


Figure 2.14: Different optical appearances of only one container type (unplanned variety) (own illustration)

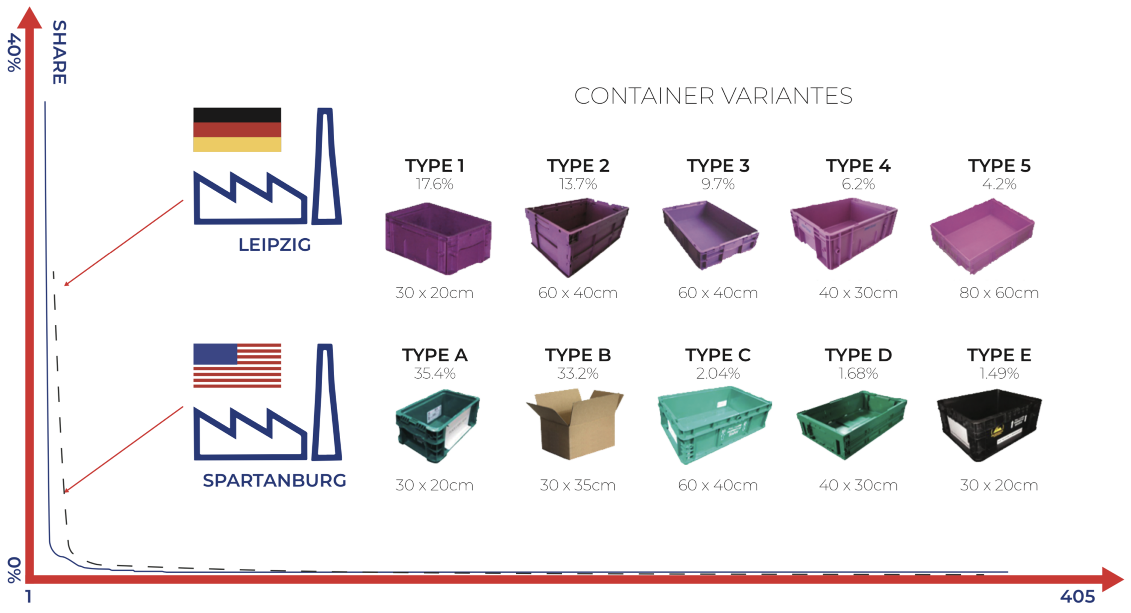


Figure 2.15: Container distribution in the entire materialflow in different plants (own illustration)

Further Objects in Logistics

For the interaction with the containers, objects that are related to the containers can also be relevant. For example, the pallets (Figure 2.16) on which they are delivered or the shelves in which the transport takes place on the tugger train or the provision on the assembly line.

The pallets are standardized and thus have uniform dimensions. However, since the pallets are exposed to the same influences already mentioned, a sizeable unplanned variance in the material flow can be observed.

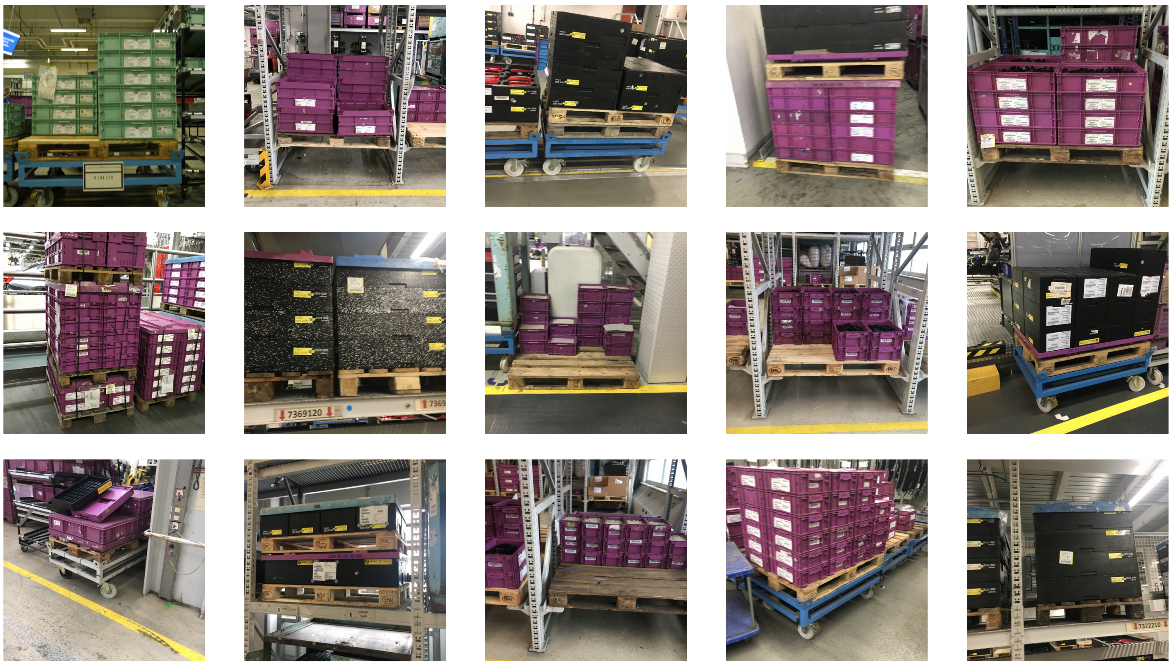


Figure 2.16: Optical pallet variety (own illustration)

Tugger train racks (Figure 2.17) are uniformly designed in most plants to allow interaction with automated conveyor technologies and the use of uniform tugger train trailers. In smaller plants, which do not have such conveyor systems, other racks can be found as well. As with pallets, the unplanned variance is also high for tugger train racking.

So-called provision racks (Figure 2.18) are used to provide the containers on the assembly line. These are individually adapted to the materials to be installed at the assembly cycle. They consist of a kind of modular system, which allows the modular adaptation of these shelves. Planned and unplanned variance can therefore also be observed here.

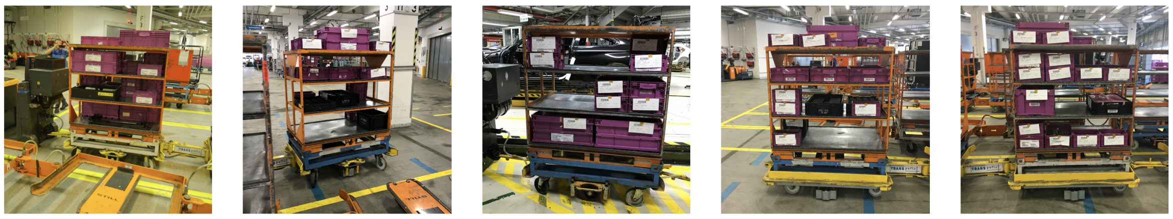


Figure 2.17: Optical tugger train shelf variety (own illustration)

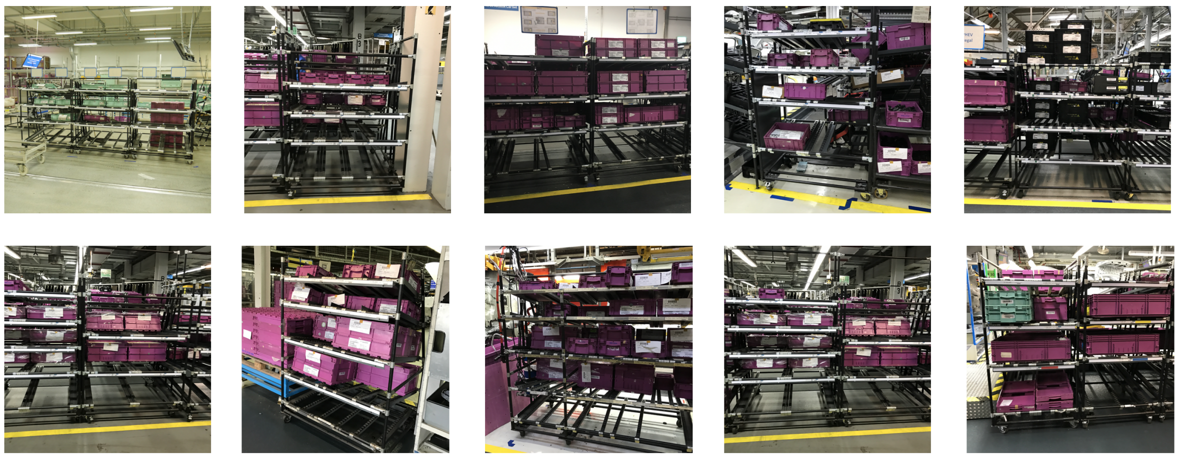


Figure 2.18: Optical shelves variety (own illustration)

2.4 Conclusion

In this chapter, the basis for understanding further work was laid. For this purpose, the basics of robotics were first explained based on definitions and definitions. In addition to the explanation of possible technical manifestations of robots, this includes the description of their areas of application and the necessary process-related framework conditions. Typical applications include car body construction, paint shops, quality control, and the handling of unrecognized objects.

Subsequently, the logistics, the processes running therein as well as the objects necessary for process execution were described, since these as well as their framework conditions significantly influence the requirements for the automation solution to be designed within the scope of this work. The derivation of the specific requirements, as well as the subsequent analysis of the need for action and the identification of possible solutions for logistic robotic solutions, follows in the next chapter.

3 Logistics Robotics

Following on from the theoretical descriptions of classical automation processes and handling steps in intralogistics, this chapter first derives the need for action that is necessary for the automation of these handling steps. For this purpose, the process characteristics are compared with each other, and the requirements for logistics robots are derived from the differences. Subsequently, conceived solutions are presented. First, the designed hardware and software structures are described.

3.1 Need for Action

This subchapter serves to derive the necessary need for action. In the first step, the forms for the automation of significant process characteristics of classical automation processes will be discussed. The logistical handling steps to be automated are then compared with these forms.

3.1.1 Process Characteristics

As already mentioned, the need for action necessary for the automation of the logistic handling steps is derived from the discrepancy of the characteristic forms of significant process properties. In this subchapter, the fundamental process properties are briefly explained.

Variance

The variance indicates the extent to which successive process executions can differ [RS02]. Such deviations are usually caused by a large number of objects to be handled. For example, in welding processes, the variance can be increased by different component variants to be joined or in quality controls by different product derivatives to be tested. In logistic processes, this is expressed, for example, by different types of containers. This property is rated low (1), medium (3), high (5).

Changeability

The process characteristic changeability provides information about the temporal changeability of the overall process. Such changes can be caused by new components to be joined, new product derivatives, new containers, or changes in the process infrastructure [Nei12]. This characteristic is evaluated about the expected frequency of such changes as rare (1), frequent (3), and very frequent (5).

Environmental Influences

The parameter "environmental influences" describes how strongly a process is distinguished from the dominant environmental influences. Such environmental influences include different, non-uniform lighting situations, contamination by upstream processes or other process participants, and possible differences concerning ambient temperatures. In the evaluation, a distinction is made between the most extensive decoupling of environmental influences (1), the presence (3) and a strong process influence (5) by such framework conditions.

Interference of upstream Processes

Processes are always embedded in other processes. As a result, each process has further upstream or downstream processes. Analogous to the environmental influences already mentioned in Chapter 2, this can result in further influences due to errors or other unique features of the predecessor processes. A distinction is made here between low (1), medium (3), and high (5) influence.

Degrees of Freedom

The process characteristic "degrees of freedom" indicates the number of process variables to which a process in its execution form is firmly defined or variably manageable. The number of possible degrees of freedom can be limited by fixed defined workpiece holders as well as fixed discharge points of the final process product. This property is evaluated with few (1), medium (3), and many (5) remaining degrees of freedom.

Human

As final process characteristics, the possible influence of human intervention and interaction within the process environment is targeted here. This is an important influencing factor, especially for automation processes, as it significantly determines the safety concepts to be used. For example, robots and humans must be completely separated from each other. If this is not possible, sophisticated sensor technology must be used to protect humans. This complexity is rated low (1), medium (3), high (5).

3.1.2 Characteristics of classical Automation Processes

In this subchapter, the automation applications of classical industrial robotics described in Chapter 2 are analyzed concerning the process parameters.

In recent years, the number of robotics used in industry has risen sharply [TZW16]. Through adapted structures and controlled framework conditions, it was possible to automate a multitude of unergonomic or dangerous activities. Irrespective of the precise application, such automation solutions, driven by limitations of the technology used and best practice experience in the respective fields, show some similarities. Thus, such automated processes show a low variance as well as a low variability of those [Hau13]. In the automotive industry, for example, this is reflected in the fact that one assembly group per robot is processed in a production line in car body construction. This is unchangeable until the next derivative is released. Given the product development cycles currently prevailing in the automotive industry, this period amounts to several years. In order to achieve stable automation, interference with the environment and from upstream processes was also reduced to a minimum. The stability of the automated processes is further increased by fixed defined lighting and complete quality controls, which ensure that no faulty components negatively influence the subsequent processes. This is also accompanied by the restriction of the number of possible degrees of freedom. Besides, such robot systems are separated from regular production operations by safety fences, thus ensuring human safety.

3.1.3 Characteristics of Logistics Processes

In contrast to classical automation processes, logistics processes are strongly characterized by a high degree of variance, necessary flexibility, and dynamic changes in interaction with challenging influences [Sch08]. As shown in Chapter 2, the variance in logistical processes is very large due to the diversity in container types. In contrast to car body construction, for example, where only one product family is manufactured per production line, all components of a vehicle plant pass through the same logistical processes [Arn+17]. This shortens the time between vehicle restarts, which in turn leads to a high degree of variability in the processes. Furthermore, logistic processes show a strong influence of changing environmental conditions. As explained in Chapter 2 in the course of the unplanned variance, this manifests itself in other lighting situations up to dirty or damaged containers. Due to the low isolation of the logistic processes as well as the generally very high proportion of manual process steps, the framework conditions necessary for the execution of the subsequent processes cannot always be adhered to. For example, labels that have not been approved have a considerable influence on the subsequent process steps. Since logistics continuously adapts to the product spectrum and the percentage frequency of the individual product variants to be produced in it, logistic processes have many degrees of freedom. This is expressed, for example, in the fact that all shelves are on wheels, or there is no fixed allocation of containers to shelves. Due to this necessary flexibility and adaptability as well as the interfaces between logistics and assembly, it is not possible to separate the processes from the inherent employees by fences, for example.

3.1.4 Conclusion

The following Table 3.1 compares these described forms of expression of the explained process characteristics.

It is evident that due to the partly considerable deviations regarding variance, variability, interferences, degrees of freedom as well as man-machine interaction, the simple or adapted use of standard automation technology for the automation of logistic processes is not feasible. Much more intelligent systems must be developed, which can react autonomously to changes in the processes and manage the challenges appearing within.

Table 3.1: Comparison of process properties of logistics and already automated processes

Process Characteristics	**class. Automatization**	**Logistics**
Variance	1	5
Changeability	1	5
Environmental Influences	1	5
Interferences of upstream Processes	3	3
Degrees of Freedom	1	5
Man Machine Collaboration	1	3

3.2 Robot Concept

The analysis of the respective handling steps reveals that classic industrial robots can not be deployed in logistics. Also, the diversity of the requirements in the respective handling steps, such as movement space or weights to be handled, imposes the design of specialized robot types for each area of application. Despite the individualization efforts for the automation of the respective handling steps, it is also necessary to fulfill robot-spanning requirements. This chapter introduces the hardware concept of logistics robots. Since the most significant influence on the robot concepts used is the required payload, the following explanations are structured after a general introduction to robot concepts for full and empty containers.

A robot consists of different clusters. For each cluster, the solution concept, the hardware used, and the necessary software are discussed in the following.

Three modules are required for the robot-based execution of generic tasks: Perception, decision, and execution [Gsp+12] [Cut06]. For use in logistics environments, these three modules are additionally supplemented by the two additional modules Safety and Interaction in this doctoral thesis. The first includes all measures that are necessary to operate the robot safely in a human environment. The second ensures that the robot is intuitively suitable for cooperation with the employees. On the one hand, the robot must be able to process information and commands from users and, on the other, it must be able to make information available to humans. The mentioned sections, as well as the chapters there, dealt with, are shown in Figure 3.1.

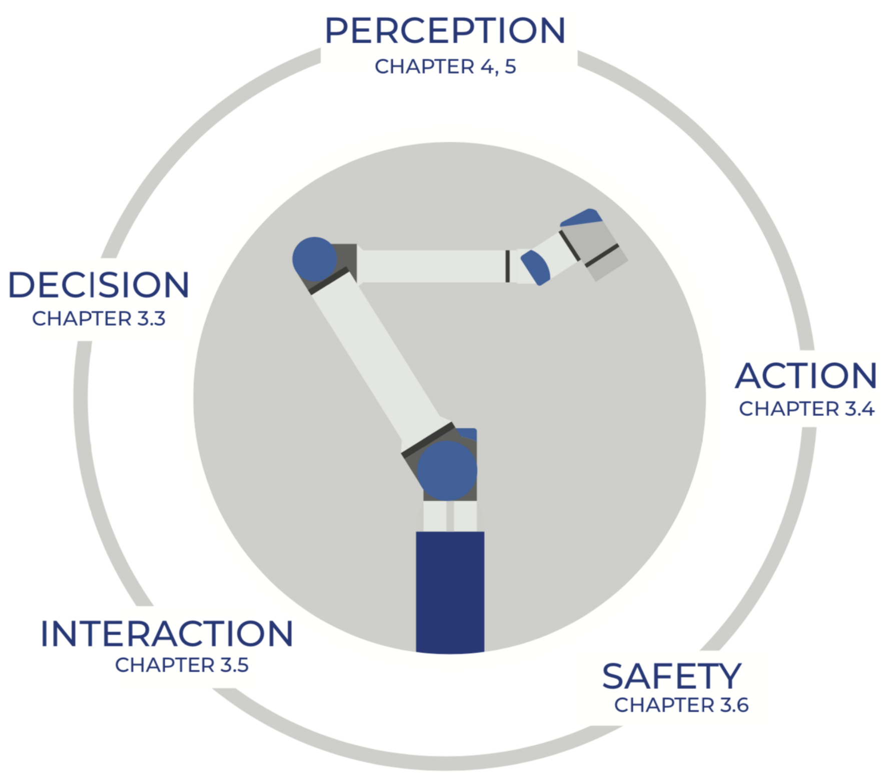


Figure 3.1: Overview about robot concepts and outline for the central part of this doctoral thesis (own illustration)

3.3 Decision

The general term "decision" refers to all decisions that affect process execution of the robot [Cut06]. It integrates the functions to be performed and divides the necessary movement units, which are described in the following subchapter, into the required sequence for each process. Since the logistics processes are not completely predictable and various types of disturbance influences must be expected, this module additionally differentiates between the behavior in the normal case and the behavior in the special cases. The latter can be used to try to provide the robots strategies or possible solutions to free themselves from special situations. As a representative of a large number of possible such scenarios, the handling of damaged containers should be mentioned at this point. It is represented in the respective illustration also "Problem". If the requirements of the regular modules cannot be fulfilled, this state gets called. In there, the decision about error handling strategies is applied. In order to keep the illustrations mentioned above simpler, the connections from all the other states to the "Problem-State" have been removed.

Despite the different robot concepts used, particular attention was paid on the one hand to ensuring uniform behavior to increase employee acceptance of the robots and on the other hand to exploiting synergies to reduce the scope of development. Nevertheless, the behavior patterns of the robots differ significantly depending on the application and the hardware. Therefore, the behavior of the palletizing, depalletizing, and providing robots is described separately below.

3.3.1 Palletizing

This subchapter introduces the sequence control of the palletizing robot. For this purpose, the rough concept shown in Figure 3.2 is first visualized, before diving more profound in the building blocks are explained in more detail.

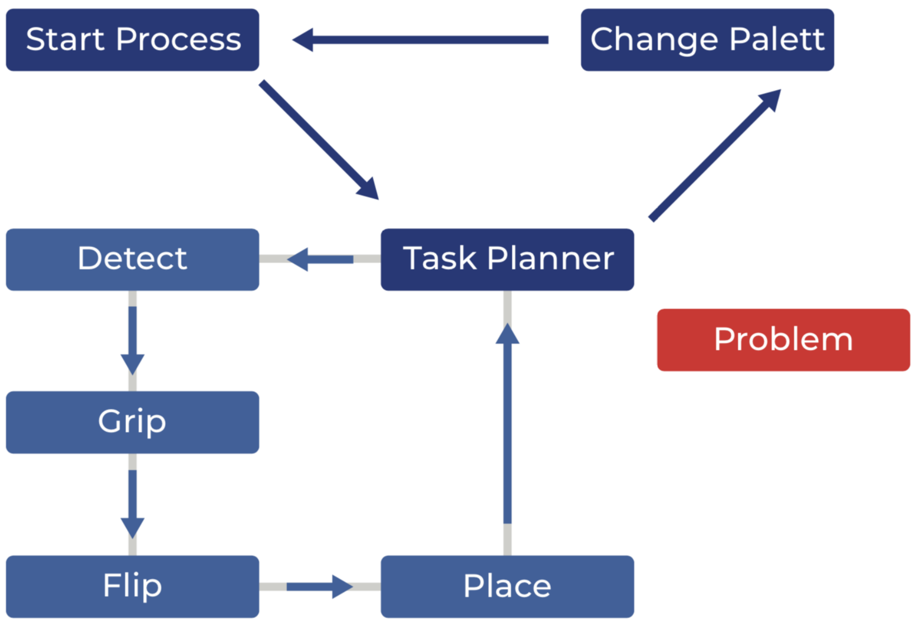


Figure 3.2: Simplified state machine of a palletizing robot (own illustration)

At the beginning of the first state, the system is first paused until all the system components required for operation are ready to start. This includes safety, robot drivers, and the interface to the user. As soon as this is the case, it is checked whether there are errors in those components and whether the general conditions necessary for the process start (e.g., no disturbance of the safety zone) are fulfilled. After successful verification, the “Play-Button” in the interface is activated, and the robot waits for the initial start signal from the user. As soon as this input has been made, the robot moves to its starting position, and the next modular sequence module is called.

If an error occurs in this execution of the steps just described, a so-called “Problem State” is called, in which decisions are made regarding the further handling of the problem. In this specific case, this could be caused by a hardware error of the arm, for instance. This results in the robot not being able to move to its intended starting position. The robot detects the error and visualizes it. It will remain there until the problem is solved. If this is the case, the robot returns to the initial state. The steps already described are now carried out again. This logic for dealing with problems is identically integrated into all states. However, depending on the generic function module, different error patterns, and thus, different possible solutions are conceivable. Similarly, when the containers are gripped; for example, these processes are repeated until an initially defined limit value is reached. If an error is caused three times in the concrete example of gripping, the gripping module is aborted, and the problem status changes over.

Following the successfully executed state “Start Process”, the state “Task Planner” is created. This can be called either at the beginning after the start of the robot or after each palletized container. On the one hand, it decides how the process is to be carried out. This means whether the pallet has to be changed because it is full, or whether additional containers can be palletized. On the other hand, in this step, all parameters are defined which are necessary for the following process steps of the robot execution. This includes, for example, the storage positions of the containers on the pallet and a corresponding adaptation of the robot trajectory to be carried out later.

If further containers can be stacked, the next step is to call the detection cluster. The robot arm is first brought into a position in which it has a good view of the containers located on the conveyor system. As soon as this position is reached, the gripping point determination is called up. Their exact procedures are described in more detail in the following chapters. If the gripping point is determined successfully, the “Grip State” is executed. If the determination is not successful, a separate procedure exists. If, for example, an error occurs during gripping point determination, it does not make sense to immediately start the process execution from the beginning, as this would unnecessarily increase the processing time of the robot. Therefore, in such cases, the determination of the gripping point under identical conditions is repeated. If this fails again, the robot arm is moved to a new recognition position. This makes it possible to repeat the process of determining the gripping point under changed conditions.

Following the determination of the gripping point, the container is picked up with a suction pad in the upcoming states. The necessary movement to be carried out is divided into several steps. First of all, the gripper, which is attached to the Tool Center Point (TCP) of the robot arm, is brought centrally over the container. Next, the rotation of the suction pad is adapted to that of the container before the gripper is lowered to hold the container. This movement is carried out quickly at the beginning. From a certain height, however, this speed is reduced. Due to the mobility of the conveyor technology, it is possible that during the gripping process, the container selected for a pick-up is moved by sliding containers. In order to avoid damaging collisions, the current value at the TCP is measured parallel to the slowed robot movement. If a limit value is exceeded, this indicates an alleged crash. The movement is then aborted, and the robot jumps back to the "Detection State". The same behavior is initiated if the container still cannot be picked up without a collision. The reasons for this are, for example, labels still in the containers or other residues from the assembly. Such a weak grip can be identified by evaluating the pressure sensor installed in the gripper. This takes place as soon as the specified gripping point has been reached.

As soon as the gripping process can be classified as successful based on the evaluation of the sensor values of the gripper, the "Flip State"is started. On the one hand, this serves to rotate the container by 180 and thus meet the requirements of logistics according to which empties containers must always be rotated for better differentiation. On the other hand, this intermediate step serves to compensate for possible inaccuracies in the gripping process by means of an additional centering step with a defined position. Such inaccuracies can result, for example, from minimal container movements, which shift the correctly determined gripping point without causing a collision between the container edge and the gripper. The flip process consists of a combination of the modules described during gripping for moving the arm as well as for controlling the gripper and evaluating its sensor. This reuse of the necessary software modules is also accompanied by the transfer of the interception steps to be executed in the event of an error.

Once the container has been rotated 180 and picked up again by the suction pad, the "Place State" is initiated. For this purpose, a central starting position above the pallet is first approached. This intermediate step is necessary to categorically exclude possible collisions with the containers already placed on the pallet. If this is reached, the delivery position is approached, and the gripper is deactivated. At this point, the internal container counter is also increased and transferred to the recurring step "Task Planner".

3.3.2 Depalletizing

Due to the already mentioned similarities between the different robot applications, it is also evident in a comparison of the Figure 3.2 and Figure 3.3 that the execution patterns are very similar in their basic structure. In this section on the functional sequence control of the depalletizing robot, therefore, only the significant differences are dealt with for reasons of clarity.

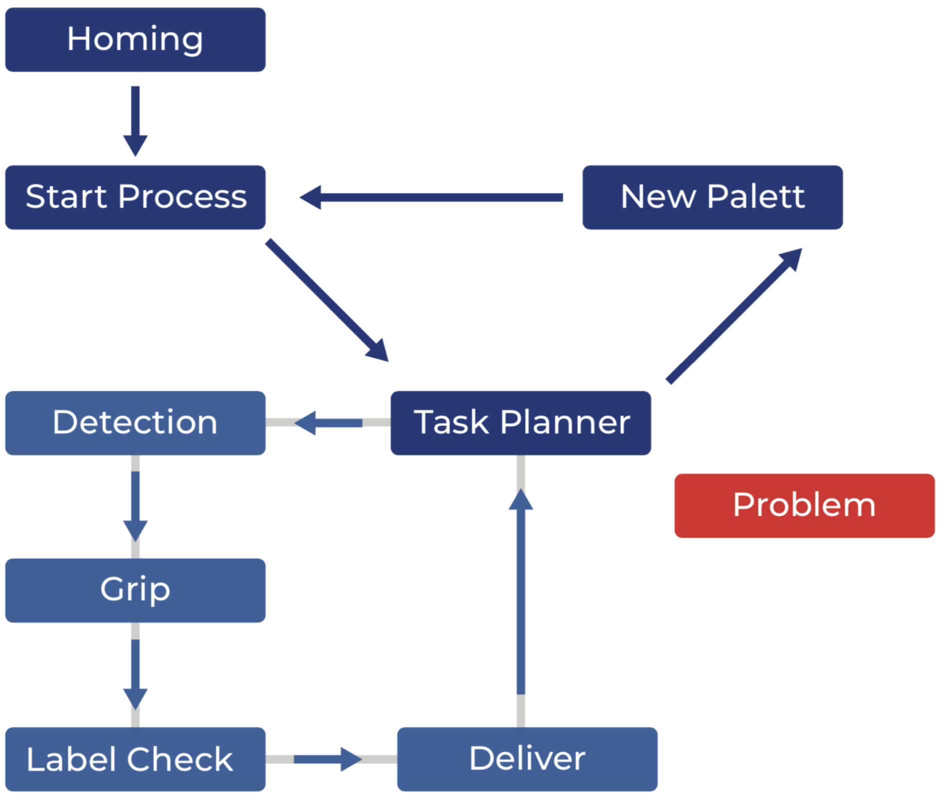


Figure 3.3: Simplified state machine of a depalletizing robot (own illustration)

The first significant difference is the “Homing State”. This is necessary because the axes used to move the robot, which is described in detail in Chapter 3.4.2 must be moved to the initial output position before their first movement [Hei16]. Reaching this position is detected by physical sensors. Starting from the coordinates of this starting position, all positions to be approached in the process sequence can be unambiguously referenced.

In the subsequent “Start Process” state, the framework conditions necessary for process execution are checked and, when the robot is restarted, the initial start procedure is waited for using an input on the user interface. If these criteria are fulfilled, the “Task Planner State” is called. Similar to palletizing, this has a counting function that symbolizes the depalletizing process and, depending on the container number, decides for the next coordinates or initiates the process for changing the pallet. Besides, a unique process exists in this module, which is called up with each new pallet or with the restart of the robot after problems or planned switch-off times. The gripper is first moved to an initial position. A picture is taken there, and information about the pallet, its container, and the situation, in general, is determined via the Perception Algorithm described later. This information can be used to determine the total number of containers. The positions to be approached for the following process steps are then derived from this.

Based on these findings, the depalletizing process itself can now be carried out. To do this, the gripping point must first be determined for each container. There, the container can be picked up by a suction gripper and pulled into the robot. Since these modules are functionally identical to the corresponding steps for palletizing, we will not go into more detail here.

A special step which is necessary for the correct delivery of the container to the subsequent conveyor is the "Label Check State". In the following conveyor technology, the containers must be identified in order to be able to store them at the intended storage points. The information required for this can be found on the labels, which are attached to the outside of the containers by the suppliers. As these are only on one side of the container, a decision must be made on which side the label is on before the container can be gripped, based on the captured images. This is also realized via object recognition. Depending on the label position, the container is then rotated clockwise or counterclockwise after it has been fed into the robot. As soon as the required target position has been reached, it is handed over to the subsequent conveyor system in the "Deliver State".

As soon as the container has been successfully transferred, the Task Planner is called, and the process starts again.

3.3.3 Assembly Line Supply

The process sequence to be carried out by the robot is very similar to the depalletizing process (see Figure 3.4). The only significant difference, however, is that an additional degree of freedom, namely the position of the robot in space, must be determined.

The process flow also begins with the initialization of the actuators. In addition to the linear axes, this also includes the omnidirectional driving modules.

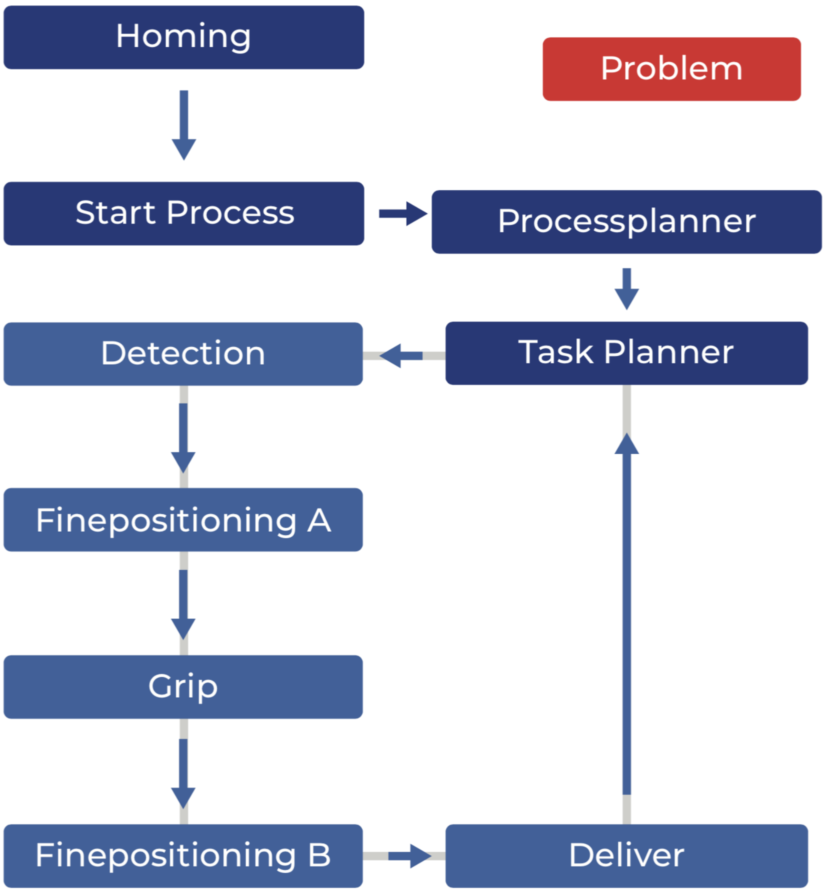


Figure 3.4: Simplified state machine of a provisioning robot (own illustration)

While the functional activities in “Start Process” are identical to the depalletizing processes, there is a clear difference in the following “Processplanner State”. Due to the already mentioned mobility of the robot in space and the associated additional degrees of freedom, not only must the corresponding containers be located for a cleanly executed provisioning process, but the robot must first be moved to the required position in space. For this, it requires information from external Information Technology (IT) systems, which transmit the position of the tugger train and the destination delivery position of the container. Based on this, the robot can move to its initial position.

Once there, the “Task Planner” known from the previous section can determine the information necessary for further processing. Since the further procedure is very similar to that of depalletizing, only the deviations, namely the added states “Finepositioning A” and “Finepositioning B”, are dealt subsequently with here. While in the other two processes, palletizing and depalletizing, it could be assumed that the suction pads could be brought to the specified positions sufficiently precisely for a stable process sequence by the actuators installed in each case, this posed a challenge when it came to assembly line supply. The additional wheelsets reduce the precision and repeatability of movement. This is since the distance traveled per wheel revolution varies depending on the robot load and ground conditions. Therefore, the process flow was adapted.

Based on the information initially provided, a room position in front of a tugger train rack is approached. Once there, the robot is aligned based on the results of the "Detection" and the achievement of the target values is checked. If a deviation between robot angle and shelf angle occurs, the target position is adjusted based on these delta values until the resulting differences permit stable process execution. The same is also done when the containers are delivered to a supply rack.

3.3.4 Software Implementation

The logic described in the previous subchapter was implemented in Robot Operating System (ROS) by SMACH. This is a task-level architecture for creating complex robot behavior. At its core, SMACH is a ROS-independent Python library to build hierarchical state machines. [ROS14]

This defines individual states in which specific functional code modules are executed as soon as the respective state is called. In order to benefit from the synergies of the described systems and generic development synergies, for example, there is only one gripping state in which the gripper is controlled. Depending on the modeled process flow, this state is usually called several times afterward. The code, which is executed within the respective states, depending on the respective modules, is described in more detail in the following.

3.4 Action

This subchapter describes the purpose and functionality of the action module. The activities to be performed by the robot can be further subdivided into gripping and moving.

3.4.1 Gripping

While how the robot movements are implemented can differ considerably, it has been possible to keep the grippers modular and generic. As described in Chapter 2, they have to cope with a high variance of different objects concerning geometry and material in their respective processes. Since the gripper cannot be changed for every type of container, a concept had to be found, which made it possible to handle all objects, even considering their planned and unplanned variety. Therefore, the decision was made in favor of vacuum suction pads as they are shown in Figure 3.5. These are primarily independent of geometric features on the container surfaces for form-fit receptacles. The only prerequisite for a solid handle is the presence of a correspondingly large free area on the container surface. With sufficient suction power, it is therefore also possible to grip polystyrene or cardboard in addition to plastic. [PIT16]

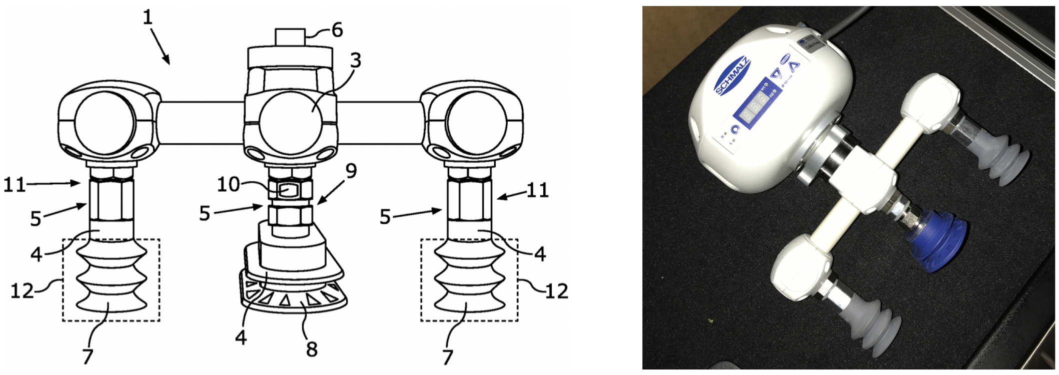


Figure 3.5: Patented draft (left) and realized version of the robots vacuumgripper (right) (own illustration)

3.4.2 Moving

The module most visible to the outside is the execution of the planned movements. Due to the already mentioned differences in the three handling tasks to be automated, different execution concepts were developed. This subchapter is therefore divided into the description full and empty containers. Despite the significant differences, attention was paid to the use of synergies during development. Since the robots are supposed to provide a financial benefit in the price competitive logistics sector, attention was also paid to the use of low-cost components. However, the robots presented in this publication are still prototypes. The serial development of the final robot versions, as well as their deployment into the individual plants, are currently forced - but is not in the focus of this doctoral thesis.

Empty Containers

The limiting variables in all handling steps are the payload, degrees of freedom, and reach of the robot. Concerning the application of palletizing empty container, this means the ability to handle up to 5 kg. This payload must then be able to be placed at all positions of the Euro pallets next to the robots. For this, a range of 1.3 m is required. Since the container must be rotated 180 degrees after pick-up, such freedom of movement must also be provided.

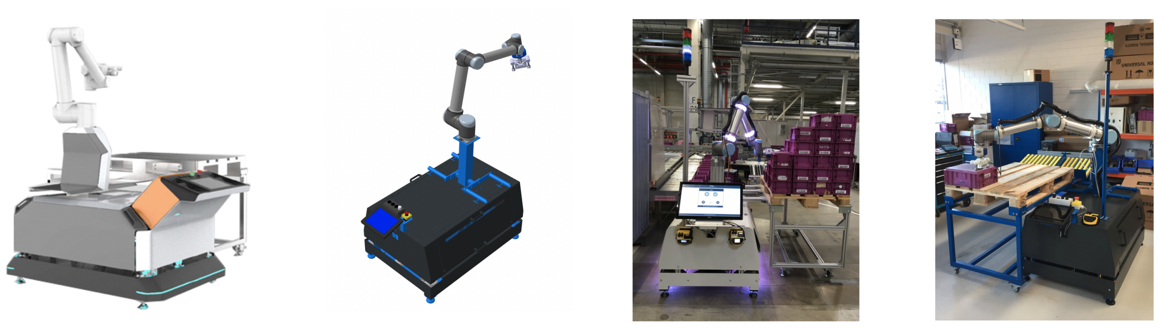
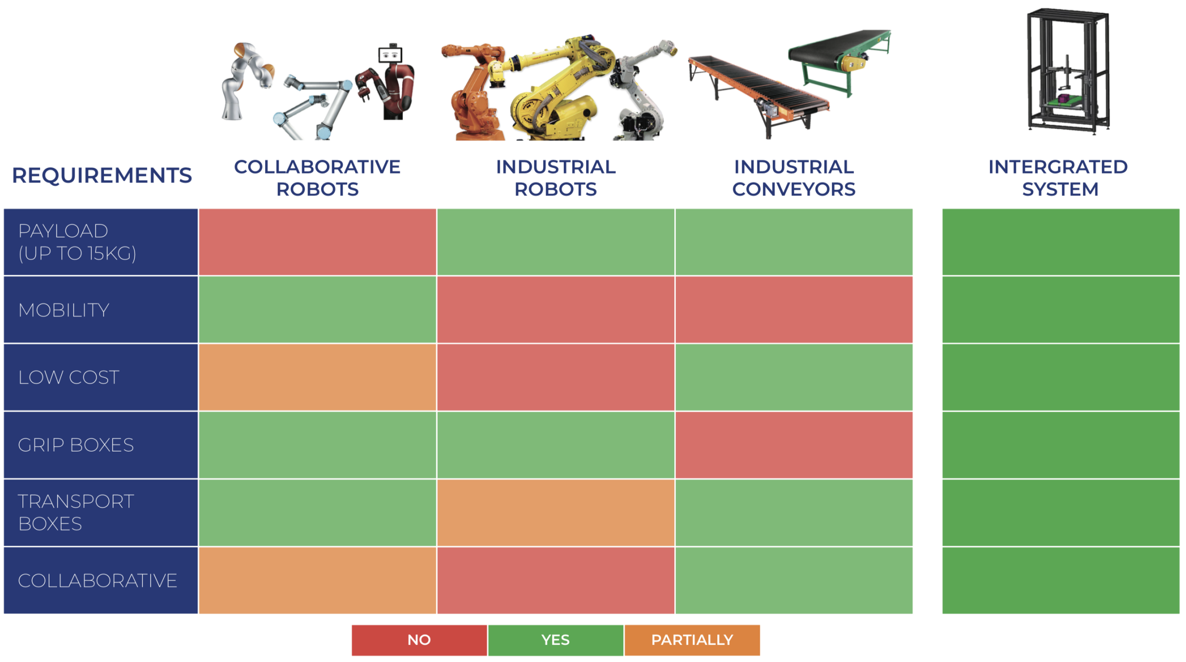


Figure 3.6: Robot for palletizing: draft, construction, prototype, serial version (own illustration)

The following palletizing concept was chosen to meet these requirements. A Universal Robot (UR) 10 from Universal Robots was selected as a robot arm. This best meets the requirements in terms of cost, range, and payload [UR18]. This is mounted on a housing. This serves on the one hand for stabilization and on the other hand, contains necessary components such as control computers and other peripherals. The pallet frame and a flipboard are firmly connected to the housing. The fixed position of the pallet rack makes it easier to use the entire system since the relevant storage positions do not have to be taught-in each time. Also, the robust process design is guaranteed, since only at the defined pallet position is it ensured that the robot could reach all required deposit positions. The mentioned flipboard aims in a similar direction. A maximum tolerance of 2 mm is required to enable stable stacking in the series process. This can be ensured via the clipboard step. As already mentioned in the explanation of the robot program sequence in the previous subchapter, the accuracy of the object recognition in the first step could only be achieved with very high effort, and a higher tolerance is permitted with this object acquisition. The object is then dropped into the clipboard, from where it can be very precisely captured, rotated, and placed on the palette. Essential for this hardware design are the two possible degrees of freedom of the palette on the left and right of the housing.

Full Containers

Even though there are many different available robot models as they have already been shown in this doctoral thesis, the handling of full containers under these environmental influences remains unsolved.



REQUIREMENTS	COLLABORATIVE ROBOTS	INDUSTRIAL ROBOTS	INDUSTRIAL CONVEYORS	INTERGRATED SYSTEM
PAYLOAD (UP TO 15KG)	🟥	🟩	🟩	🟩
MOBILITY	🟩	🟥	🟥	🟩
LOW COST	🟧	🟥	🟩	🟩
GRIP BOXES	🟩	🟩	🟥	🟩
TRANSPORT BOXES	🟩	🟧	🟩	🟩
COLLABORATIVE	🟧	🟥	🟩	🟩

🟥	🟩	🟧
NO	YES	PARTIALLY

Figure 3.7: Conceptualization of a robot concept for handling full containers (own illustration)

The use of HRC-capable robot arms is not possible when handling full loads because the payload is too low, as containers weighing up to 15 kg are handled there. The use of industrial robots, as presented in Chapter 2 is also not possible. These would fulfill the requirements regarding the necessary load capacity but are neither suitable for collaborative use with humans nor inexpensive scaling in an industrial environment. Therefore unique robots were developed in order to benefit from automation solutions in logistics, especially in unergonomic tasks involving full containers handling. These use elements of robots as well as standard plant technology. This leads to the concepts that are shown in the following Figure 3.8.

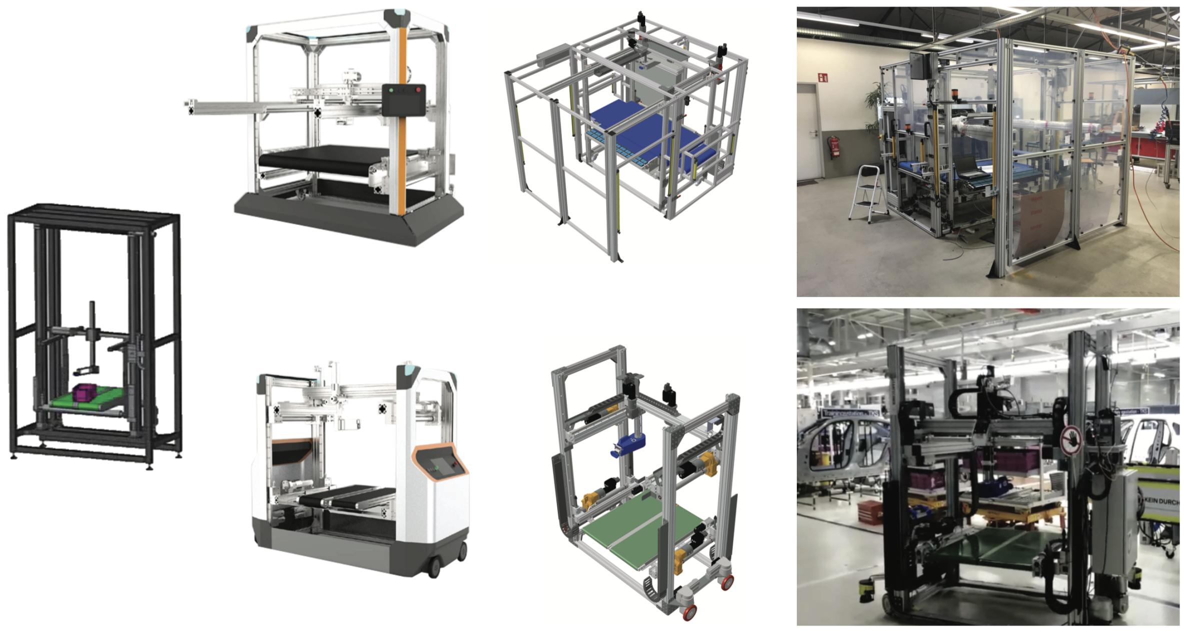


Figure 3.8: Robot for handling full containers: draft, construction, prototype, serial version (own illustration)

The core idea behind both concepts is not to move the containers with their 15 kg hanging on the gripper in the air, but to lift them slightly using the grippers used and then pull them over existing objects. These can be represented by shelves, pallets or further rows of containers. From there they are then pulled onto the conveyor belts of the robot. The vertical movements of these belts as well as the x, y, and z movements of the gripper module are realized via linear axes. Thanks to their standardization and widespread application in plant engineering, these are particularly inexpensive and robust. The conveyors used to rotate and move the containers are also standardized industrial components. Depending on the application, the containers can be brought into the target state required for the process and transferred to the downstream stations.

One robot per storage stitch can be used stationary at goods receipt due to the inherently high usage. When the robots are deployed on the assembly line, on the other hand, they have to be transported individually to the intended working area. Since the assembly lines per plant can extend over a length of several kilometers, the supply robots must be able to move independently and mobile to the respective target positions. Omnidirectional driving modules were used to avoid lengthy maneuvers between the tugger train and the provisioning rack, which could be expected through the use of differential drives. Besides, further energy modules were added to the provisioning robot to enable a mobile operation.

3.4.3 Software Implementation

In the software execution of the basic robot movements, a distinction is also made between the two rough hardware concepts, i.e., the handling of empty and full containers.

Since the palletizing of empties in the form of the UR robot arms can be based on existing hardware, the main focus, in this case, is on the implementation of communication between the industrial Personal Computer (PC) and the robot controller. These two components are connected via Ethernet. Using robot drivers provided by UR, it is possible, for example, to transmit target points to the controller. Therein the robot trajectories are calculated from the waypoints, and this determines the joint angles necessary for the respective movements.

The other two robots communicate with the actuators via Controlled Area Network (CAN)-Protocol. The gripping-coordinates to be approached are determined in the industrial PC and then split up and translated into CAN messages. These message blocks are then sent to the actuators to move the respective axes. The various actuators of the different axes are all connected via the CAN network. A special feature to mention at this point is that the height axes are controlled via a so-called master-slave procedure. This means that the target coordinates are only transmitted to one motor. This then gives the procedure path while the second actuator follows. Accidents in which the axes move in different directions or at different axis speeds due to false messages are also avoided, which would lead to irreversible damage to the robot hardware. [Kio+01]

In both robot concepts, not only the control itself is implemented. Besides, a message was created which is published by the actuators - also via CAN - to the industrial PC as soon as the desired target position has been reached. This is necessary to ensure that the robot is in the planned target state before subsequent movements are initiated.

3.5 Perception

As already explained in Chapter 2, dynamics, variance, and flexibility are characteristic of logistic processes. However, in order to ensure robust execution of the automation task assigned to the robots, the correct perception of the object and environment is of great importance.

For stable gripping, it is necessary to determine the pose of the container to be handled precisely. This must be done independently of the influencing factors and the infrastructure surrounding the container. Racks, pallets, and conveyor systems must, therefore, do not influence this localization. Since the detection of objects and the determination of their gripping points must also be independent of environmental influences such as contamination or labeling, the complexity of the perception task for the object and gripping point determination also increases. The solution to this problem is, therefore, the focus of this doctoral thesis. Chapters 4 and 5 deal with this in more detail.

In order to avoid errors during the process execution, in addition to the visual perception described above, further sensors are used. For example, a vacuum sensor is installed in the suction pads of all robots. These can be used for detecting incorrect grips (no pressure build-up), the sudden dropping of objects (rapid pressure drop) and successful container transfers (planned pressure drop). Depending on the check results, further steps can be initiated by the Decision-Modules.

Another important aspect is the creation of the map and the localization of the mobile tape provisioning robot. Based on the distance values to distinctive infrastructure elements such as columns or walls, which can be read out by the built-in laser scanners, the robot can determine its position based on an initially recorded map. In the scope of this thesis, this module will not be discussed further in the doctoral thesis.

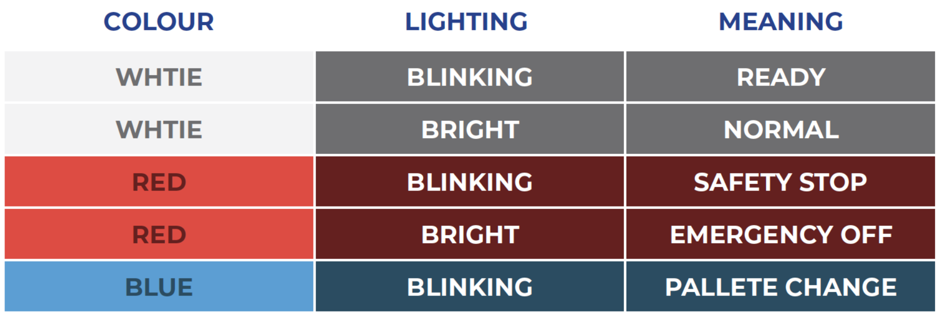
3.6 Interaction

Since all robots work in the environment of humans and are partly dependent on their support, the cluster "Interaction" is a critical module. Despite the hardware and process-related diversity of the robots, human-robot and robot-human communication should function as intuitively and uniformly as possible.

The communication principles used for this purpose are divided into three sub-modules. Light-Emitting Diode (LED) strips are installed on the robots to signal the current operating status to humans and, if necessary, to point out any necessary interaction in the event of errors. The robots each have a touchscreen interface for a quick overview of the current process status and interaction options, for example, in case of a pallet change during palletizing. Since the robots can be distributed over several halls and, in the case of the mobile systems, can occur at different locations each time and therefore the communication media on the robot cannot be viewed, a mobile app is also made available to the process-relevant persons. This notifies them if necessary for malfunctions, provides insights into process statistics or the installed components of the respective systems. These three submodules are explained in more detail below.

3.6.1 LED-Interface

In order to give users a quick indication of the current operating status of the robot, all robots are equipped with an Red Green Blue (RGB)-LED band. The colors and the type of light can be varied. The following states are implemented:



COLOUR	LIGHTING	MEANING
WHTIE	BLINKING	READY
WHTIE	BRIGHT	NORMAL
RED	BLINKING	SAFETY STOP
RED	BRIGHT	EMERGENCY OFF
BLUE	BLINKING	PALLETE CHANGE

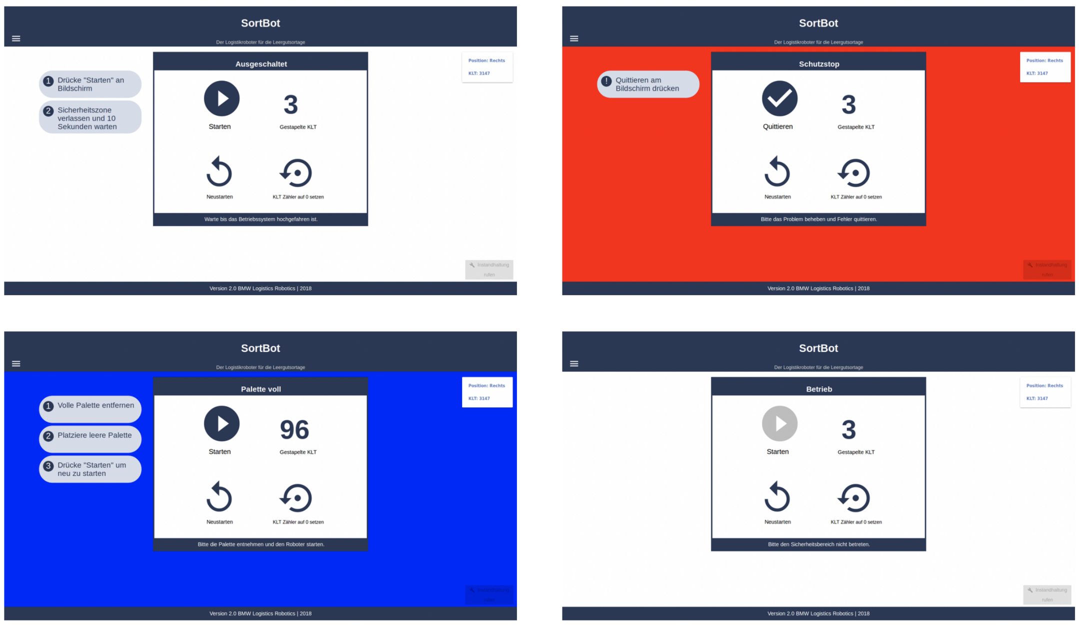
Figure 3.9: Possible robot operating states and their visualization (own illustration)

The type of lighting is decided between flashing and continuously lit. Flashing colors always indicate to the employee that intervention is necessary. Depending on the robot, these states can vary in addition to this basic implementation. These adaptations often depend on the plant and the operator. It will, therefore, not be discussed further here.

The information about the respective state of the robot can be sent from both the Decision-Module and the Safety-Module. Process-driven states, such as the call to change the pallet, are determined in the respective state of the state machine and sent to the Programmable Logic Controller (PLC). This then switches the LEDs accordingly. Safety-critical states (red) are detected in the PLC itself and switched accordingly. These states have the priority and overlay the other state possibilities. How these safety-relevant states are determined is explained in Chapter 3.7.

3.6.2 Robot-Interface

The robot interface provides the necessary interaction options required for operation and initial robot setup. An overview of the different possibilities to visualize robot states is given in Figure 3.10.



1 Drücke "Starten" an Bildschirm
2 Sicherheitszone verlassen und 10 Sekunden warten
Warte bis das Betriebssystem hochgefahren ist.

! Quittieren am Bildschirm drücken
Bitte das Problem beheben und Fehler quittieren.

1 Volle Palette entfernen
2 Platziere leere Palette
3 Drücke "Starten" um neu zu starten
Bitte die Palette entnehmen und den Roboter starten.

Bitte den Sicherheitsbereich nicht betreten.

Figure 3.10: User interface screenshots of the different robotstates (own illustration)

The standard display in operating mode consists of information and status-dependent Interaction-Modules. As information, the respective operating states are displayed analogous to the LED visualization. Depending on the state, prompts for action are displayed as well. The possibility of interaction with the robot is realized comprehensively in the form of a “Start/Stop-Button” as well as in case of malfunctions of an acknowledgment button. Also, there are process-dependent setting options, such as a container counter.

In addition to the standard display, a so-called expert mode is implemented. This offers the possibility for the analysis of process data, such as the successfully gripped containers. Besides, errors that have occurred can be analyzed in greater depth by accessing historical data such as the specific gripping points. It is also possible to adapt the robot to serious environmental changes if required. For example, camera parameters or the height of the conveyor system can be adjusted.

Technically, the interface is implemented as a Web Application [DBG14]. A touch screen captures the user's interaction gestures. Depending on the interaction, the changed variables or prompts such as Start / Stop / Acknowledge are passed on to the respective states of the state machine in the form of ROS parameters.

3.7 Safety

In order to test and operate the robots in the vehicle plants, a wide range of safety conditions must be met. Since these differ greatly about the respective robot type, stationary arm robots, stationary portal robots, and mobile robots will be dealt with individually in the following.

Since the development of the safety concepts themselves is mainly oriented towards the interpretation of existing standards and regulations and therefore has only a small scientific character, the following explanations are kept short. However, they are still important, since the final safety concepts influence the robots presented here both concerning the hardware package and in the area of control technology.

In summary, there are two ways to meet the safety criteria. Either it can be ensured that man and robot are in separate working areas, or the robots must be designed in such a way that no injuries can occur when man and machine touch each other. The therefore relevant documents are the following: [Par14a], [Par06], [Par14b], [10211], [10212], [70111], [13811], [34914] and [15017].

3.7.1 Stationary - Robotarm

Since in the area of empties sorting for the palletizing of empties, continuous human intervention in the process, for example when changing full pallets or removing damaged containers, is necessary, a completely separating mode of operation between employee and robot is not possible.

However, complete elimination of the separate workspaces employing sensitive robotics is just as ineffective. The technical restrictions, such as a substantial reduction of the speed of movement to maintain the standardized force values in collisions, would make it impossible to use the systems in real processes.

The middle course was, therefore chosen. Using a sensor-based separation of the workspaces, it is possible for employees to interact with the robots without any significant loss of process time to be expected in the absence of employees. If a human approaches the robot within a certain radius, this is detected by the built-in laser scanners, and a stop signal is sent to the actuators through a real-time capable, deterministic PLC. When leaving the safety-critical zone, this state is canceled, and the execution of the state machine described in the previous chapter can be continued.

3.7.2 Stationary - Robotportal

In the case of stationary robot gantries, i.e., in the case of the work at hand in the depalletizing area in the goods receiving area, the safety zone around the robot can be dispensed with for geometrical reasons using simple Plexiglas panes on the gantry walls. Since the omission of additional scanners represents a simplification of the control system, this path was chosen.

3.7.3 Mobile

The core principle of the mobile robot protection system used to this extent is a theoretical separation of the gripping and travel functions. If the robot is stationary, the arm or the linear axis system may be moved. The general safety conditions described in the two previous subchapters apply.

If the system moves, the protection takes place similarly as with stationary arms utilizing laser scanners installed in the robot. The unique feature here is that the radius of the safety zone is not only determined based on the arm length and the speed of the TCP, but also the travel speed of the system at the particular time.

3.8 Conclusion

At the beginning of this chapter, the already automated processes in the industrial environment were compared with the processes in logistics to be considered in the context of this doctoral thesis. A comparison of the respective process characteristics thus led to requirements being derived for the respective robots.

In order to implement these criteria, a modular logistics robot concept was subsequently presented and further explained, broken down into the functional sub-areas Decision, Action, Perception, Interaction, and Safety.

Since the encounter of dynamic processes with highly variable objects under aggravated industrial conditions mainly depends on the ability of the systems to warn those objects as well as possible environmental conditions, this system will be developed in the following chapters.

4 Functional Perception Algorithm

This chapter focuses on the conception, development, and implementation of the Perception Algorithm. This algorithm aims to determine the gripping point of the objects relevant to the respective process robustly under industrial conditions and influences. First, the principles used in the literature for this purpose are analyzed. From the algorithms found, the decision for the underlying architecture of the algorithm is then made based on preliminary theoretical considerations. Next, the respective modules of the conceptual algorithm are developed. These are then implemented in a robot for validation purposes.

4.1 Basis Architecture

In this subchapter, a Perception Algorithm for the determination of the gripping point of logistic objects in an industrial environment is developed on a conceptual level based on a more in-depth analysis of the current state of the art. The gripping point is understood to be the point on the surface of the object being searched for that allows the suction pad to be placed entirely on it without containing any error-provoking interfering contours.

The position of the perception robot cycle perception - decision - execution, already shows their importance. Errors that occur in the process are difficult to correct in the subsequent steps. Perception is computationally intensive, complex, and influenced by numerous environmental factors [Hel+09]. Therefore, for a long time, it was only possible to use robots for automation tasks if their working task could be limited as independent of perception as possible. The best examples of this are the welding robots already mentioned in Chapter 2 in car body construction. However, since such perfected working environments do not occur in real life, research in recent years has focused intensively on the topic of "perception" in the course of care, household and picking solutions [Par+07] [Geh+11]. An overview of those is given in the following subchapter.

4.1.1 Related Work

Following a general introduction, the algorithms found are explained in more detail here. Congruent, with its basic architecture, this description is divided into holistic and modular approaches.

General

Research in the field of perception of robots in real environments is still relatively young, and so far, no "silver bullet" has been able to establish itself. This explains the variety of basic approaches to this topic, starting with the sensor input data up to their further processing in the different algorithms. This is illustrated in Figure 4.1 and is described in more detail below. At this point, it should be added that many of the cited sources address only part of the overall perception processing process.

The starting point for this process is real or sometimes even a simulated scene with objects. Using appropriate sensors, the robot can capture these in the form of two- or three-dimensional images (RGB or Red Green Blue Depth (RGB-D)) and point clouds [TSV05] [Sta+14]. These go through a multi-stage process before the final gripping point can be determined.

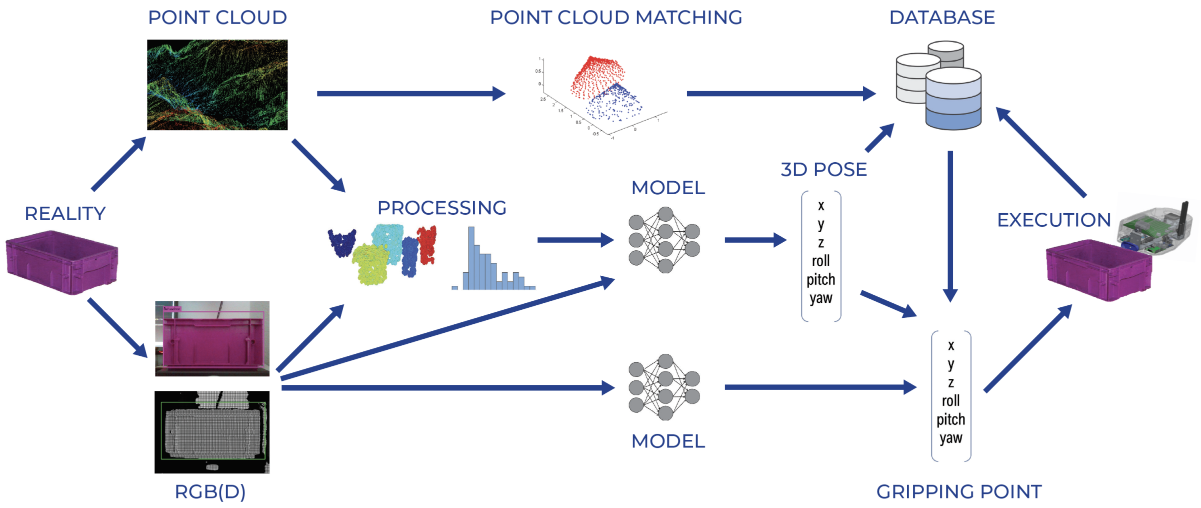


Figure 4.1: Overview about different approaches for grippingpose detection (own illustration)

One significant intermediate result is the determination of the 3D pose of the object. The recorded input data is first segmented before the released objects can be classified [BBV00]. Following on from this, the precise position of the object in space can be estimated. Alternatively, some approaches predict the spatial position based on two-dimensional RGB images through neural networks without the mentioned explicit intermediate stages [Mah+17b] [Mah+17a]. Another method is point cloud matching, which attempts to place three-dimensional models of various objects in the captured point clouds [HY12].

If the 3D pose of an object is known, the gripping point can be determined. A detailed examination of the state of research reveals two approaches: analytical and empirical. In the latter approach, the gripping point is determined by comparing the 3D pose of the object with already stored gripping scenarios from databases. For an analytical determination, the areas available on the segmented object are examined about various criteria necessary for successful grasping. The criteria depend on various influencing factors, such as the surfaces of the objects to be gripped or the type of gripper (finger or suction gripper). [SP17]

Once this gripping point has been determined, the manipulation task can be executed by the robot. The complete gripping process can then be evaluated concerning its success. This generated knowledge can be used either directly for re-training the model or as a further example in a database.

In addition to the subdivided approaches described above, there are further attempts to predict the most suitable gripping point via neural networks or a combination of these. In particular, procedures via reinforcement learning and reinforcement learning combined with imitation learning have received increased attention in recent years. [WO14] [LLS15] [Gu+17]

In order to find the most efficient combination for the deployment scenario described in Chapter 2, these approaches will be examined in more detail in the following subchapter with a view to their advantages and disadvantages. Building on this, the final Vision-Module to be created within the framework of this work will be designed.

4.1.2 Concept

Input Data

The first decision to be made in this step is the selection of the input data to be used. This means with which sensor and in which format the real scene is captured. This decision must be made at the outset, as it limits the sensors to be used and the algorithms available for selection. Since the algorithm aims to determine a gripping pose in three-dimensional space, the recording format used must support depth data. There are, therefore, two options to choose from:

- Point Clouds
- RGBD Images

RGB-D With typical pixel-based images, pixels can be localized by the (x, y) coordinates, and then one can obtain three properties (R, G, B) color, respectively. In the RGB-D image, each (x, y) coordinate corresponds to four properties (depth, R, G, B). The only difference between RGB-D and a point cloud is that in the point cloud the (x, y) coordinates represent the actual values in the real world and not the simple integer values. [Sta+14]

Point Clouds Point clouds can be created from RGB-D images. If one has an RGB-D scan and knows the peculiarities of your scanner camera, the point clouds can be generated from the RGB-D image by merely calculating the real world (x, y) with the peculiarities of your camera. It is called camera calibration. Therefore, the RGB-D image must be raster-aligned while the point cloud is in a thinner structure. [HY12]

Comparison RGB-D images are used for further use within the Perception Algorithm to be developed. There are two main reasons for this. RGB-D images can be processed with relatively little effort due to only one additional channel with depth information and therefore do not place any special additional requirements on the computer systems installed. Furthermore, many algorithms can be reused by this conceptual similarity with classical RGB images. Especially when using neural networks, this is an important prerequisite, since these matrices require input data. Classical processing chains in point clouds, on the other hand, are based on the application of classical computer vision algorithms, which are based on an intimate distinction between object and background.

4.1.3 Algorithm Concept

By pre-selecting the input data to be used, the possible solution space of the Perception Algorithm has also been limited. Thus, there remain two fundamental solutions. The direct prediction of the gripping point by a model, the prediction of the 3D pose of the object with a subsequent determination of the gripping point by an additional algorithm. [Mah+17a] [Mah+17b]

It can be seen from the enumeration of the results of holistic approaches that these tend to be more limited in their overall application due to the enormous scope of consideration. The results presented here can be achieved under the framework conditions presented. A simple adaptation to new conditions is therefore not possible [Mah+17a]. Likewise, the customization options for individualizing the algorithms are minimal or can only be realized with unjustifiably high effort. For example, the integration of a partial algorithm for selecting the next box to be grasped for the process in a neural network that calculates the input data directly to grasp coordinates would not be conducive.

Therefore a modular approach will be used in the following. Individual modules are sought for the core functions for processing the input data up to the output of the gripping point. These have the advantage that, on the one hand, there is a much greater choice of solution and, on the other hand, these have existed for some time, which leads to more reliable statements regarding performance and efficiency. For example, it can also be seen from the top picture that there is considerably more work available to determine the gripping coordinates from the 3D pose of the object than with the holistic approaches described [Zen+17] [SWJ17] [TB18]. According to the described application case, the Perception Algorithm for the autonomization of the depalletizing in the incoming goods department as well as the one for the provisioning robot must fulfill the following partial tasks:

- Identification of the searched objects
- Selection of the next required object in the process execution
- Determination of the gripping point of this object in space.

Accordingly, the Perception Algorithm can be put together as follows (figure 5). In the following subchapters, the selection or adaptation to the use case will be discussed in more detail, belonging to the individual submodules.

4.2 Module 1: Detection

In this section, the module of the Perception Algorithm is created, which recognizes all necessary objects. To this end, this objective is initially set more narrowly. Subsequently, existing algorithm concepts are considered, and preselection for the concept considered in this thesis is made. The basic principles and individual implementation options are then discussed by the preselection made. From the latter, the most suitable implementation option is then selected and adapted to the outlined use cases in the intralogistics material flow.

4.2.1 Target

This module aims to identify and locate objects of the searched classes in the field of vision of the robot. Congruent to the general requirements of the Perception Algorithm, the following aspects must be fulfilled. The Detection-Module has to

- have the necessary precision to provide good input data for the following modules,
- be robust enough to enable stable process execution independent of dynamically changing environmental influences,
- be fast enough in execution so that the robot can meet the cycle time requirements for the respective handling steps, and
- be intuitive and simple enough to be adapted and implemented with manageable manual effort.

For this, a module is required, which outputs the position of the searched objects based on two-dimensional color images.

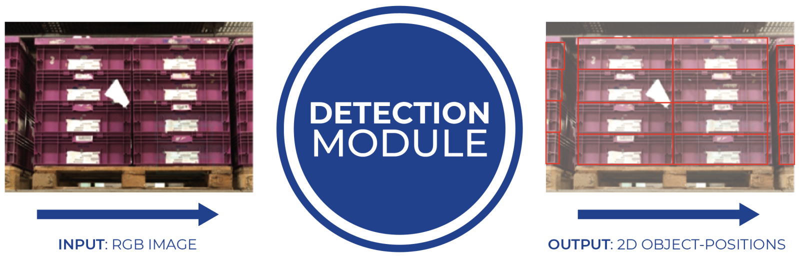


Figure 4.2: Target-specification for the Detection-Module (own illustration)

4.2.2 Related Work

In the field of computer vision, there are various approaches to solving this problem. Four of these stand out and will be examined in more detail below, following on from the generic description of this topic, which is furthermore visually shown in Figure 4.3.

General

The procedures presented here generally consist of the same conceptual contents:

- Sliding Window
- Image Pyramids
- Feature Extraction
- Classification.

In the first step, significant features are extracted from the input pixel regions. These characteristics are then used to train the classifier. This enables predicting whether an object is located in a pixel region or not. In order to conclude the exact position of the object from this functionality, the classifier must always be provided with a different image section. The so-called Sliding Window approach does this. There, a rectangle is systematically moved over the entire image [Woj+08]. The resulting image sections are then classified in each case. In order to obtain additional information about the dimensions of the object, this procedure is further combined with an image pyramid. This varies the zoom level of the image based on the image sections [Ade+84]. This combination results in a large number of classified image sections. Finally, the section

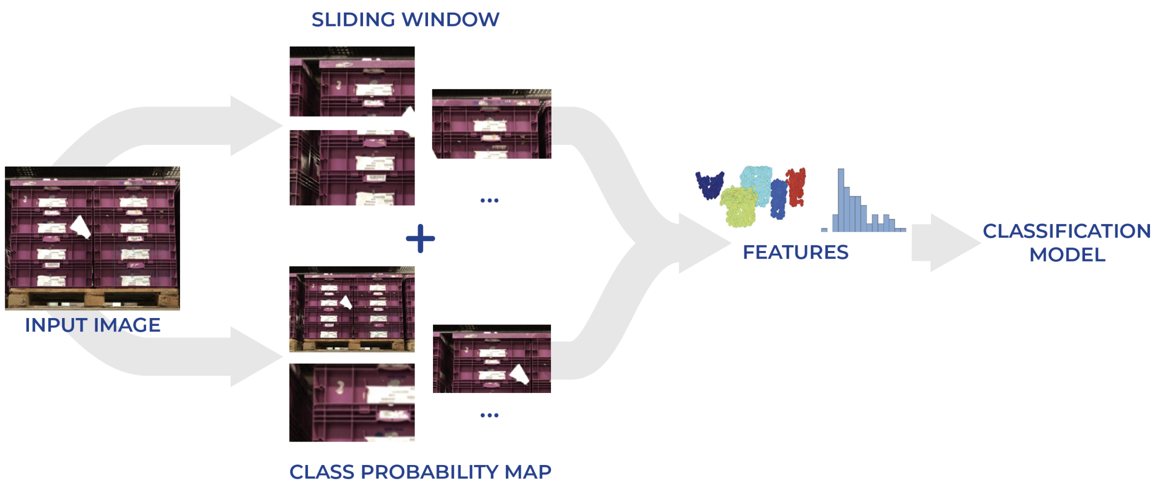


Figure 4.3: Generic overview of object detection methods (own illustration)

with the highest confidence of the classifier can be selected from these and regarded as the detected object position. For this final selection, non-maxima suppressions are used.

In particular, the first two steps are implemented in a variety of different ways in the literature. The most popular are briefly explained in more detail below. These are hair cascades, HOG + linear Support Vector Machines, Deep Neural Networks for classification combined with a Sliding Window approach as well as single hot detectors.

Haar Cascades

Haar Cascades is an object recognition algorithm introduced in 2001 by for face recognition. The system is provided with a certain number of positive and negative images. The selection of the features, as well as the training of the classifier, is done using Adaboost and Integral Images. [Soo14] [VJ01]

Features are determined in the form of pixel subtractions of different rectangles. These subtractions are put together differently according to the distribution rectangle. Those are shown as an example in Figure 4.4 For two rectangle features, the sum of the pixels contained in the white rectangles is subtracted from the sum of the pixels contained in the gray rectangles. [Soo14] [VJ01]

The process of "Boosting" works with the learning of a single simple classifier and rewriting the weight of the data where errors were made with higher weights. Afterward, a second simple classifier is learned on the weighted classifier, and the data is reweighted on the combination of first and second classifier and so on until the final classifier is learned. Therefore, the final classifier is the combination of all previous n-classifiers. The Ada-boost cascade of classifiers is one of the fastest and robust methods of detection and characterization. However, it presents some limitations on complex scenes, especially those that change shape. [Soo14] [VJ01]

According to the original implementation, Haar Cascades are used today, mainly in the area of facial recognition. The advantage of this approach is the very low required computing time. In particular, their accuracy has a negative effect, as they often tend to have high false positive rates. Besides, depending on the parameters made available to cascade, they can completely miss objects depending on the respective parameters chosen in the settings. The main problem here is the necessary image-selective fine adjustment of the parameters. [Soo14] [VJ01]

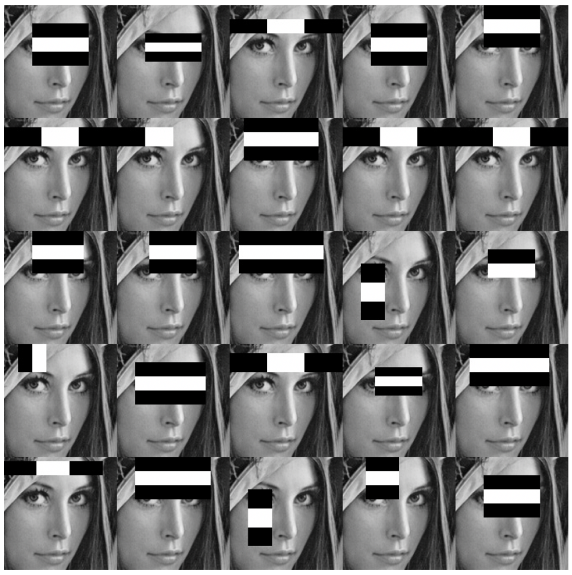


Figure 4.4: Visualization of haar cascade features (own illustration after [VJ01])

HOG and Linear SVM

The Histogram of Oriented Gradients (HOG) for Human Detection by was presented in 2005. In the literature, this approach is usually referred to as HOG and Linear Support Vector Machine (SVM). The objects to be detected by the algorithms often vary greatly in terms of colors (e.g., cars), while the object structure is more uniform. These generalizable object structures are attempted in this approach in the form of gradients in the respective specific directions to capture. These are called Histogram of Oriented Gradients. Therefore pixels are grouped into cells, and the predominant orientation is captured. [Dal+10]

Based on these features, a classifier is then trained analogous to the hair cascades. Combined with Sliding Window and Image Pyramids, this allows conclusions to be drawn about the position and dimensions of the objects being searched for. Compared to hair cascades, the training effort is lower due to less manually adjustable parameters and higher recognition accuracy. The disadvantage is the required computing speed, which is comparatively slow. [Dal+10]

Sliding Window combined with Neural Networks

Another often used approach to image classification in recent years has been the usage of neural networks. These were able to beat conventional classifiers or classical machine learning methods in terms of general validity and performance, regardless of the selected application domain. This also applies to image classification. [Had+14]

A neural network is a biologically inspired structure designed to model the behavior of information processing capabilities of human brain neurons. An artificial neural network is composed of mathematical processing units called artificial neurons [Ger03]. One of these is shown as an example in Figure 4.5.

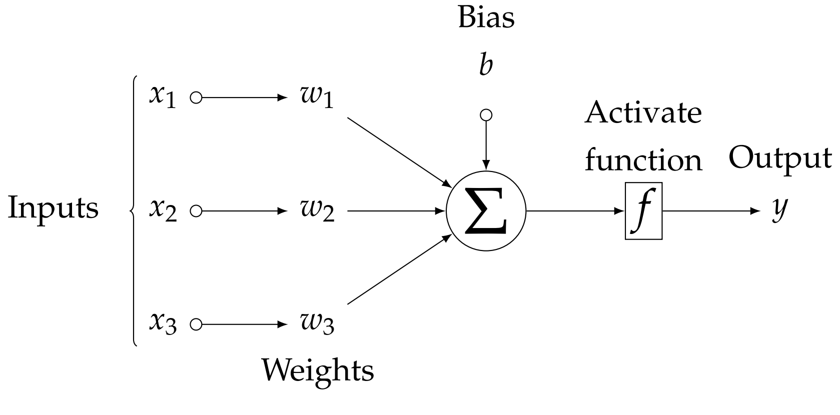


Figure 4.5: A single neuron with inputs (x_1,x_2,x_3), their corresponding weights (w_1,w_2,w_3), a bias b and the applied activate function f (own illustration after [Ger03])

A single neuron gets different inputs. These can be either external input variables such as pixel values or output variables of upstream neurons. The input values are multiplied by weights and added independently of their concrete origin. At the output, the sum of previously weighted inputs and bias is passing through an activation function. This function represents an evaluation of the neuron output. If the activation threshold has been exceeded, the output is activated and transferred to the next neuron. There are several nonlinear transfer functions, of which sigmoid, tanh and rectified linear unit [Sze+17].

The neuron function is described in the equations where x corresponds to the input vector, w to the weights, b to a bias, f to the activation function and y to the final output [Sze+17].

$$z(x, w, b) = w^T.x + b \tag{4.1}$$

$$y = f(z(x, w, b)) \tag{4.2}$$

Several neurons are combined in layers depending on the complexity of the application area. The linking of several such layers results in so-called artificial neural networks. There is always one input, one output, and several hidden layers available. If there are several hidden layers, this structure is called Deep Neural Network (DNN). This is shown in Figure 4.6.

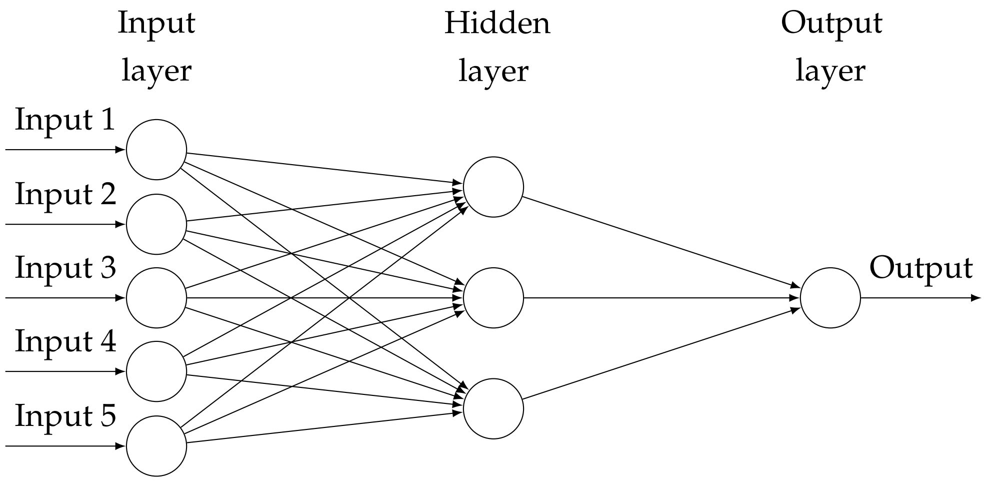


Figure 4.6: A simple artificial neural network, consisting of an input layer, one hidden layer and an output layer [B M14]

In the most common Feed Forward Networks, the final result is determined as a buzzer of the various activated or non-activated neurons. In order for the networks to be able to map the context to be captured from reality, they must be "trained". For this purpose, the weights of the individual neurons are adjusted depending on the calculated results. During training, the predictions of the network are compared to the labeled training data known as ground truth, and a loss function is calculated. The loss function shows how far away the output of the network is from the expected result.

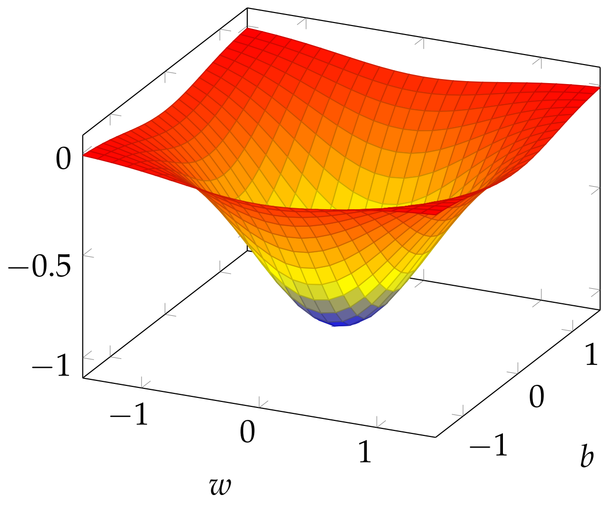


Figure 4.7: Visualisation of an example loss function [Sze+17]

Minimize the loss function is an optimization problem that defines the learning process. To this aim, different algorithms with hyperparameters may be used. The choice of the algorithm and its parameters has an essential impact on the performance of the learning process. A single flow of information through all the layers of a network to the loss function is a forward propagation or forward pass. However, the hand-engineered set of n-dimensional weights and biases is computationally expensive and time-consuming. The technique consisting of updating the parameters incrementally by calculating the gradient of the loss function after each iteration is called back-propagation [Sze+17].

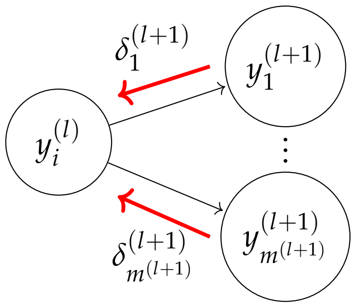


Figure 4.8: Once evaluated for all output units, the errors $\delta_i^{(L+1)}$ can be propagated backwards [Rud16]

Gradient Descent Gradient descent is an optimization algorithm that trains the network using the back-propagation technique. Gradient descent algorithms minimize the cost function J to the parameters θ (weights and bias) for the entire training dataset. With all weight and bias updates calculated, the weight and bias vectors are adjusted as follow [Rud16]:

$$\theta = \theta - \eta * \nabla_\theta J(\theta) \tag{4.3}$$

where learning rate η is a hyperparameter which influences the learning's speed.

After one epoch, the weight and bias vectors are moving in the direction of the global cost minimum. After a k-number of epochs, the gradient descent arrives at the global cost minimum.

With a large number of data points, it might take a longer time for the algorithm to converge. Therefore, other versions of gradient descent that differ in the amount of used data are implemented to minimize the loss function. For example, the Stochastic Gradient Descent (SGD) is used to optimize the computation speed. Instead of using the whole dataset, stochastic gradient descent updates the weights and biases after each data sample [Rud16]. Due to its stochastic nature, SGD takes an indirect path towards the minimum [Qia99]. Different algorithms for optimizing gradient descent are also implemented into state-of-the-art neural networks. For instance, the mini batch gradient descent is a widely used and, as its name states, performs an update for over mini batch of n training data [Rud16]. For example, a typical batch contains 256 examples from the training data set. The evaluation of the gradient of the loss function is then performed every 256 examples, which makes the operation much faster. Thanks to its effectiveness in term of accuracy and speed, mini-batch gradient descent is typically the algorithm of choice when training a neural network.

The learning rate η of such algorithm is a vital hyperparameter that significantly affects the performance and has to be defined before the training. With a small learning rate, the network converges too slowly. If the learning rate is too large, the model will fluctuate around the minimum, reach a local minimum or even diverge. A common approach to define the learning rate in deep learning is to start the training with a respectively tremendous learning rate and reduce it as the number of iterations increases. This technique is called the learning rate schedule.

Moreover, adaptive learning rate methods known as model optimizers are applied in order to boost the speed of convergence of the gradient descent algorithm and thus the learning [Rud16]. The most popular optimizers, which differ by the equation used to update the network's parameters, are Momentum, RMSprop, and Adam.

Momentum Momentum proposed by in [Rud16] adds a factor δ of the update vector and the parameters are adjusted using an intermediate variable.

$$v_t = \delta v_{t-1} - \eta \nabla_\theta J(\theta) \tag{4.4}$$

$$\theta = \theta - v_t \tag{4.5}$$

Therefore applying momentum optimizer, the learning algorithms avoid oscillations around local minimum and heads smoothly towards the optimal values of model parameters [Rud16]. Usually the momentum factor δ is set to 0.9.

RMSprop RMSprop, introduced by Geoffrey Hinton, proposes a method to define the learning rate. It splits the learning rate in an exponentially decaying average of squared gradients [Rud16].

$$v_t = \delta v_{t-1} + (1-\delta)(\nabla_\theta J(\theta))^2 \tag{4.6}$$

$$\theta = \theta - \frac{\eta}{\sqrt{v_t + \epsilon}} \nabla_\theta J(\theta) \tag{4.7}$$

Usually δ is also set to 0.9 and learning rate η to 0.001 [Rud16].

Adam Adaptive Moment Estimation [KB14] known as Adam is a method to resolve the aggressive decrease in learning rates where not only the average of previous squared gradients v_t is stored also the past gradients m_t. This operation is described in the equations below.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)(\nabla_\theta J(\theta)) \tag{4.8}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_1)(\nabla_\theta J(\theta))^2 \tag{4.9}$$

To update the parameters θ Adam uses the formula:

$$\theta = \theta - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{4.10}$$

where:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1} \tag{4.11}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2} \tag{4.12}$$

The parameters of Adam are per default set to β_1=0.9 and β_2=0.9999 [KB14].

In addition to the layer patterns mentioned above, individual layer types have explicitly resulted in each use case. These are explained exemplarily for the field of application of the classification of images in the context of the method presented here for object recognition as an ensemble of Convolutional Neural Network, Sliding Window, Pyramid Image, and Non-Maxima Suppression. The unique feature of this approach compared to the two other methods presented is that CNN learn the features that are needed as parameters for the classifiers themselves.

CNN are structured differently than standard neural networks to be adapted individually to computer vision applications. Regular Neural Networks need input data in vector format to transform it through a series of hidden layers. Every layer is composed of a set of neurons, where each layer is fully connected to all neurons in the previous one. Therefore to apply them to unstructured data, particularly for images, a transformation into vectors, would take a vast amount of parameters to characterize the network or would make the data lose the spatial information. A single color image is represented by three-dimensional vectors, namely height, width, and depth. The depth of an image is the channels that describe color values of each pixel, namely red, green, and blue. Therefore, to process a small size image with only 600 * 600 pixels through a standard neural network, the input layer itself would have around one million neurons.

Consequently, a CNN is structured to perform directly on matrices describing the image, in order to conserve all dimensions of the image and to use the spatial information between the pixels. Similar to a standard network, a CNN also consists of multiple layers with several neurons. However, CNN architecture extends the ANN structure by adding two new types of hidden layers, in which the neurons are not fully-connected to all the neurons in the layer before but only to a small region of it [AL18], namely:

- Convolutional layers
- Pooling layers

The new types of layers enable the output to be reduced to a single vector, organized along the depth dimension. In CNN's, the input layer is always a convolutional layer. This layer receives from the environment the image as a three-dimensional input, extracts features, and generates the corresponding feature map following the operation in 4.13. The feature map or the convolved image is the resulting matrix $I * K$ of an operation called a convolution operation where a filter or kernel K, is applied and slid over the image matrix I. In the first step, the elements of the filter are multiplied by the elements of the input matrix in each filter position, as shown in the figure. The sum of the multiplications is an element of the feature map [K O15].

$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} I_{x+i-1,y+j-1} \tag{4.13}$$

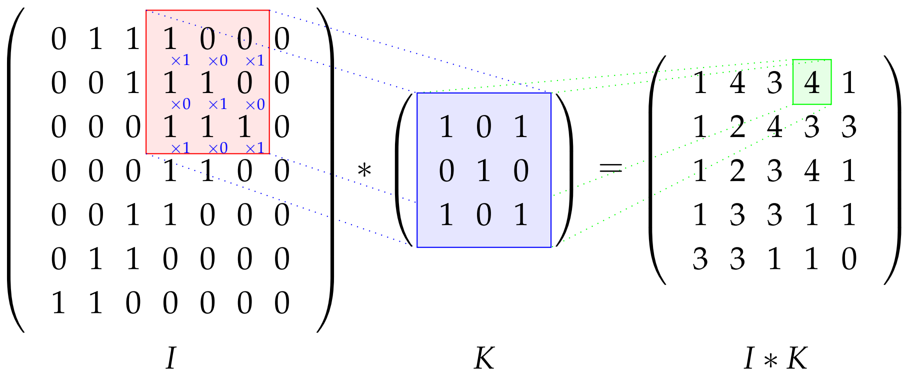


Figure 4.9: Example of 2D convolution operation with a kernel K and image matrix I [K O15]

One drawback to the use of convolutional operations is the increase of the dimensionality and computational costs for the entire network. Therefore, a pooling layer, called also downsampling layer, is applied after one or multiple convolution layers and aims to decrease the dimensionality, and thus simplify the model and reduce the computational effort significantly. A pooling layer operates over each resulting feature map by passing only the most active element. The mostly used pooling layer is the MaxPooling layer. It merely uses the highest value of a kernel matrix and discards all others.

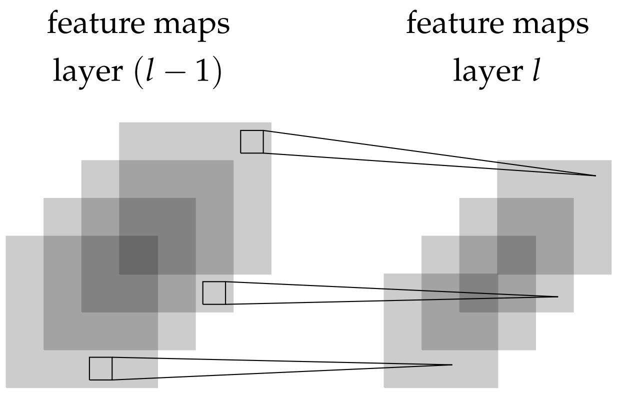


Figure 4.10: Layer l is a pooling layer and given $m_1^{(l-1)} = 4$ feature maps of the previous layer, all feature maps are pooled individually [AL18]

Finally, the last pooling layer is followed by one or more fully connected layers that generate the predictions. The fully connected layers connect every output from the previous layer with each neuron and perform the classification on the features extracted and downsampled by the convolutional layers [AL18].

A typical and simplified CNN architecture of a classifier is illustrated in the following figure. An image containing a container is passed through two convolutional layers including convolution and pooling operations to be later fed to fully connected layers that output a class probability of the object (1 for class 'container').

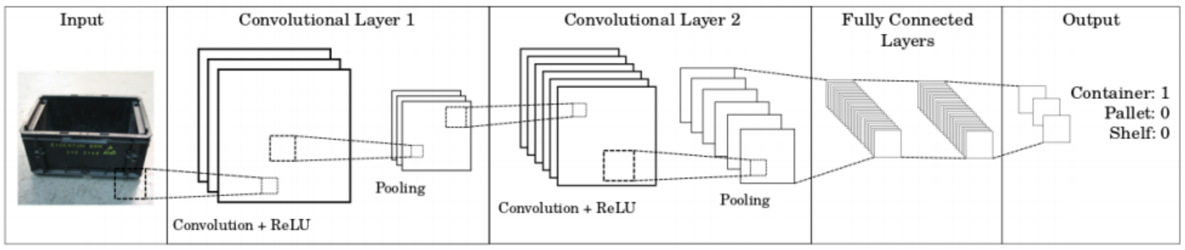


Figure 4.11: Architecture of a CNN classifier (2 convolutional layers followed by fully connected layers) (own illustration after [AL18])

The more detailed features allow more precise predictions about the objects contained in the images. Any Deep Learning model can be used. It is also possible to use pre-trained architectures, which are then adapted to the respective application by further training data, albeit to a small extent. The time disadvantages already mentioned in the previous methods, which result from the combination of the Sliding Window and Image Pyramid, are also evident here. The manual effort is also very high due to the training of the Deep Neural Networks as well as the granular parameter adjustment for the scaling of the Image Pyramid and the step size of the Sliding Windows.

Neural Networks for Object Detection

In order to counter these disadvantages, the trend in recent years has been towards holistic Deep Learning Object Detection Frameworks. These usually consist of two main components: the object recognition framework and the basic network, which is integrated into the first.

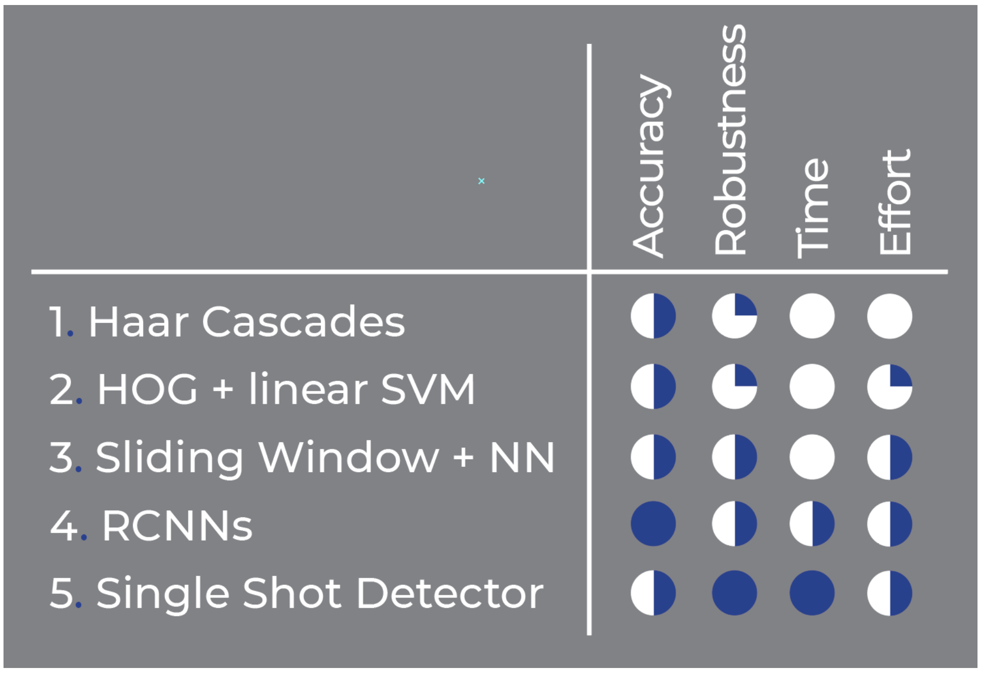
As such, classification CNN architectures are used that have been pre-trained on an extensive data set such as ImageNet to learn a broad spectrum of different filters. VGGNet, ResNet, MobileNet or DenseNet, for example, are used as such networks.

No best practice path has yet emerged for the object recognition frameworks. Instead, there are several comprehensive approaches in the literature which achieve comparable results but determine them in different ways. Simplified these can be divided into two groups: network combinations and single shot detectors. Since the exact details of these architectures will be explained in more detail in the following subchapters, only two possible solutions will be discussed here on a superficial basis. These are Faster R-CNNs representative for network combinations and Single Shot Detectors (SSDs), which represent the cluster group of the same name. For the former, potentially relevant image regions are first identified. These are then analyzed in a network. A characteristic feature of this network architecture is that in addition to the classification output, a Fully Connected Layer with softmax is implemented as a parallel Fully Connected Layer, which represents the Bounding Box Regressor. In contrast, the so-called Single Shot Detectors do not require any image region suggestions. A single network is used, which uses different bounding boxes, which are then adapted as part of the prediction. This procedure leads to significantly higher speed with comparable performance.

Pre-Selection

At this point, the methods for object recognition listed in the previous subchapter are evaluated concerning the postulated requirements for the Detection-Module of the

Perception Algorithm. The result of this analysis is shown in Figure 4.12. It should be mentioned here that the evaluation of the respective approaches is exclusively theoretical and based on literature analyses. This has to be considered especially concerning the requirements regarding precision and robustness, as these are strongly influenced by the expected fields of application and the training data available from them.



	Accuracy	Robustness	Time	Effort
1. Haar Cascades	◑	◔	○	○
2. HOG + linear SVM	◑	◔	○	◔
3. Sliding Window + NN	◑	◑	○	◑
4. RCNNs	●	◑	◑	◑
5. Single Shot Detector	◑	●	●	◑

Figure 4.12: Comparison of different object detection methods (own illustration)

Overall, this comparison initially shows that the use of Deep Learning significantly increases the fullfilment of requirements in all four areas. Traditional approaches to object recognition (Haar Cascades or HOG + linear SVM) can no longer keep up concerning robustness, calculation time, and manual effort. Approaches 3 - 5 distinguish all perform comparably well across all requirements. Due to the high expenditure of time, approach three is excluded from further investigations. Region-based networks are usually more accurate than single shot detectors. Therefore they are explained in detail in the following together with the Single Shot Detectors despite the known disadvantages in the areas of effort and time.

4.2.3 Deep Neural Networks for Object Detection

In this subchapter, the already mentioned large variance in the solution space of neural networks for object recognition is taken into account by presenting the different approaches in more detail. The following architectures are based on their popularity considered for this purpose:

- Regionbased CNN (R-CNN, Fast R-CNN, Faster R-CNN)
- R-FCN
- SSD
- FPN
- RetinaNet
- Yolo

Regionbased CNN

As mentioned in this chapter, the main disadvantage of the combination of Sliding Window, ImagePyramid and CNN is that the calculation time results from the very high number of iterations which result from the respective image sections together with the different zoom levels. As an alternative, Faster R-CNN showed outstanding results in recent years.

Faster R-CNN is the third improved version of R-CNN and as the name states the faster one [P Y17]. Indeed, to present Faster R-CNN, its predecessors, namely R-CNN and Fast R-CNN, have to be first introduced. Proposed by Girshick et al., R-CNN is an approach that extracts around 2000 regions from the image using a selective search algorithm based on segmentation of an image, to see if any of these selected regions contains any object. In a second step, each region, called region proposal, is warped to square size and fed into a convolutional neural network that extracts features. Finally, R-CNN uses a Support Vector Machine (SVM) to classify the region and a linear regressor to refine the resulting bounding box (Figure 4.13).

Compared to approaches without deep learning within its generation, R-CNN has significantly improved the detection precision. However, the model has significant limitations: too slow and too expensive in space and time [P Y17]. This is since it requires a CNN forward pass, including features extractor, a classifier, and a regressor, for each object proposal, without sharing computation [Gir15]. To overcome the problem, the second version of R-CNN [Gir15] was proposed by one of the co-authors of the

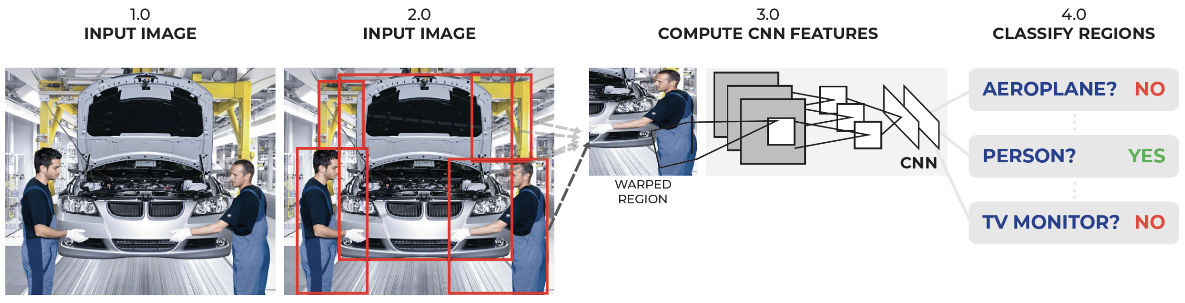


Figure 4.13: Architecture of R-CNN (own illustration after [Gir+13])

original paper, Ross Girshick. Fast R-CNN aims to reduce the time caused by the high number of CNNs used to process all the region proposals for one image.

Fast R-CNN comes with the introduction of RoI (region of interest) pooling. Instead of applying a CNN for each region proposal, the whole image is fed to a single CNN composed of multiple convolutional layers and max pooling layers to generate a feature map [P Y17]. From the feature map, regions of interest (RoI) are generated with the selective search method and reshaped with an RoI pooling layer. RoI feature vectors are then created. Additionally, Fast R-CNN replaces the classifier and the regressor by two layers on top of the network. Thus, it unified the three steps: the feature map generator, the classifier, and the regressor in one model (Figure 4.14).

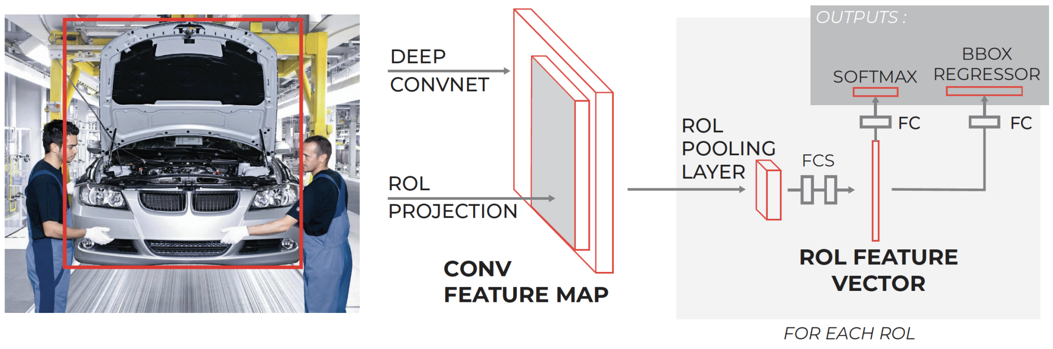


Figure 4.14: Function principle of Fast R-CNN (own illustration after [P Y17])

With this new architecture, Fast R-CNN improved the time consumption, but the region proposal step with the selective search method is still computationally expensive. Therefore, the most recent version Faster R-CNN, proposed by S.Ren et al. [P Y17], introduced the region proposal network (RPN), a fully convolutional network, that replaces selective search methods. The architecture of Faster R-CNN is shown in figure

4.15.

Faster R-CNN is built with two networks: A region proposal network (RPN) to select and generate region proposals to be fed to a second network for detection. Similar to Fast R-CNN, Faster R-CNN employs at first a CNN to extract feature maps from the image.

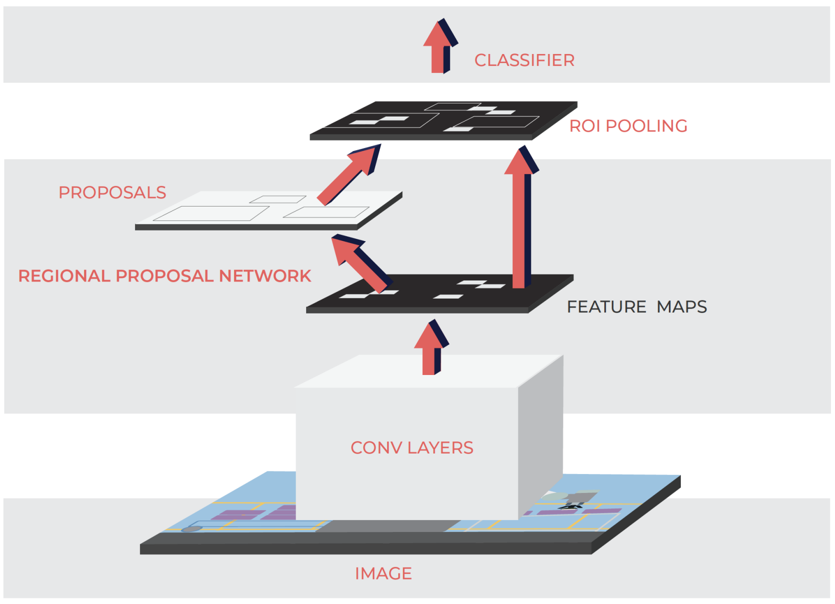


Figure 4.15: Architecture of Faster R-CNN (own illustration after [P Y17])

For region proposal, RPN is applied. RPN is a two convolutional layers based network that generates region proposals from a feature map using k number of scaled prefixed boxes known as anchor boxes. To this end, a $n * n$ sliding window moves across the feature map. For each location of the sliding window, the RPN generates multiple possible regions using k anchor boxes. The proposed algorithm employs three types of scales and three aspect ratios, which result in $k = 9$ anchor boxes per sliding window. Each resulting sliding window is then mapped to a vector that is fed into fully-connected layers: a class prediction layer and a regression layer (Figure 4.16). Finally, Faster R-CNN uses also additional layers, namely ROI pooling and classifier before getting the predicted bounding boxes.

R-FCN

While in the approaches just described the regions are derived from the feature maps in order to reduce the number of regions to be analyzed in the further course and

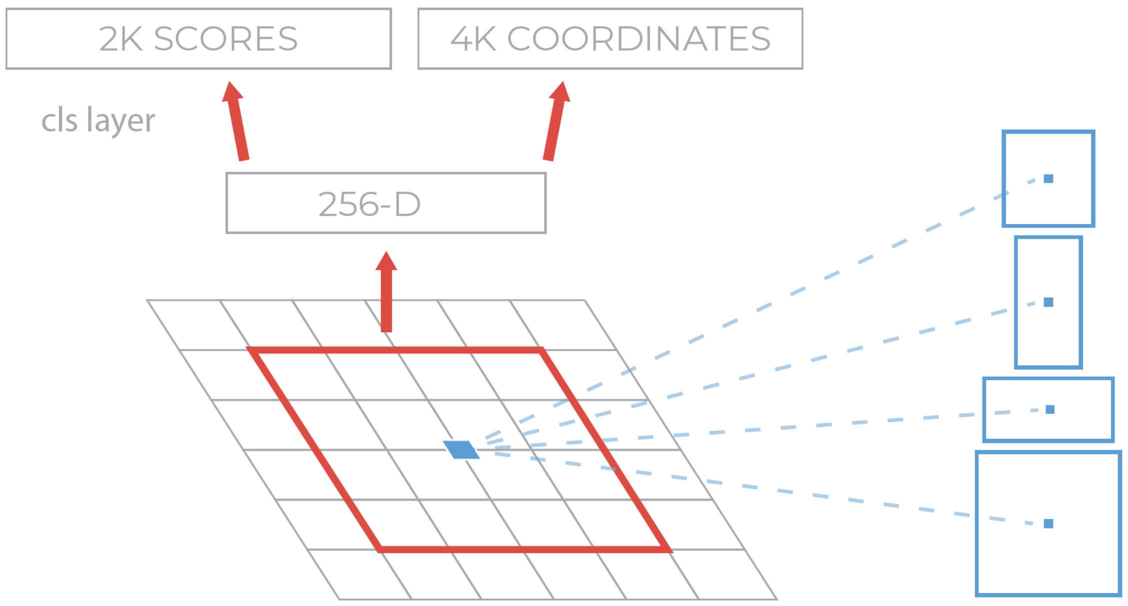


Figure 4.16: Region Proposal Network (RPN) (own illustration after [P Y17])

thus increase the speed of the algorithm, it is the approach of Region-based Fully Convolutional Networks (R-FCN) to reduce the computational effort per region further. The region-based feature maps, which are independent of the selected regions, are calculated once initially and not repeated for each region. This reduces the effort for classifying the respective regions since the feature maps are already available as initial input.

When using Faster R-CNN and R-FCN on the same data set, an increase of the speed by a factor of 20 can be detected. For the exact details of the network, please refer to the original publication [Dai+16].

SSD

Introduced by C. Szegedy et al. in the paper [Wei+16], the Single Shot Multibox Detector succeeded in reaching high performance for object detection tasks. Contrary to Faster R-CNN, SSD is a single stage network combining regional proposals and feature extraction in one network. The architecture of SSD, presented in figure 4.17, makes it possible to speed up the process in combining it in a single network. The general idea behind SSD is predicting class scores and offsets for a predefined set of default bounding boxes [Wei+16].

Therefore, SSD uses the base of the network, to extract multi-scale feature maps and applies convolution filters for each cell to make predictions using defaults boxes. It discards then the fully connected layers. In a single feature layer, the same set of default boxes centered at the corresponding cell are used. However, in different layers, SSD uses different sets of default boxes to scale all objects at different sizes and resolutions. These predictions can be under the form of a class score or an offset. During the whole process, SSD predicts the output after multiple convolutional layers that operate at different scales and progressively reduce the size of the input [Wei+16].

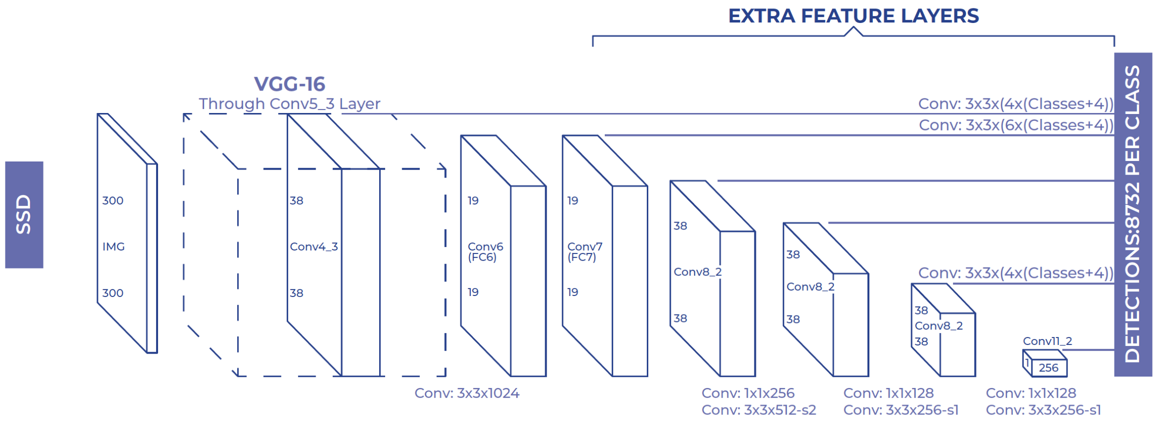


Figure 4.17: Architecture of SSD (own illustration after [Wei+16])

As shown in Figure 4.17, the SSD net architecture outputs a certain number of regions after a certain number of Feature Extraction Convolutions. A certain number of bounding boxes results for each of these regions. These have different sizes and length- aspect ratios. For each of these cells, class results and offset values relative to the original bounding box are subsequently returned. Depending on the network layer anodes shown, a total of 8732 bounding boxes can be detected with SSD.

The disadvantage of this approach is that the deep layers of the nets do not generate good enough resolving features which would be necessary for the detection of smaller objects. The necessity of complex data augmentation also points to a sizeable necessary training database.

RetinaNet

RetinaNet consists of the main network and two task-specific subnetworks. This can be seen as well in Figure 4.18 Analogous to the networks described above, the basic network is used to determine feature maps based on a large number of convolutioal layers. In this case, a feature Pyramid Network is used, which is based on ResNet50 or ResNet101. The underlying principle of the Pyramid Networks feature is that unlike the Image Pyramid, where multiple images are generated by applying different zoom levels, only the feature maps for such manipulations are used based on an image. This can increase the accuracy, as even small objects in the images can be highlighted accordingly. Also, the inference speed is higher than with the original approach, since only one image needs to be extracted for the corresponding features. [Tsu17]

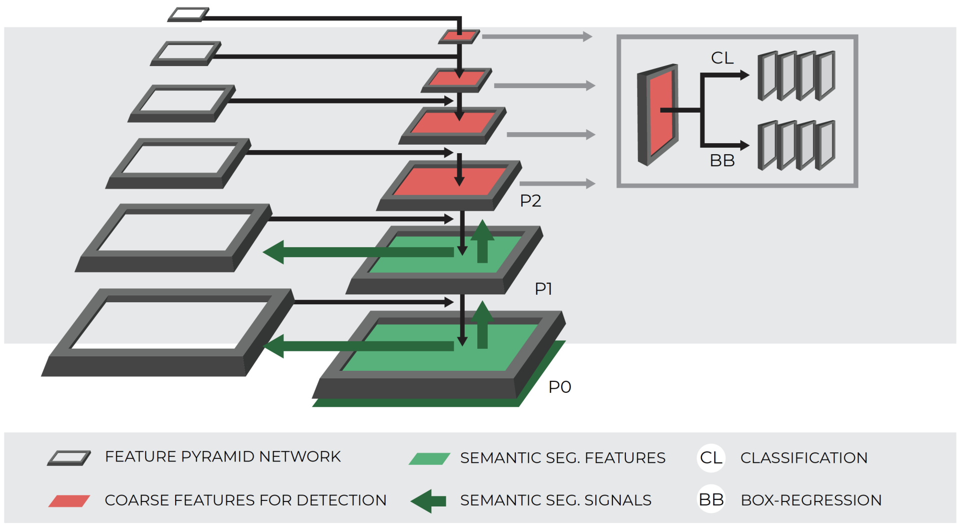


Figure 4.18: Architecture of RetinaNet (own illustration after [Tsu17])

Based on this, the output of the basic network is classified in the first subnetwork. For this purpose, the probability of occurrence of each class is determined for all anchor points. For this purpose, conv layers with several filters and subsequent ReLU activation functions are used. Finally, sigmoids are applied to the output. The second subnetwork deals with bounding box regression. This is identical in structure to the classification network, whereby no parameters are shared. [Tsu17]

Yolo

YOLO belong to the same category of detection algorithms as SSD: one-stage detectors. The underlying core principle of YOLO is to consider the detection problem as one regression problem. Thus, YOLO predicts directly from an input image objects locations and class probabilities [J R18]. At the input, the image is split into a grid. A grid cell is responsible for detecting an object if the center of that object is inside the cell. For each cell, two predictions are generated, namely, scores for object proposals (bounding boxes + confidence) and class probabilities, as shown in figure 4.19.

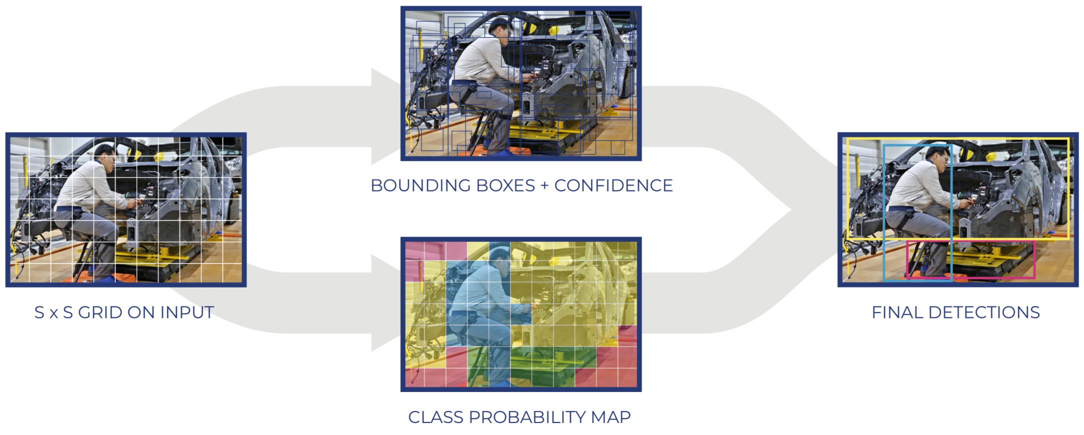


Figure 4.19: YOLO [J R18]

The predictions of bounding boxes around the object maybe duplicated, if the object is located within two or more grid cells. Therefore again, a non-maxima suppression is applied. It consists in selecting, among the overlapping bounding boxes, if the overlap exceeds a certain threshold, the one with high confidence. The original YOLO structure has 24 convolutional layers for feature extraction and two fully connected layers for output class scores and location coordinates, as shown in Figure 4.20. YOLOv2 is the second improved version of YOLO. One difference to the older version is the ability to deal with different resolutions of the image to increase the detector accuracy. Moreover, YOLOv2 has 19 convolutional layers and 5 (max)pooling layers [J R17]. Lately, YOLOv3 was introduced to be the third and most recent version of YOLO. Indeed, YOLOv3 uses a vast and deeper network with 53 convolutional layers. YOLO is considered to be highly compatible with real time systems thanks to its records in speed on different datasets [J R18].

4.2.4 Selection

The network architectures presented in the previous subchapter are difficult to distinguish in terms of their performance, as these vary only marginally from one another on the one hand and are dependent on the data sets used on the other. However, since the manual effort to train and optimize all presented architectures individually for each application would be too great, three architectures are selected at this point, which will

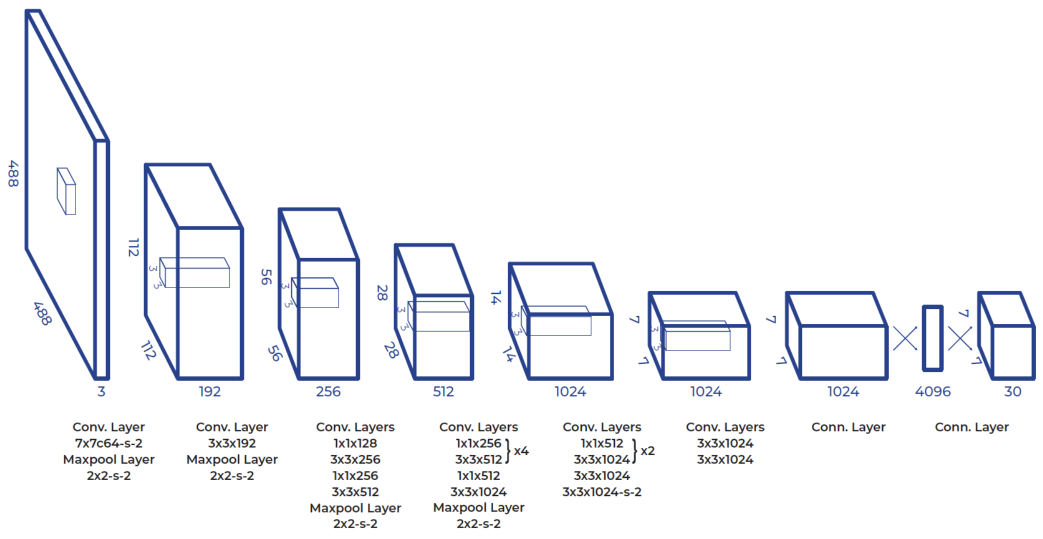


Figure 4.20: Architecture of YOLO (own illustration after [J R18])

be used in the Detection-Module depending on the specific application. These three architectures are:

- Yolo
- SSD
- Faster R-CNN.

Yolo was chosen because of its high frame rate and the consideration of the overall context of the image in object recognition, Faster R-CNN because of its high precision and robustness, and SSD because it has the same one-shot learning advantages as Yolo, but in theory performs better on close objects.

Evaluation Methods

In machine learning, there exist several measures for testing the performance of binary models. The standard evaluation is a performance analysis of a model while comparing the predictions and the expected results (ground truth). The most popular evaluation method, namely the mean average precision (mAP), is based on the confusion matrix (Table 4.1).

The four instances, TP, TN, FP, and FN, describe a comparison between the predictions

	Predicted = 0	Predicted = 1
Actual = 0	TN	FP
Actual = 1	FN	TP

Table 4.1: Components of a confusion matrix including true positives TP, false positives FP, false negatives FN, and true negatives TN [al14]

and the ground truth. Ground truth is the set of the images labeled with classes of observed objects and the correct bounding boxes of each object. Thus, the overlapping of the two areas, also called intersection over union IoU, is calculated. If the IoU exceeds a threshold, the prediction is described as true positive (TP), else it is considered as a false positive (FP). A ground truth bounding box which does not have a matching prediction is called false negative (FN), and the true negatives (TN) are the non-objects which are not detected. Based on these definitions, the mean average precision (mAP) is calculated to measure the accuracy of object detectors by taking the mean average precision over an IoU threshold. In information retrieval, mAP for a set of query results is the mean of the average precision scores for each query[source]. In object detection, an image with detections is comparable to a set of query results.

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{4.14}$$

where Q is a number of queries. Average Precision (AP), in turn, represents the average of maximum precision values overall correct detections. In other words, it is the area under the precision-recall curve.

$$AP = \sum precision(x) * (recall(x) - recall(x-1))) \tag{4.15}$$

Precision measures how many correctly predicted positive observations in proportion to the total predicted positive observations and thus calculated as a fraction of ground truth objects from all detected objects:

$$Precision = \frac{TP}{TP + FP} \tag{4.16}$$

Whereas, recall is the ratio of correctly detected objects to all ground truth objects:

$$Recall = \frac{TP}{TP + FN} \tag{4.17}$$

Figure 4.21 compares the mAP and speed of current state-of-the-art methods to examine the accuracy-performance trade-offs. The mAP is measured using an IoU threshold of 0.5, and the time performance is evaluated through the metric of frames per second (FPS). The standard PASCAL VOC 2012 dataset is chosen as the evaluation dataset. The results show that SSD succeeded in reaching high performance for object detection tasks in terms of precision, scoring over 76% mAP (mean Average Precision) at 19 frames per second [J R17], which outperforms Faster R-CNN (73.2% mAP at 7 FPS). On the other hand, YOLO manages to be the fastest object detection method on PASCAL VOC. While running at 45 frames per second, it is more than twice as fast as SSD and Faster R-CNN. However, it performs with an mAP score of 63.4%

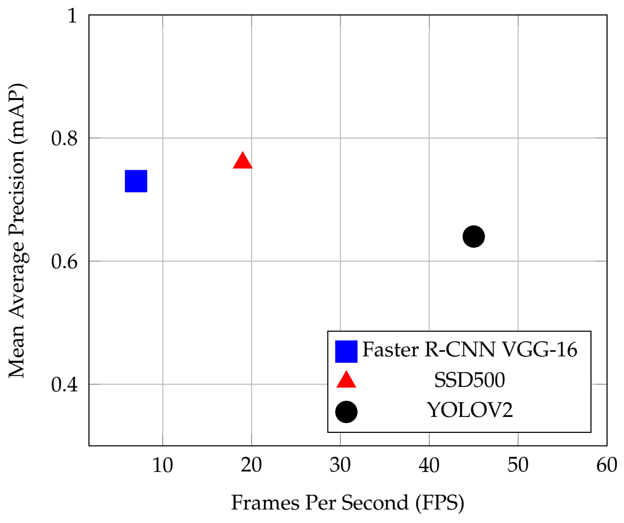


Figure 4.21: Performance of detection frameworks on the publicly available dataset PASCAL VOC 2007 (own illustration after [J R17])

Trained on open-source datasets, state-of-art methods succeeded in achieving high performance in terms of precision and speed to solve the object detection problem. For several computer vision applications such as pedestrian detection for autonomous driving or tracking systems, these methods can be beneficial. To this end, the fine-tuning approach allows the applicability of state-of-the-art models on different tasks, while avoiding the building and training of models from scratch. Fine-tuning of deep learning models describes the technique that consists of tuning the model parameters for the target application from the parameters pretrained on a similar application [Ouy+16].

The fine-tuning steps where previously-trained weights are available and thus used as a starting point for a new training with a new dataset (e.g., transfer learning) are as follow [Sze+17]:

- Select a pretrained state-of-the-art model
- Fine-tune the training parameters
- Train, the model with the new dataset (from target application)

In general, the selection of a model that suits the target application consists of testing several models with different training parameters and choosing the model with the highest accuracy.

4.2.5 Data Creation and Pre-Processing

One essential key for the training of a deep learning-based object detector is the dataset creation. This step is very time consuming, since preparing the dataset for the training requires acquiring a large number of representative samples and hand-labeling of the depicted objects and their positions in the image. The labeled images represent the ground truth, that should correspond with the range of scenarios in which the object detector should operate [Den+09]. The ground truth should contain pictures of different objects from the application's environment of the object detector from different poses and conditions. In this use case, the images are collected according to the described (Chapter 2) logistics areas.

The labeling process is done manually, where each object is defined by the drawing of a bounding box and the definition of its class. For this purpose, there are different open source labeling tools such as imglab, Labelbox, and Labelme. The main disadvantage of using such platforms is that they are hard to customize and, if the tool requires the images to be uploaded to external servers, it violates the confidentiality terms. Therefore, an in-house designed automated labeling tool that runs in the local server and can be adjusted to internal needs is used. The labeling tool generates text files automatically with the corresponding labels.

At this point, it should be noted once again that the selection of this labeling method was made for reasons of time and performance. Other annotation options, such as coloring the complete objects to train segmentation networks, are much more time-consuming.

On the other hand, these networks cannot yet fully keep up with the performance of the object detection networks described here. However, since performance plays a significant role in an industrial application, segmentation methods were not considered any further. The disadvantages resulting from this decision will be compensated by the following vision modules, which will be explained in more detail later in this chapter.

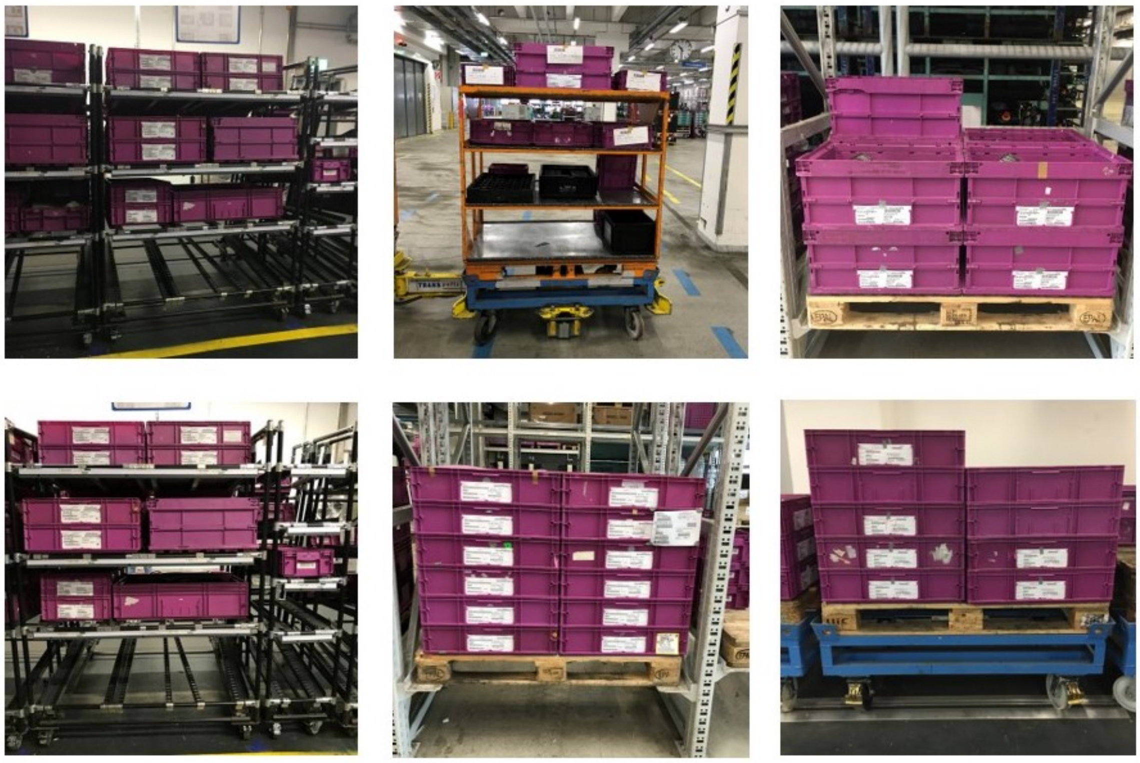


Figure 4.22: Samples of the collected training data (own illustration)

To gather enough samples for the training, around 4.000 images from different plants of the BMW Group in Germany (Munich, Leipzig, and Regensburg) were collected. Moreover, seven object classes were labeled, namely container, assembly line shelf, tugger train shelf, dolly, wheel, cage box, and pallet.

Preprocessing To improve the generalization of the data, two preprocessing settings are applied. First, all the images are resized to have a fixed shape. Secondly, data augmentation techniques are used, since a larger dataset is a key to a better performance of the network. Data augmentation is a method to artificially increase the amount of data without the effort of collecting and labeling new images. Trivial geometrical transformations of the images such as rotating, scaling, and shifting are applied [DMS18].

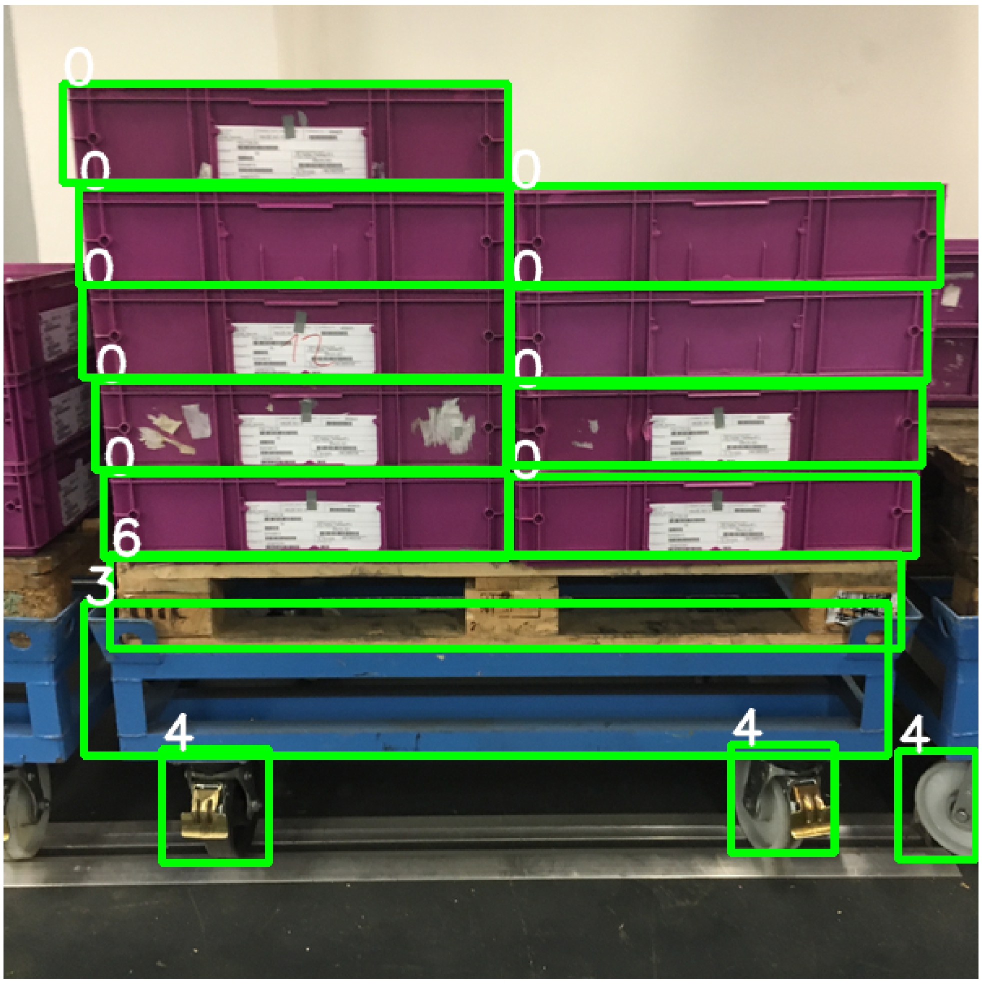


Figure 4.23: Example of a labeled image with annotations (own illustration)

To evaluate the performance of a trained network in detecting objects, unseen images are being used to compare the predictions and the ground truth. Therefore, the collected ground truth dataset is split in a training set to feed and train the network and a test set used to evaluate the results of the net. The dataset is randomly divided into training and testing set (respectively 80% and 20% of the total dataset). For the selected training parameters described below, a systematic, uniform procedure was chosen to determine them. Thus, various experiments were carried out based on the selected values in the original scientific publications. The best results for the particular net types could be achieved with the training parameters mentioned in each case. Since the focus of this doctoral thesis is not on the perfection of neural networks themselves, but on the perfection of the entire module of perception, the respective series of experiments will not be dealt with further at this point. Slightly, as already explained, the three preselected network architectures are compared in their parameterization, which has shown the best results on the existing data sets to determine the final structure.

Training Fine-tuning technique refers to the process that initializes the deep model parameters for the target application from the pre-trained model on another related application [Ouy+16]. To this end, the most accurate state-of-the-art object detectors, Faster R-CNN, SSD, and YOLO are fine-tuned and trained on an NVIDIA GeForce GTX 1050 Ti graphics card (Graphics Processing Unit (GPU)). To select the right architecture that achieves the best speed-accuracy balance, a performance evaluation is carried out. In general, open-source convolutional neural networks are implemented with various machine learning frameworks such as *Tensorflow, Caffe, Keras* and *Darknet*. In this thesis, for training Faster R-CNN and SSD architectures, models available on the *TensorFlow API* platform are used. *TensorFlow Object Detection API* is an open-source platform built on top of *TensorFlow* that provides various of object detection models ready to be re-trained and deployed. *TensorFlow*, developed by Google, is an open-source machine learning software library for numerical computation using data flow graphs [Aba+15]. However, YOLO is implemented with a different framework; *Darknet* [Red16]. The frameworks are installed on Ubuntu 16.04.

Results

Faster RCNN The pretrained *Faster R-CNN Resnet50* architecture provided by the *TensorFlow API* is used. The data is converted to the standard *TensorFlow* format as TFrecords. The best performance of the network is recorded with the following fine tuned training configuration and hyper-parameters, presented in the table 4.2.

Parameter	Value
Image Resizer	yes
Data Augmentation	yes
Batch Size	16
Momentum	0.9
Initial Learning Rate	0.0003
Learning Rate	Step Decay

Table 4.2: Training hyperparameters for Faster RCNN

SSD Different models based on SSD architecture were tested. The *ssdlite mobilenet v2* architecture available on the *TensorFlow API* platform achieved the best performance.

For the training, the TFrecords format for the data is required. The configuration and hyper-parameters used for the training are shown in the table 4.3.

Parameter	Value
Image Resizer	yes
Data Augmentation	yes
Batch Size	12
Optimizer	RMSprop
Momentum	0.9
Initial Learning Rate	0.004
Learning Rate	Exponential Decay

Table 4.3: Training hyperparameters for SSD

YOLO The last version of YOLO, *YOLO V3* implemented with the *Darknet* framework is fine-tuned and trained. The table 4.4 shows the configuration and hyper-parameters used during the training.

Parameter	Value
Image Resizer	yes
Data Augmentation	yes
Batch Size	12
Momentum	0.9
Initial Learning Rate	0.001
Learning Rate	Step Decay

Table 4.4: Training hyperparameters for YOLO

As a guideline, the training process is considered completed when the loss reaches a constant value and stops decreasing. The time needed for the training of *Faster R-CNN resnet50*, *ssdlite mobilenet v2 coco* and *YOLO V3* is respectively 12h, 9h and 48h. The focus of the robotics applications described here is primarily on the highest possible reliability and accuracy of results in order to guarantee the robustness of the processes in the industrial environment. Therefore, the training time itself was not further optimized during these training sessions. Particularly with the Yolo training sessions, significantly shorter training times could have been achieved through larger learning rates - but with lower overall efficiency.

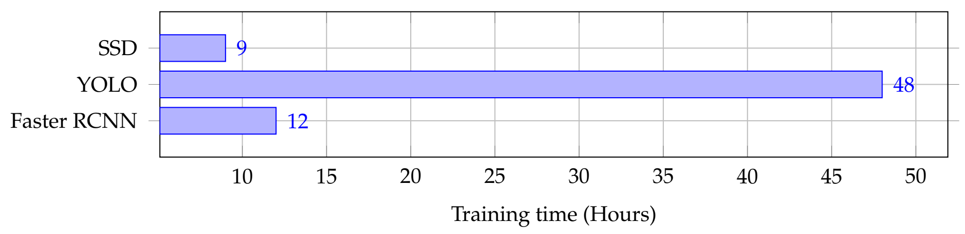


Figure 4.24: Training time comparison (own illustration)

Selection To evaluate the performance of the object detectors, the test dataset is used. For each image in the test dataset, the predicted bounding boxes annotations and the time needed to process the image are stored. Each predicted bounding box is represented with five descriptors:

- The class of the object
- The top left x coordinate
- The top left y coordinate
- The right bottom x coordinate
- The right bottom y coordinate

To measure the accuracy of the object detectors, the predictions on the test dataset are compared to the ground truth. Therefore, the mean average precision mAP with an IoU threshold equal to 0.5 is computed. The AP separately for each class and the overall mAP over all classes (7 classes) are both calculated. Besides, the number of frames per second FPS is computed to evaluate the speed of the detectors.

The models achieve different performances over different classes. An imbalanced class distribution can explain this in the dataset and the fact that object detectors are sensitive to object's sizes and specific visual appearance. For instance, some object classes like "container" have more samples than the other object classes such as "pallet". Therefore, for the comparison, the overall mAP is considered. In industrial applications, visual systems, used in robot motion control for material handling purposes, have to be precise and fast. To this end, the question is which detector and what configurations give us the best balance of speed and accuracy. Below is the overall accuracy v.s. speed trade-off of *Faster R-CNN Resnet50*, *ssdlite mobilenet v2 coco* and *YOLO V3*.

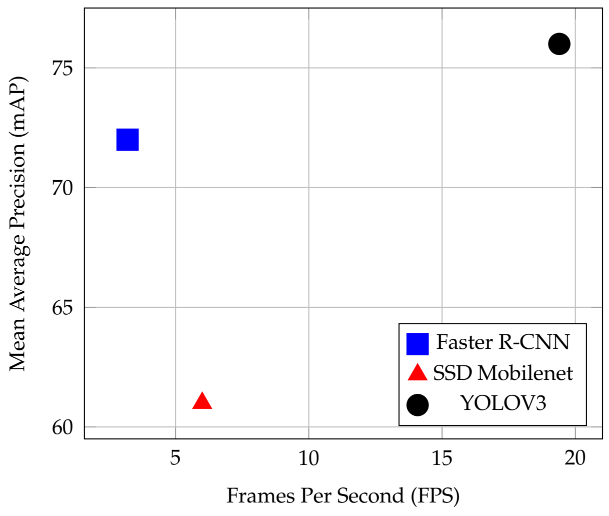


Figure 4.25: Speed and accuracy comparison. The accuracy is measured by Mean Average Precision over all classes in %. The Speed is measured by the number of Frames Per Second (own illustration)

Inference results indicate that, in terms of precision and speed, YOLO is leading with the score of mAP equal to 76% and FPS value equal to 19.4. Faster R-CNN, on the other hand, has a good performance in terms of precision with mAP similar to 72% but was prolonged. Whereas, SSD was faster than Faster R-CNN but had a low accuracy with only 61% mAP.

To give further insights, the precision-recall curves of the respective classes for the selected network are shown in Figure 4.26 and Figure 4.27.

Based on its outperforming result in terms of precision and speed, YOLOv3 is suitable to be the object detector in the vision system and used for the Detection-Module in the following.

4.3 Module 2: Selection

The output of the first module of the Perception Algorithm is the set of objects detected in the corresponding image of the searched class. This is checked in the second module

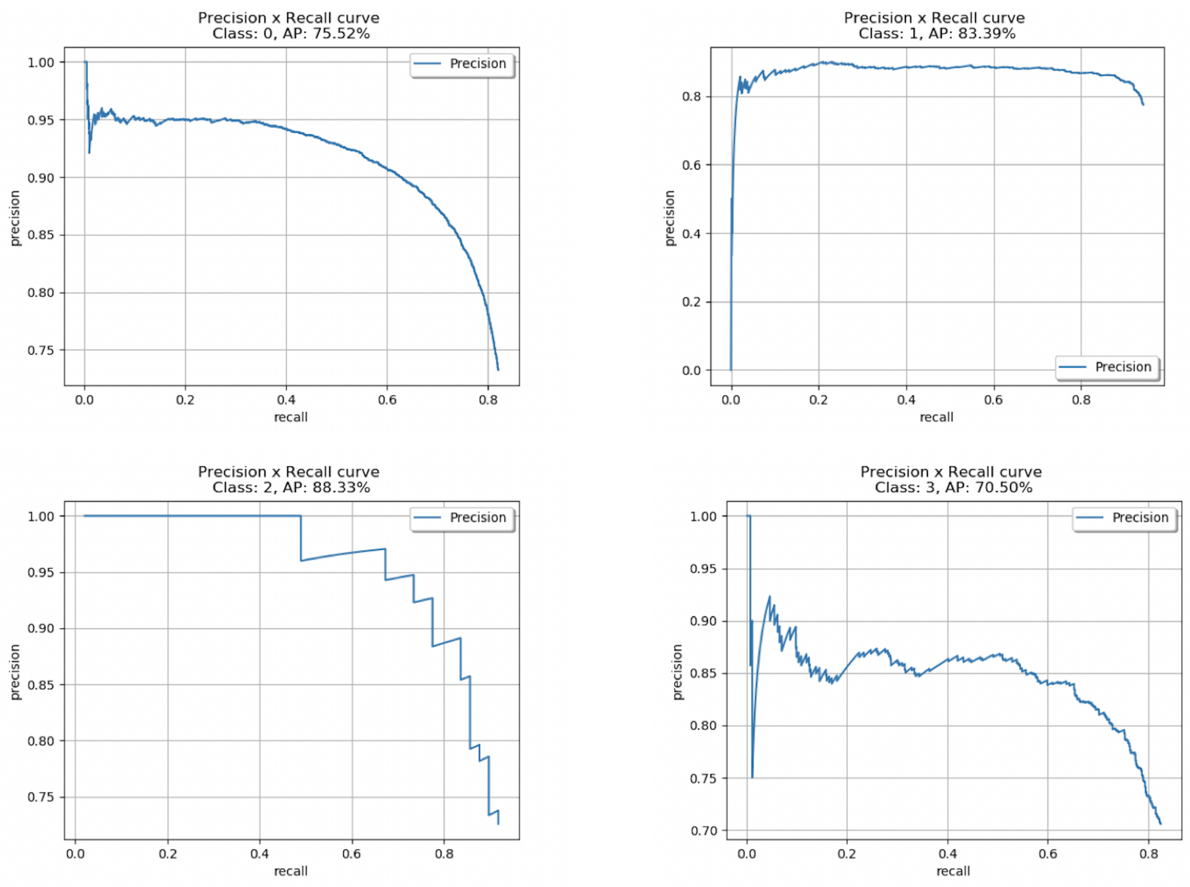


Figure 4.26: precision recall curves of the respective classes (part 1): container (AP = 75,52 %), shelf (AP = 83,39 %), tugger train shelf (AP = 88,33 %), dolly (AP = 70,50 %) (own illustration)

"Selection" concerning its process relevance so that finally an object remains which is to be gripped by the robot.

The scope of the module to be implemented will first be defined. These functions are then designed and performed based on an analysis of the current state of the art.

4.3.1 Target

To do this, all objects that are not in the robot's workspace must first be removed from the result lists at this point. Next, the relevant object for the current gripping process must be selected from the remaining objects based on a processing strategy to be designed. It should be noted that it has to be ensured that the handling process neither damages other objects nor restricts the robot in the execution of the overall process. This could be the case, for example, when depalletizing, if the robot grabs the containers in such an

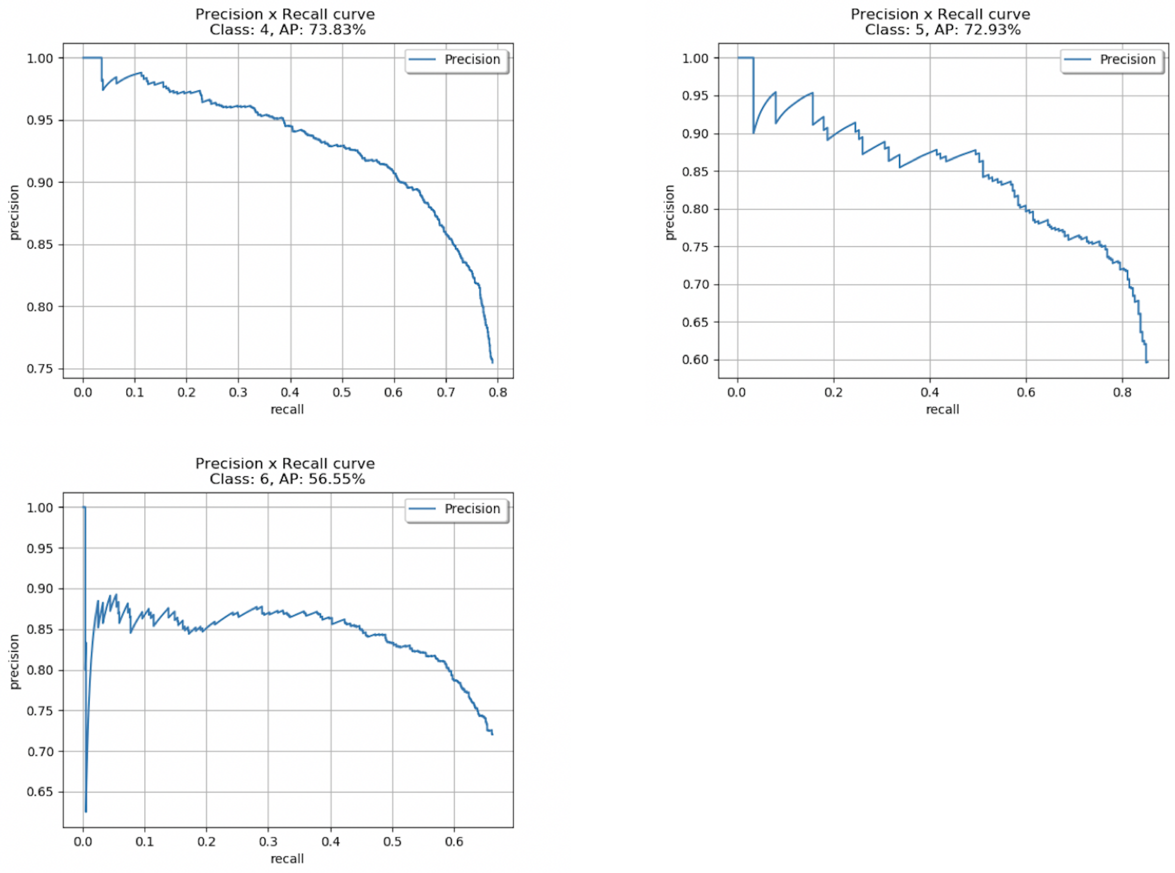


Figure 4.27: precision recall curves of the respective classes (part 1): dolly wheel (AP = 73,83 %), cage box (AP = 72,93 %), palette (AP = 56,55 %) (own illustration)

order that it can no longer pull the following containers over the previous ones.

4.3.2 Concept

The functional explanation of this module described here is very specific to the topic to be solved in the context of the present work. Due to this specialization, no overall algorithms or algorithm modules could be found in the literature on which to build at this point.

Following the described objective, the module itself is divided into two core components, namely the

- Selection of the attainable physical objects for the robot (working space selection)
- Selection of the objects tangible for the current process step (process step selection).

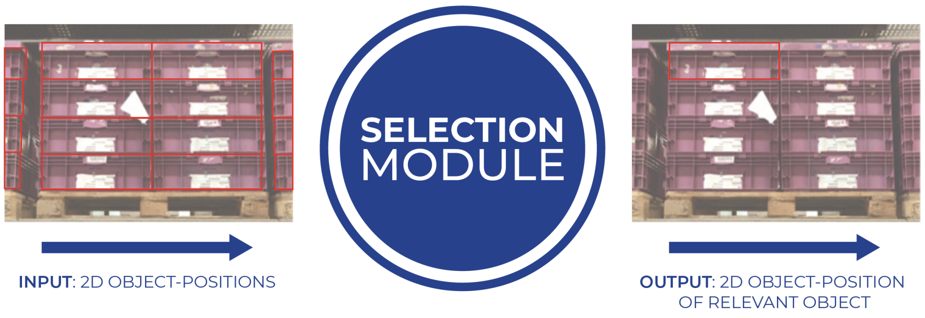

SELECTION MODULE
INPUT: 2D OBJECT-POSITIONS
OUTPUT: 2D OBJECT-POSITION OF RELEVANT OBJECT

Figure 4.28: Target-specification for the Selection-Module (own illustration)

Workingspace-Selection

The number of physically accessible objects for the robot is limited on the one hand by the robot hardware used and on the other hand, by the infrastructure involved. During depalletizing, the detected objects are limited by the pallet height and the maximum layer height in the y-direction. Since all objects to be depalletized must be on the palette, their length represents the x-direction boundary.

In the case of belt provision, the objects to be handled must fulfill the general condition that they are within the tugger train trailer rack when full loads are provided. In return, when empty containers are collected, the empty bottles must be part of the relevant staging rack. Besides, there is the working instruction that to better differentiate between full and empty containers, the latter is only provided in the lowest shelf line for the tugger train collection.

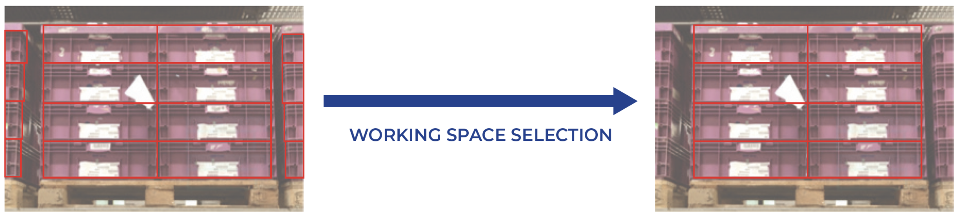


Figure 4.29: Example for applied workingspace-selection (own illustration)

During the final palletizing of the empty containers, only containers located on the conveyor system may be used for determining the gripping point. Here, the respective delivery stitch must be delimited so that two potentially alongside robots do not block each other.

Process Selection

As with workspace selection, the criteria for process selection must be postulated robot-selectively. In general terms, decisions must be made in two categories, namely based on the sequence of grips and the behavior in the event of an error. Whereas the former involves deciding the sequence to be processed, the latter specifies how to deal with mistakes.

In contrast to the supply of the assembly line or palletizing of empties, process selection during depalletizing at goods receipt is of particular importance. It must be ensured under all circumstances that the robot depalletizes in such a way that it continues to be able to retract the other containers after each container has been removed. The layer pattern to be processed for this is as follows. At first, the detected object is started in the upper left corner, and the first row is removed, as shown in Figure 4.30. Then the containers behind them - again from left to right - are drawn in one after the other. Once the entire first layer has been removed, you can start with the layers below. These are processed according to the same scheme until the pallet is entirely depalletized.

Since this sequence must be adhered to during depalletizing, no special measures such as gripping alternative containers in the event of errors can be actively carried out by the robot. Therefore, an employee must be called to solve the problem so that the robot can continue to work according to its predefined scheme.

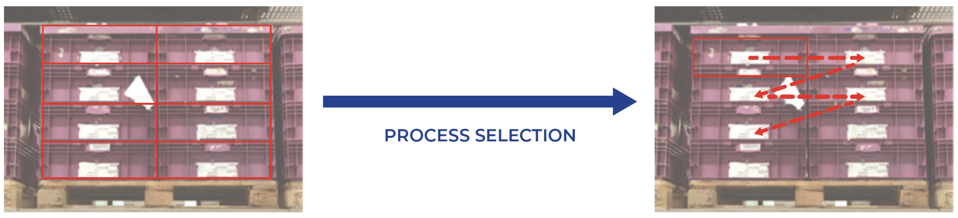


Figure 4.30: Example for applied process selection (own illustration)

The process does not have to adhere to a defined sequence when providing the assembly line. Although the containers are drawn into the robot via the linear axis system as in depalletizing, the containers do not have to be moved one above the other for this as they are located in roller-driven shelves. In this case, the decision as to which container is to be gripped can be made based on two decision criteria. On the one hand, it is possible to grab the container next, where the perception module has the highest confidence. On the other hand, it is also possible to process the containers in such a way that the robot's paths and thus the unloading time of the tugger train can be minimized. For this, a further vision module is necessary to identify the actual container contents. This can be achieved by reading the barcodes on the container. By the corresponding fixed allocation to the rack storage bins, the constant tuggers and the optimal unloading strategy can be determined.

In the case of false detections or even incorrect grips, it is also possible to continue working with the next most relevant container, since the higher-level process execution is not negatively affected in the long term.

When palletizing objects, the decision is made primarily about the confidence of the object recognition. Since the objects are re-arranged in the working area of the robot by the roller conveyor after each container handling operation, no restrictions about the processing sequence are necessary here. One phenomenon that can be observed more frequently here, however, is intangible containers. This can be caused, for example, by residual packaging from the components transported in the containers. Also, partial soiling may occur, or labels may still be present in the containers. If this happens, the specific gripping point is no longer taken into account in the selection for the next iteration, even if the confidence contradicts this.

4.3.3 Conclusion

Based on the results of the “Detection” Module, the positions of the objects located, there are first converted from pixel to world coordinates. Subsequently, the detected objects of the searched class are first examined in the workspace selection and then in the process selection concerning application-specific rules, and the object relevant for the next process step is selected.

4.4 Module 3: Localization

After the first two modules of the Perception Algorithm have detected the searched objects and selected the object relevant for further process execution, the third step is to determine the exact gripping pose. First, the precise target of the module is specified in more detail. Subsequently, relevant approaches are selected based on an analysis of existing methods and theoretical considerations. These are then adapted and implemented to the framework conditions required here before the final selection is made.

4.4.1 Target

The target of the "Localization" Module is to determine the gripping pose in three-dimensional space as it is visualized in Figure 4.31. Depending on the application, other coordinate vectors may be required. A list of the relevant target values is shown in Figure 4.32. For example, when depalletizing, the object rotations can be set to zero, since it can be assumed that the containers are positioned horizontally and vertically on the pallet, as otherwise, it would not be possible to transport the objects at all. Besides, small tolerances are no problem as they can be compensated by the flexible plastic suction cup of the gripper. When providing the objects, the gripper must be adapted to the new position of the shelves. Likewise, rotations of the container itself can occur in the roller-driven gravity racks. These must be able to be recognized. When palletizing empty containers, in addition to the x, y, and z positions, the rotation of the container must also be taken into consideration.

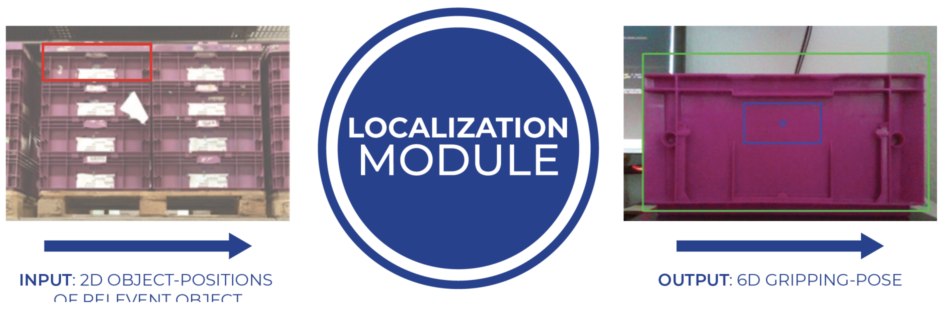


Figure 4.31: Target-specification for the Localization-Module (own illustration)

It is essential for the determination of the gripping point that neither significant contamination nor roughness occurs on the selected surface regions, as this would impede the sealing of the suction pad. With suction pads, the force that can be applied is directly proportional to the size of the suction pad. To enable a stable gripping process, this was, therefore, dimensioned as large as possible. However, this means that a very high degree of precision is required to determine the gripping pose, as otherwise, the gripper would rest on the webs on the surface of the container. Since, besides, it is only possible to grip on the container surface or, during palletizing, also on the inside of the container, it is necessary to ensure that the identified gripping surface does not contain a label or other sticker.

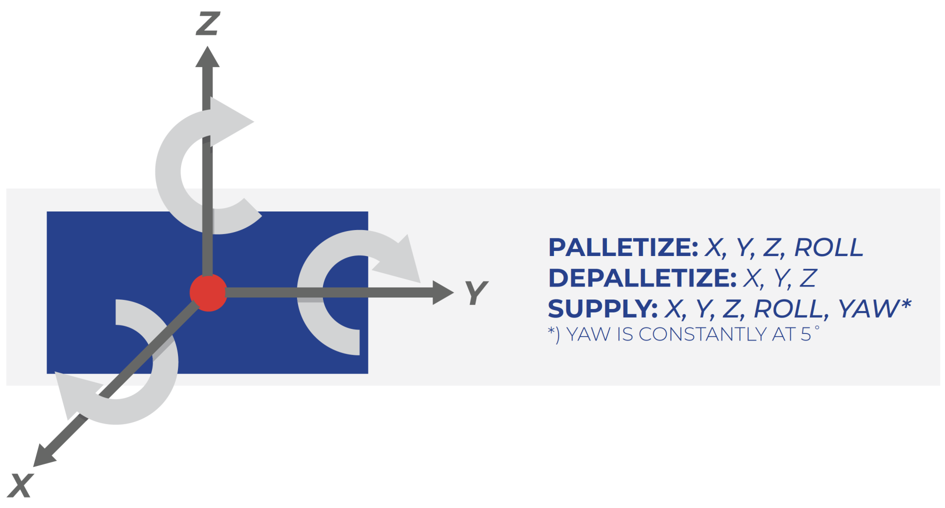


Figure 4.32: Overview about necessary target values for gripping informations (own illustration)

4.4.2 Related Work and Experiment Setup

Analogous to the defined overall concept of the Perception Algorithm, algorithms or algorithm modules are searched for in this literature search, which determines the corresponding three-dimensional gripping pose based on a known 2D position of the object. This narrow search space is also further limited by the use of the vacuum suction pad.

The only suitable approach that was found in this systematic search refers to a participant of the Amazon Picking Challenge 2016. A point cloud matching method was used to identify the objects. As soon as the product to be gripped was identified, the gripper was moved in the desired direction until a specific vacuum could be created between the gripper and the object. This procedure would be decomposable in some application cases where a rotation of the objects can be excluded. However, a low speed must be selected for the gripping process to avoid excessive collisions between the gripper and the object. With the strongly limited cycle times available in logistics, this approach is therefore not meaningful even if it is technically applicable. [Zen+17]

As no empirical knowledge can be built on at this point due to a lack of literature, three different procedures are tested first for the conception of this module. Based on these test results, the final module can then be selected and implemented. It should be noted that not all available container types can be validated (see Chapter 2). On the one hand, the scope would be too broad, and on the other hand, even with such validation, the complete functionality of the module could not be proven, since, as also described in

Chapter 2, the container spectrum is subject to continuous change. To nevertheless be able to make a representative statement about the module performance, exemplary containers are selected, which best reflect the relevant influencing variables.

4.4.3 Experiment 1: Gripping Points as Bounding Box Centers

The most obvious approach for determining the gripping point is to calculate the center of the bounding box and to determine the distance from the depth image for these coordinates as it is suggested by [Zen+17]. It is also conceivable that not only the distance of the center point can be determined, but also the rotation and tilting of the container can be deduced by measuring the distance of other points.

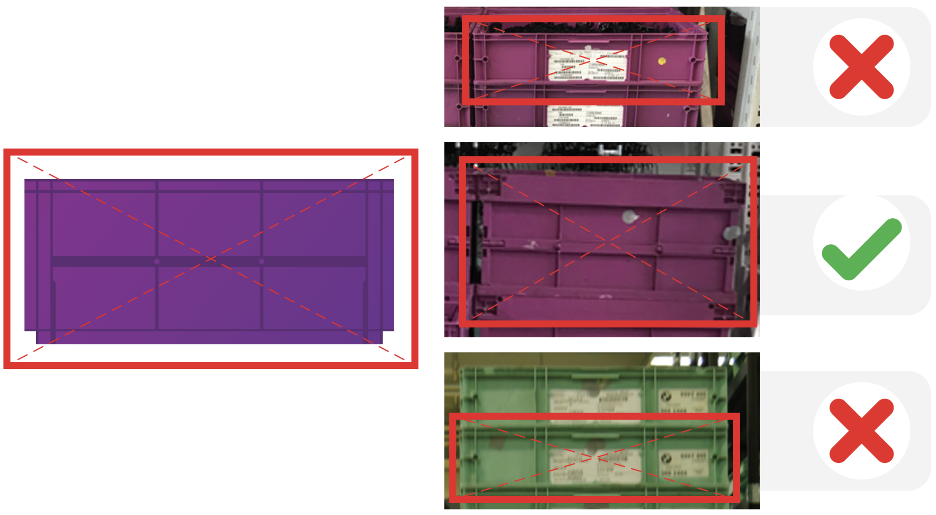


Figure 4.33: Gripping point determination

As the tests based on the representative container clusters show, the choice of the center as the gripping point is not valid since some container types have structure elements or labels there. Here the gripping point would have to be adjusted accordingly. On the one hand, it is noticeable that the angles are detected incorrectly. On the other hand, the repeated performance of the experiment with the same containers shows significant differences concerning the previously determined values. This allows the conclusion to be drawn that both the measurement inaccuracies of the distance measured by the camera and the choice of points to be measured if they accidentally fall on protruding notes or containers, which can never be ruled out, are problematic. This approach will, therefore, not be pursued further.

4.4.4 Experiment 2: Gripping Areas as predicted Objectclass

A second approach for determining suitable gripping surfaces is to consider these surfaces as an additional object class for the "Detection-Module". This is to counter the disadvantage of the center point selection of the previous experiment. To configure the best fitting approach in this section, different tests have been carried out by combining different frameworks and network architectures. To increase the number of evaluated examples, the Selection-Module mentioned above has been deactivated. So the Localization-Module has not only been carried out on the next relevant object but all detected objects in the particular scene.

Training

As mentioned, the experiments in this chapter are based on the already described network architectures. But, as the focus in this application now is more based on accuracy for precise gripping point detection, Yolo has been dropped as its benefits are more in detecting objects correctly or not and not on the pure precision of those detections.

The entire training dataset consists of 500 RGB-D images. They were taken by hand at a distance of $1 - 3m$ from the shelves in the BMW plant in Regensburg. For each container type, approximately 100 images are labeled. The images are labeled with bounding boxes and two types of classes: container (container) and Grip (gripping area). The latter is a plane that can be gripped by a vacuum gripper. Instead of labeling every possible gripping point, the entire possible area is labeled by one single element. This tremendously simplifies the labeling process.

The object detector should only detect grippable containers. Therefore, containers were only labeled as such if they are at least 80% within the image and if the gripping area is fully visible. This way of labeling results in a class balance, meaning that there are as many containers as gripping area classes. Typically, CNNs are trained on hundreds of thousands of images. Therefore this dataset is too small to train a CNN from scratch. Nevertheless, using transfer learning and data augmentation, CNNs can achieve satisfying results.

As described in the previous section, each container is labeled with one single gripping area, because the desired output of the detector is bounding boxes for containers, each with one separate gripping area. To compare detectors for this desired output, the gripping point metric can be used, which is described in the following.

When the algorithm detects a container, it also checks whether a gripping area is identified. For example, if a container is detected with one gripping area and both detections have a sufficiently large Intersection over Union (IoU) with the ground truth data, then the result is three correct predictions of three total predictions. Another example, where a container is localized correctly, but no gripping area is detected, results in one true prediction of two total predictions.

Additionally, detectors can be evaluated for each object class. In essence, the precision for Grip or container can be assessed independently of the other. All in all, four metrics are evaluated. One of them is the established Mean Average Precision (mAP). Additionally, Grips and containers are evaluated separately. Finally, the gripping point metric, most suitable in the given case for comparing the detectors in terms of gripping point detection, is applied.

To improve the training of detection networks, data augmentation methods are applied that artificially enlarge the size of the dataset. The container dataset contains large objects, such as the Foldable-container, and also small objects, such as gripping areas. The data augmentation method "cropping" can significantly improve the detection of large or small objects. To enhance the performance of detecting large objects, a random sub-region of the image can be scaled to the standard input size, creating additional samples of large objects. Likewise, an image can be padded and scaled to the standard input size to create other examples of small objects.

Another unique attribute of the container dataset is the presence of wide objects. To solve this, an aspect ratio of four was included, which is not used in the proposed parameters of SSD or Faster R-CNN. The next section lists which parameters are used for both meta-architectures.

While depth information is beneficial and robust to shadows in contrast to color or gray images, CNNs learn better on more massive datasets, and RGB datasets are significantly larger than RGB-D datasets. As a consequence, Redmon and Angelova pre-trained a CNN on a large-scale RGB dataset and then replaced the blue channel with the depth channel for the training on the Cornell Grasp dataset. This RGD CNN achieved superior results than an RGB CNN.

The containers of the container dataset are all violet, which is a combination of red and

Faster R-CNN Parameters	
Input data	RGB & RGD
Feature extractor	ResNet-50
Pre-trained weights	COCO
Anchor box scales	$\begin{bmatrix}0.25 & 0.5 & 1.0\end{bmatrix}$
Anchor box aspect ratios	$\begin{bmatrix}0.5 & 1.0 & 2.0 & 4.0\end{bmatrix}$
Optimizer	ADAM
Learning rate schedule	Step 0: 10^{-6} Step 300,000: 3×10^{-7} Step 600,000: 0
Batch size	1
Data augmentation	random horizontal flip, random pixel value scale, random adjust brightness, random adjust contrast, random adjust saturation, random adjust hue, random crop image, random crop pad image, normalize image, random distort color, random jitter boxes

Table 4.5: Chosen parameters for Faster R-CNN

blue. Consequently, the green color can be neglected, theoretically. However, it turns out that the green channel contains a lot of information, where the depth replaces each color. The blue channel does not carry as much information as the other color channels do. This approach might improve the standard RGB object detectors. Hence, another CNN model was trained on RGD images with the same parameters, i.e., with pre-trained COCO weights.

While Faster R-CNN is accurate, SSD is faster. For this work, SSDlite (SSD with MobileNetV2 as feature extractor) was also trained on the container dataset. The parameters are listed in 4.6. MobileNet consists of depthwise separable convolutions and is therefore very efficient in terms of time and space complexity. Efficiency is a desirable attribute for a robot because it could then either rapidly detect objects and therefore work faster, or it could use a CPU instead of a GPU and therefore be more economical. Ideally, SSDlite would be sufficiently accurate but with a faster runtime.

SSD was not trained on the container dataset only, but also on the large-scale RGB-D Object dataset to optimize an RGB-D feature extractor. The additional depth information could increase accuracy. The depthwise separable convolutions in MobileNet are a promising choice since they distinguish between color and depth. In addition to MobileNetV2, the successful VGG-16 was trained on the RGB-D Object dataset. The

SSDlite Parameters	
Input data	RGB
Feature extractor	MobileNet V2
Pre-trained weights	COCO
Anchor box scales	$\begin{bmatrix}0.2 & 0.36 & 0.52 & 0.68 & 0.84 & 1.0\end{bmatrix}$
Anchor box aspect ratios	$\begin{bmatrix}0.25 & 0.5 & 1 & 2.0 & 4.0\end{bmatrix}$
Optimizer	ADAM
Learning rate schedule	Step 0: 10^{-6} Step 300,000: 3×10^{-7} Step 600,000: 0
Batch size	24
Data augmentation	random horizontal flip, random pixel value scale, random adjust brightness, random adjust contrast, random adjust saturation, random adjust hue, ssd random crop, normalize image, random distort color, random jitter boxes

Table 4.6: Chosen parameters for SSDlite

RGB-D-SSD Parameters	
Input dimensions	$320 \times 240 \times 4$ & $300 \times 300 \times 4$
Feature extractors	VGG-16
Pre-training on	RGB-D Objects
Anchor box scales	$\begin{bmatrix}0.05 & 0.15 & 0.25 & 0.35 & 0.45 & 0.55\end{bmatrix}$ $\begin{bmatrix}0.05 & 0.20 & 0.35 & 0.50 & 0.65 & 0.80\end{bmatrix}$
Anchor box aspect ratios	$\begin{bmatrix}0.5 & 1.0 & 2.0 & 3.0 & 4.0\end{bmatrix}$
Optimizer	ADAM
Learning rate	10^{-3}
Batch size	32
Data augmentation	random vertical flip, random horizontal flip, random blur, random adjust brightness, random adjust contrast, random crop, salt-and-pepper noise, random depth sensor noise

Table 4.7: Chosen parameters for RGB-D-SSD

parameters are listed in 4.7. One of the data augmentation options is random depth sensor noise, which is introduced in the following.

In random depth sensor noise, some pixels of the depth image becomes 0. This is typical of depth sensors, for example in the Intel RealSense. However, the probability should not be set too high. Otherwise, depth values might be neglected. Therefore, the probability was set to 0.01.

Evaluation

In the entire test, containers from all six positions should be gripped four times, resulting in a total of 24 grips. During the test, the following errors occurred:

- **No detection:** No container or gripping area respectively was detected.
- **Mechanical gripper problem:** An unfortunate error of the robot, where the air pressure of the gripper leaked. Hence, it was not able to grip the objects. Nevertheless, this was not an error of the vision system.
- **Wrong container:** The wrong container was gripped. This happened, for example, when the left middle container should be chosen, but only the middle right container was detected. The right container was then also the most left. Thus the wrong container was gripped.

Another test shows how well the vision system can handle a variety of objects. All five different types of containers were tested. The experiment was conducted in the laboratory in Munich. There were four shelf positions, two in each of the two levels. Apart from these differences, the test was similar to the previous test. Each type of container was gripped four times resulting in 20 grips.

To evaluate the object detector, the precision of it was measured on the established metrics: AP50, AP75, and mAP. Additionally, the precision of the gripping areas and the containers were evaluated both separately and together, and if a container was detected together with its gripping area, the intended result was reached. Consequently, 4 different types of precision were evaluated, on three different thresholds: IoU 50, IoU 75, and mean IoU@[.5,.95,.05]. The detectors were evaluated on a test set, including 50 images from the BMW plant in Munich. Last but not least, the runtime was measured, not only for the detection but also for the entire localization process.

Meta-architecture	Faster R-CNN			SSD
Feature extractor	ResNet-50			MobileNetV2
Trained on **Tested on**	RGB RGB	RGD RGB	RGD RGD	2*RGB
Established Metrics				
AP50	0.895	**0.921**	0.838	0.879
AP75	**0.854**	0.846	0.766	0.728
mAP	**0.734**	0.706	0.666	0.639
KLT Detection				
AP50	0.932	**0.942**	0.847	0.914
AP75	0.916	**0.92**	0.829	0.906
mAP	**0.797**	0.77	0.732	0.746
Grip Detection				
AP50	0.812	**0.892**	0.82	0.833
AP75	0.717	**0.75**	0.665	0.483
mAP	0.589	**0.624**	0.567	0.493
Gripping Point Metric				
AP50	0.939	**0.957**	0.927	0.842
AP75	0.916	**0.927**	0.896	0.763
mAP	**0.809**	0.786	0.789	0.657

Table 4.8: The precision of the convolutional detectors

Two of the four presented architectures are based on the SSD meta-architecture. One SSD model was trained on RGB data and is called SSDlite, and the other one was trained on RGB-D data and is called RGB-D-SSD. The different detection models are based on the Faster R-CNN meta-architecture. One Faster R-CNN model was trained on RGB data, and the other one was trained on RGD data. However, the latter was not only tested on RGD data, but also RGB data. The prefix of the name indicates on which data the model was trained, and the suffix indicates on which data the model was tested. For example, FRCNN RGD2RGB is the Faster R-CNN model that was trained on RGD data and tested on RGB data.

Replacing the blue channel with the depth channel is an inadequate approach since the blue information gets lost. Additionally, the blue color distribution differs from the depth distribution. The first layers of the network were not optimized for such an input since only the weights of the last layers are changed in transfer learning. Additionally, as the noise in the Intel RealSense is significant, the RGD images are highly distorted. As a result, the *FRCNN RGD2RGD* performed worse than the *FRCNN RGB2RGB*.

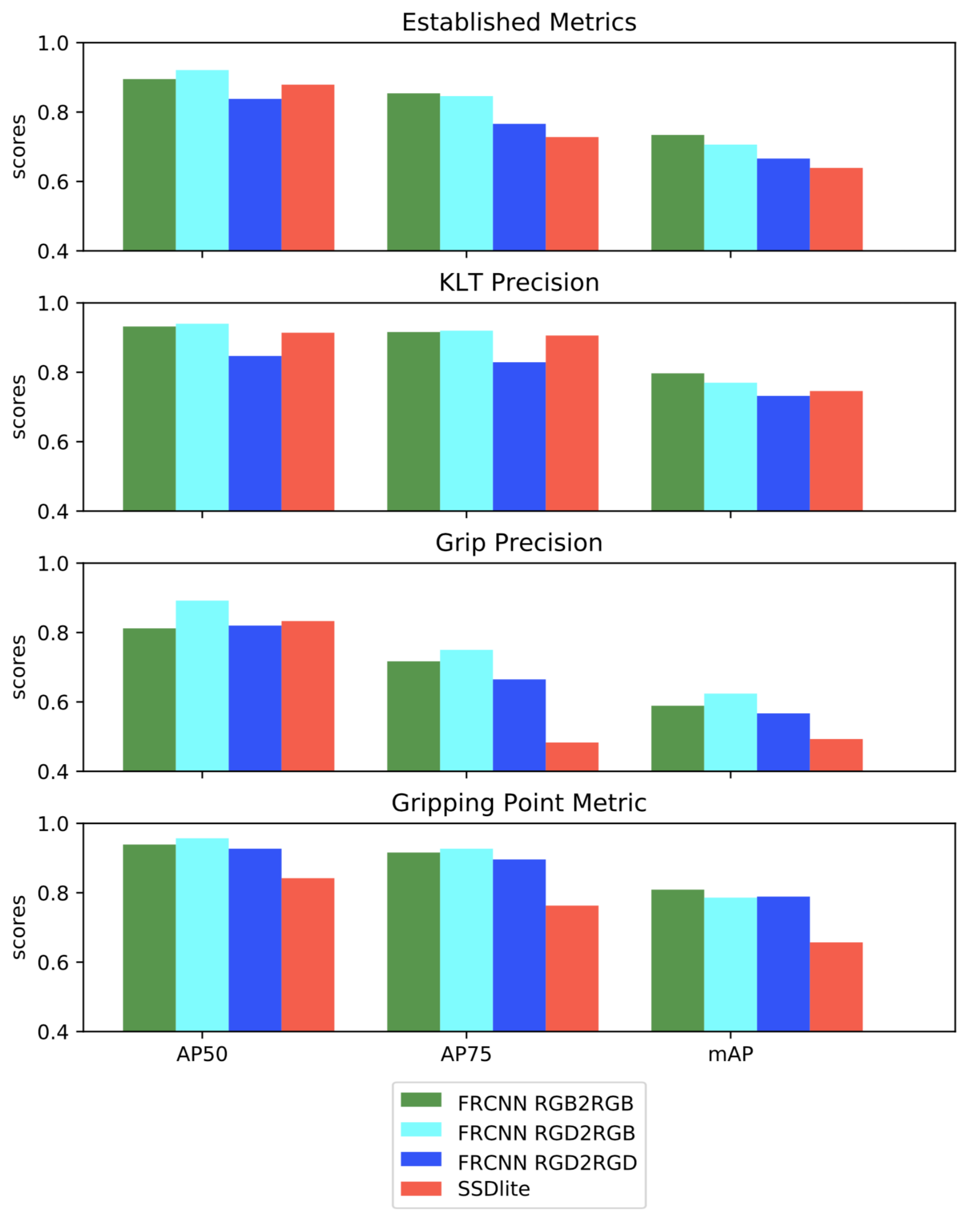


Figure 4.34: The precision of the convolutional detectors (own illustration)

Surprisingly, *FRCNN RGD2RGB* performed better than *FRCNN RGB2RGB*. 4.35 illustrates that a gripping area was detected by the *FRCNN RGB2RGB* even though it is not fully in the image. Hence, the performance was evaluated worse than the *FRCNN RGD2RGB*. However, without doubts, the area that was detected is a gripping area. Moreover, the data augmentation method cropping may also put cropped objects at the image boundary. As a result, changing the labels of the dataset is recommended. In essence, objects should be labeled even though they are not fully visible in the image. As a consequence, the detector might detect more objects and especially also objects that are not entirely in the image. For this reason, more validation methods must be added to convolutional object detectors.

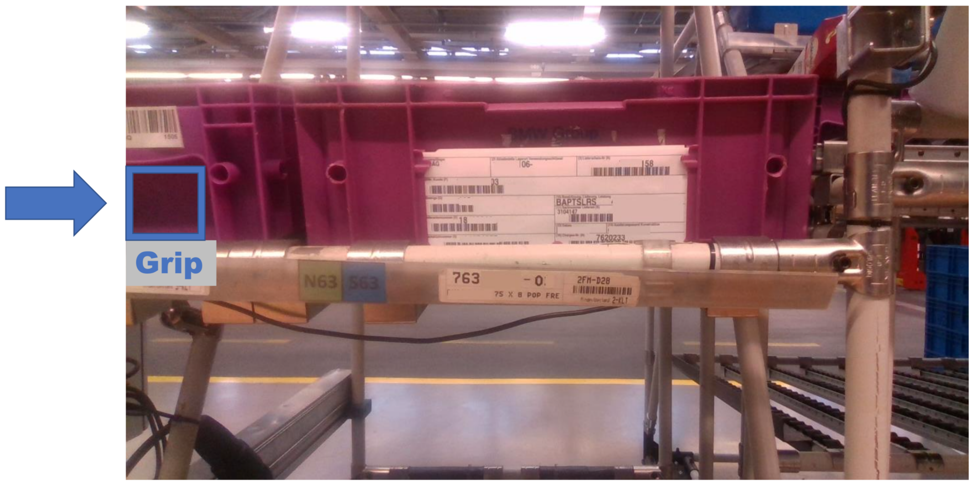


Figure 4.35: Detected gripping area that is not fully visible in the image (own illustration)

As the results show, an advantage of this procedure is that the distance of the object can be calculated not only at one point, but over the entire resulting bounding box as an average value. Thus the independence against noisy measurement data can be increased. However, as the results of the experiment show, no net can be trained with reasonable manual effort that can meet the high precision and robustness requirements of this module. In the case of small containers, for example, which have a comparatively high proportion of the total material flow, this approach works very stably. As already mentioned those kinds of Deep Neural Networks are not 100 % accurate. By adding the gripping region as an additional class, the error-rate caused by the "Detection-Module" would increase. And in parallel, the possibilities to improve the overall robustness are limited to changing training parameters even further (only small improvements realistic - if at all) or to drastically increase the training data.

Additionally to the just mentioned disadvantages when detecting the gripping points from the side, the gripping pose detection from the top of the boxes is not feasible as those bounding boxes are always vertically and horizontally aligned to the image borders. So there is no possibility to get the object orientation out of it.

Drawing all those results into conclusion, this attempt will not be focused further.

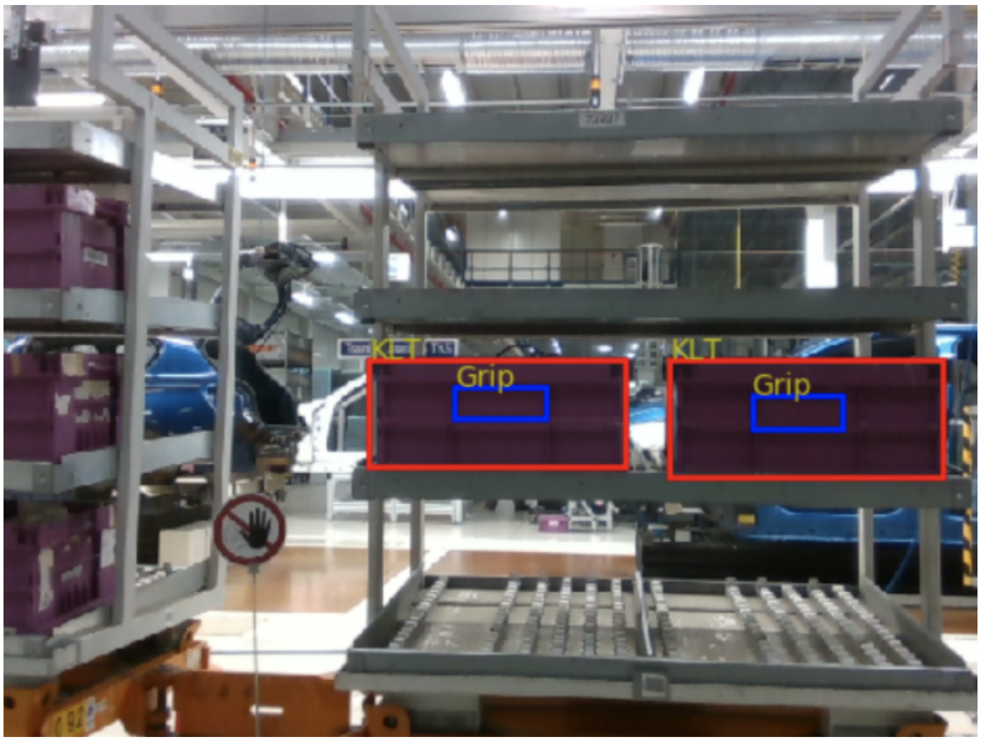


Figure 4.36: Image from the testing of this approach for gripping pose determination (own illustration)

4.4.5 Experiment 3: Gripping Area as Result of filtered Depth Images

The final approach is to determine the gripping points of the respective objects by analyzing the recorded depth images. Three-dimensional depth images do not have the same quality as RGB images and, depending on the object material, can be subject to medium to high noise levels. To do this, they pass through several filter stages. Otherwise, the overall performance of this approach would drop drastically, and the whole approach could not be used for an industrial application. [HST13]

It should also be noted that, depending on the application, two different implementations for gripping pose detection are required, which, have the same generic approach. Accordingly, the object grip pose is determined in the following subchapter if the robot looks at the object from above. The gripping pose determination algorithm for a lateral view of the object is dealt with in the second subchapter. Since both in literature and applied practice, there are many different ways to analyze images through the use of filters, several approaches are discussed in the following chapter. The most suitable combination is finally selected at the end of this chapter.

Top View

The first case that is described here is calculating the gripping pose when looking onto the scene from a top view. This can detect the container and determine its orientation when looking at the object from above in a depth image taken from above looking at the container. The algorithm can be used, for example, in palletizing to identify the gripping container pose on the delivery stitch and determine their orientation. The input is a depth image, the camera information, and a time stamp. The width and depth of the container, the center point (x, y, z) and the angle of twist alpha are returned as return values. Alpha represents the angle between the lower edge of the image and the longer side of the container.

Image Preparation Before the respective gripping points can be determined, it is necessary to prepare the recorded images. This image pre-processing serves to manipulate an original image without losing relevant information. Rather, errors in the image are to be compensated by smoothing, and certain image areas are to be emphasized more strongly so that the object recognition provides better results in a later step. Especially with depth images, it can happen that the camera cannot determine a value for every image pixel. These pixels are colored black. To compensate for these errors and noise in an image, the image can be smoothed. It is important here that the smoothing changes no edges. Various filters are used for this purpose. To calculate the resulting pixel, these include the surroundings of the pixel in the original image. Filters can be divided into linear and non-linear filters. Linear filters are characterized by a lower computational effort, which is based on inverse transformation. Low pass filters are used to smooth an image. These include the mean value filter and the Gaussian filter. These are briefly explained in more detail below. [TSV05]

Mean Value Filter The mean value filter consists of a filter core. This matrix can be of any size. The larger the filter, the more surrounding pixels are included in the calculation and the blurrier the resulting image becomes. For each pixel, the values of the core pixel are then multiplied by the values of the underlying image pixel. These results are then added for the new pixel value. The filter is then moved to the next pixel. The average filter suppresses noise, but the edges are more or less blurred, depending on the size of the filter [Haa08].

Gaussian Filter The Gaussian filter is used in the same way as the average filter. However, the entries of the filter core matrix are determined from the Gaussian function. The advantage of the Gaussian filter is its independence of direction [Haa08]. Depending on the type of noise, linear filters do not work. Adaptive Gaussian noise can easily be eliminated with an average filter. However, if non-additive salt-and-pepper noise is present, the performance of linear filters is poor. Another possibility is the use of non-linear filters. This category includes, for example, the median filter and morphological operations.

Median Filter The median filter uses a mask. This is moved from pixel to pixel. To determine a new gray value, all original values that lie within this mask are used and sorted according to their size. Then the median - the middle pixel - is taken as the new gray value. Due to this approach, this filter is particularly well suited for binary noise. [Kra04]

Morphological Operations In morphological operations, a distinction is made between the two basic operations erosion and dilatation and their combinations opening and closing [Soi98]. In erosion, a filter core is guided iteratively over each pixel in the image as a 2D convolution. For each pixel in the original image, it is checked whether all surrounding pixels are within core 1, then the resulting pixel also becomes 1. If this is not the case, the resulting pixel becomes 0. This results in a scaled-down object. Dilatation is opposite to erosion, but not inverse. It leads to an enlargement of the object. Here the resulting pixel is exactly 1 if an environment pixel is in core 1. Thus it is possible to close holes and cracks in objects with the dilatation, as well as to unite different objects to a large object. However, the disadvantage is that the object becomes larger. If an image contains noise, the error points can first be removed by erosion. This makes the object narrower. To counteract this, dilatation can then be applied. This scales the object back to its original size. However, parts smaller than the filter cannot be restored. This combination is called opening. If dilatation is applied first and then erosion, this is referred to as closure. The object is first enlarged so that holes are closed. In the following step, however, the object with the erosion is reduced in size again. Figure 4.37 shows that in this case, the morphological operation Close produces the best result. This is due to the nature of the recording error. The recording error acts like salt and pepper noise, which is particularly well smoothed with non-linear filters. In this case, the morphological operations were not applied to black-and-white images as usual, but grayscale images. They function like the median filter as ranking operators. The result pixels are not replaced with 0 or 1, but with the smallest or largest value from the filter core. The combination of opening and closing can be used to smoothen a greyscale image [Soi98]. The results are shown in Figure 4.39. This makes it obvious that the small black dots could be removed from the original image by the described procedure.

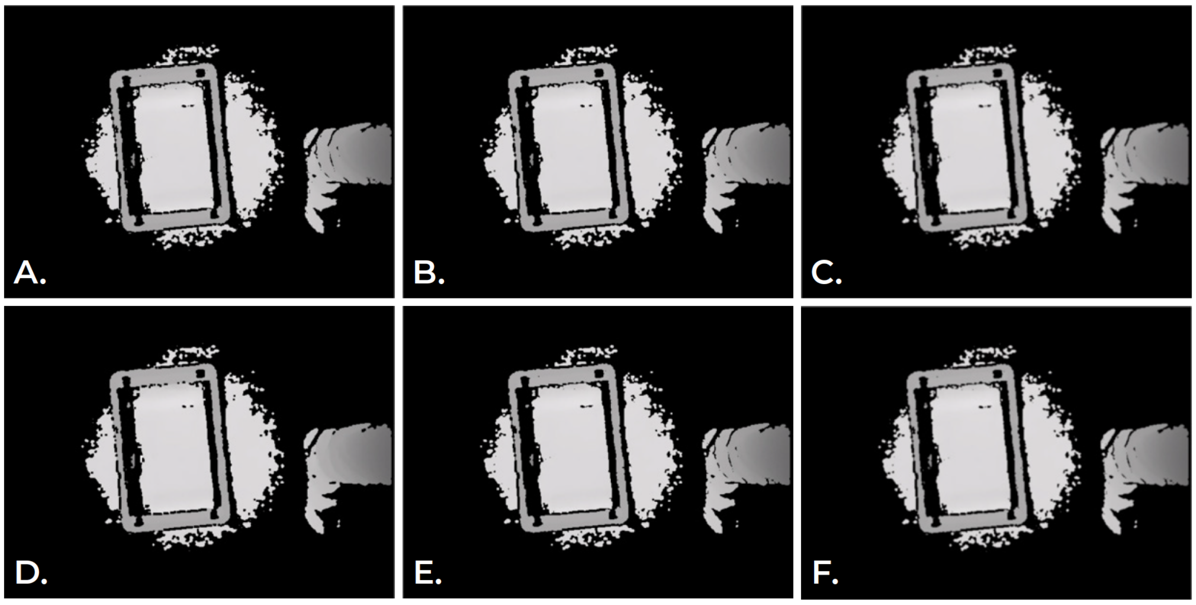


Figure 4.37: Visualization various applied filters to captured depth images (own illustration). A) Original Image, B) Mean-Filter, C) Gauß-Filter, D) Median-Filter, E) Closing, F) Opening.

Segmentation The threshold method is used to segment interesting and uninteresting areas. Since it is known in the present scene that a container stands on a surface such as a roller conveyor when palletizing empties, and the frames of the robot may also be in the picture, the threshold value method is ideal.

One of the most straightforward variants of the threshold value method is the one with a global threshold value. A gray value image with a constant gray value threshold is converted into a binary image. For each pixel, it is checked whether it is smaller or larger than the gray value threshold and set accordingly to 0 or 1. The threshold value must always lie between object and background. Otherwise, both background and object would disappear.

Since the objects in the image are always at a different distance from the depth image camera, no constant value can be taken in this case. Instead, a separate value is calculated for each image. Figure 6.5 shows the original image and its histogram. A histogram shows how many pixels have which gray value and therefore gives an indication of where the optimal threshold value must be to separate the foreground from the background. In x-direction, all grey values are plotted, which are found in the picture. To better recognize the relevant parts of the histogram, all pixels with the value 0 were excluded from the histogram because they do not contain any information about the scene. The y-axis shows the number of pixels with the respective gray value. In the present case, it is known that the container stands on an underground. This can also be seen in the histogram: the conspicuous maximum around 220. Since the ground is always furthest away from the camera, it can be removed by an upper threshold. This threshold should be placed between the two discrete maxima, as the second maxima represent the edge of the container. Since the container is empty, the bottom of the container is also removed. As the object detection is based on the detection of the container edge, this is no problem. Also, the maximum height of the container is known so that everything above it can also be removed. For this reason, no classical threshold method is applied. Instead, a threshold value procedure with two threshold values was carried out. The two threshold values were calculated using the mean value from all image pixels. Then all pixels with a value higher than the greater threshold value were set to 0, and all pixels with a value less than the smaller threshold value were also set to 0. All pixels in the middle between the two thresholds are set to 1. This is shown in the bottom half of Figure 4.39.

Object Recognition After the image has been smoothed and then segmented, the next step is to find the desired object in the image. There are various possibilities, which are presented in the following: Get lines method, template matching, and a method based on contours. Once the object has been found as such, it can be analyzed to see how it is aligned and where the center is.

A new approach uses an innovative subspace clustering method for edge detection based on K-Means. At first, a Canny edge detection is applied. Straight lines are then found in the detected edges using Hough Lines. Then the center points and the slope of these lines are calculated. These properties are then used to cluster the sides of the containers. First, the lines are divided into two groups according to the gradient: Long side of the container and short side of the container. The two groups are then treated separately and spatially separated from each other. The center of the four subspaces created is then determined. A rectangle can be constructed with the help of the gradient. This can be seen in Figure 4.40.

In template matching, a complete search is used to find a predefined template in a larger

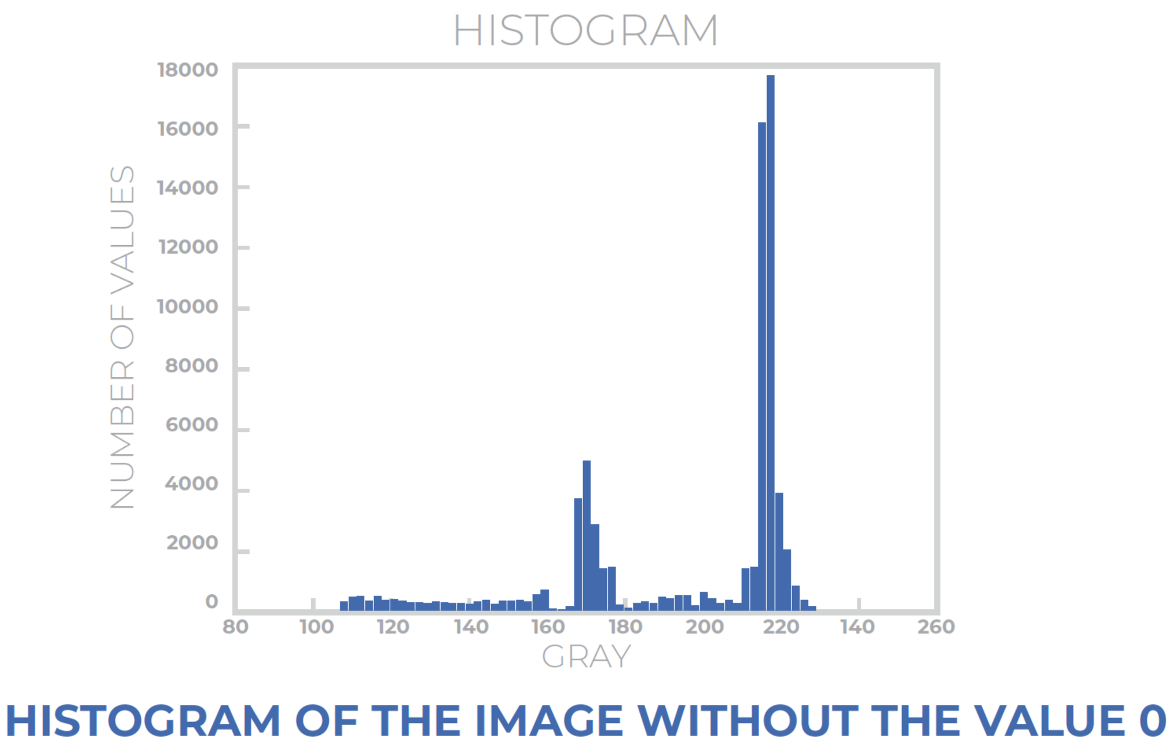


Figure 4.38: Image Histogramm (own illustration)

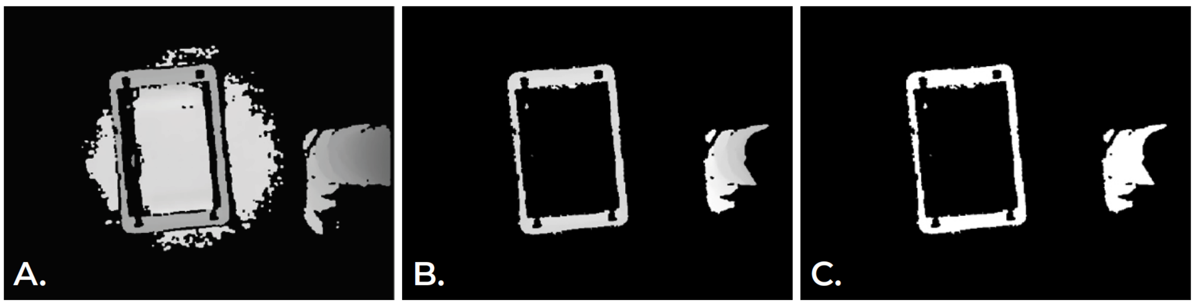


Figure 4.39: Foreground background separation by thresholding (own illustration). A) Smoothed Image, B) Image after the application of lower and upper thresholds, C) Final image after setting all pixel values in those boundings to 1.

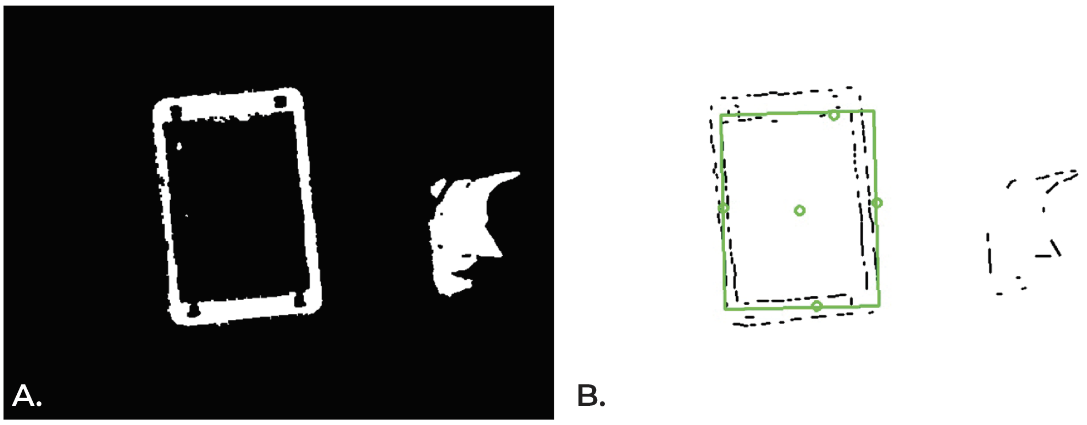


Figure 4.40: Angle detection by k-means subspace clustering (own illustration). A) Image in which the object should be found, B) Canny edge-detection and Hough Lines with the detected object by subspace clustering.

screen. The template is moved over the input image, and the template is compared with the subarea of the image. How well a template fits into an image region can be expressed by the correlation of a template with an image f(x,y) [GW17]. There are different methods to compare the template with the image section. Template Matching searches for precisely the same template - same size and same orientation. Since the container can lie on the belt in any position and orientation in the application case, the template must be rotated during the search. Thus the maximum value is searched for per rotated template. These values are then compared with each other, and again, the most significant value is determined. If the largest value of all templates is found, the conclusion can be drawn as to where and in which orientation the searched object can be found in the image. Figure 4.41 shows the difference between two differently rotated templates. The original image shows the object in 95. This can be seen in the partial image (c) since the brightest pixel can be found here.

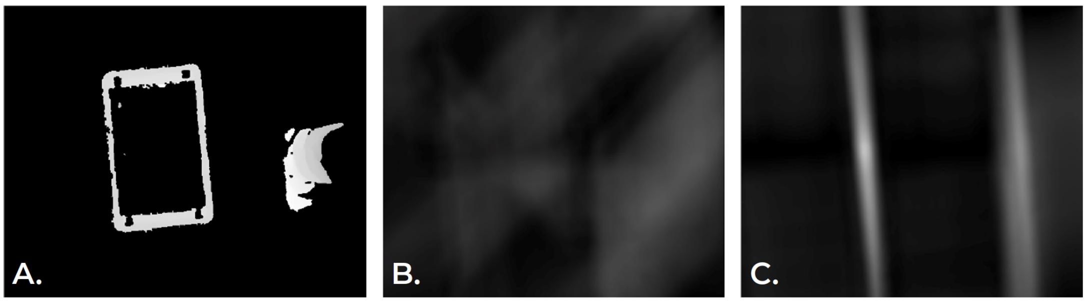


Figure 4.41: Template matching results (own illustration). A) Image in which the template should be located, B) Result of template matching with a template angle of 45 degrees, C) Result of template matching with a template angle of 95 degrees.

Another way to detect objects in an image is to focus on contours. A contour is a curve that encloses all pixels connected by similar intensity. OpenCV [Ope18] contains the function cv2.findContours(), which was implemented after Suzuki [Tel85]. Input images are handled as binary images. The function can be used with different modes and methods. The modes can be used to decide which contours are to be found. It is possible to find only the external contours or to find further contours within these contours. In this application, it is only necessary to find the outer contours of the container. Therefore cv2.retrExternal was used as the mode. The approximation method can also be selected:

- to get all the points
- to get only points that were combined to horizontal, vertical and diagonal - so that a rectangle consists of only the four corner points
- or the Teh-Chin chain approximation algorithm [Chi89].

To be applied. In this case, cv2.chainApproxSimple was used, which returns only the four border points.

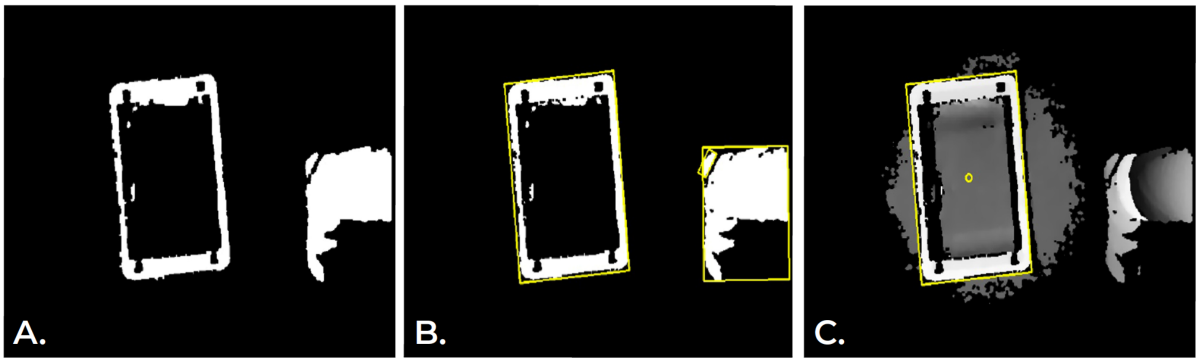


Figure 4.42: Angle determination via contours (own illustration). A) Image in which the template should be located, B) Detected Contours by cv2.findContours(), C) Final result in the original image.

If there are several objects in the input image, the contour function also returns several contours. This can be seen in Figure 4.42 (b). Various test series have shown that in this application, it can be assumed that the largest object in the image is the relevant one in the specific application. For this reason, the different contours can be checked to see how large the enclosing surface is. This can be done with the OpenCV function cv2.contourArea(). The contour with the largest surface area is used for further machining.

Once it is known which contour to use, the function cv2.minAreaRect() can be applied. This is responsible for finding a minimum rectangular area that surrounds all points of the shape. The rectangle can be rotated to get the smallest possible area. The function returns the center point of the rectangle, the width, and height as well as the rotation angle. Figure 4.42 (c) shows a rotated rectangle around the contour with the largest area and its center.

The function cv2.boxPoints() is used to get the corner points of the minimum rectangle from the center point, the angle and the height and width of the rectangle. The rotation angle is always specified between the parallel line to the lower edge of the image, through the lower corner of the rectangle and the right corner of the rectangle. In this application, however, the angle between the lower edge of the image and the longer side of the box is required. Therefore, the system first checks whether the right-hand corner belongs to the long or short side of the rectangle. If it belongs to the long side, the angle is taken over, otherwise 90 are added.

Sideview

As in the top view gripping pose detection, the image is first processed during lateral gripping pose determination. In the first step, regions in the depth image that are irrelevant for the gripping point are removed. The Bounding Box of the Detection-Module allows to know where the object is located in the RGB image. Therefore, in the depth image, all areas outside the bounding box of the container are removed first. The result can be seen in Figure 4.43.

To remove the stripes that form during rectification from the image, the depth image is smoothed by morphological closing. This is illustrated in Figure 4.43 (a). Although the neuronal network has already limited the region in which the container is mapped, it is still possible that parts of the background or foreground can be seen. These are removed with upper and lower thresholds, as explained above, see Figure 4.43 (b). The holes which have arisen during the threshold value procedure are then reduced or removed again by morphological closing (Figure 4.43 (c)).

To obtain a higher-contrast image, histogram equalization was performed, as shown in Figure 6.13. This leads to a spread of the gray values from 0 to 255. Since the jump between the values 0 and the smallest value is relatively large, all 0 values are set to the smallest value first.

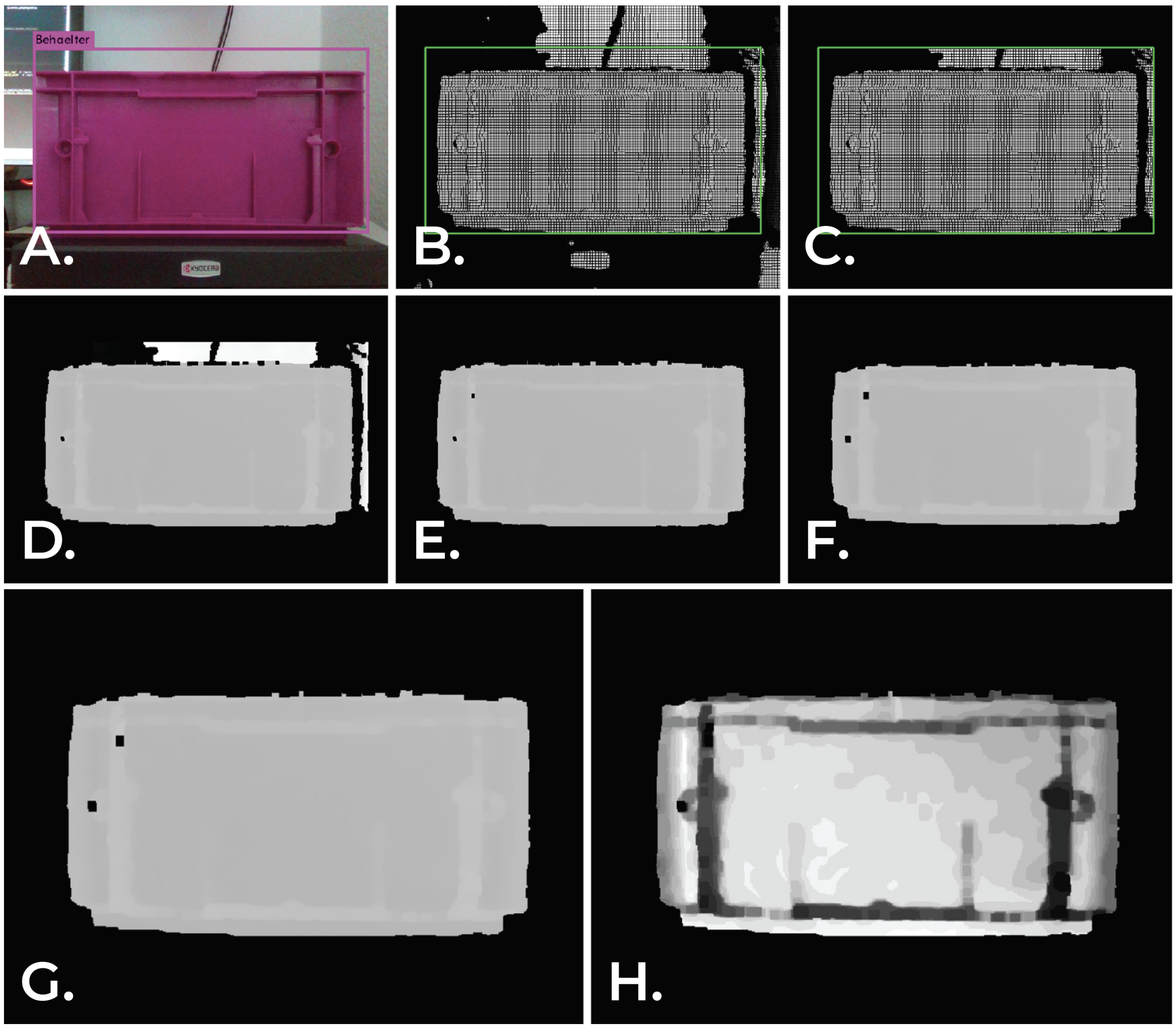


Figure 4.43: Applied depth filters for side view gripping point determination (own illustration). A) Bounding Box predicted by Neural Network, B) Depthimage with Bounding Box, C) Removed border of unimportant image information, D) Morphological Closing, E) Remove Background, F) Morphological Closing, G) Image before Histogram Equalizer, H) Image after Histogram Equalizer.

The Region of Interest (ROI) within the container has the property of a flat surface. To find these, another threshold value is used after the histogram compensation. Since the flat surface is not necessarily parallel to the camera, it can be represented by a gray value gradient. Therefore, different threshold methods were considered. The differences between a normal and an adaptive threshold can be seen in Figure 4.44.

Figure (a) shows the result with an average as the threshold. In (b), however, the median was used. Figures (d) and (e) show the adaptive threshold. The adaptive threshold method calculates a threshold value for a smaller region of the image. Thus one receives different threshold values for different regions for a picture. In OpenCV

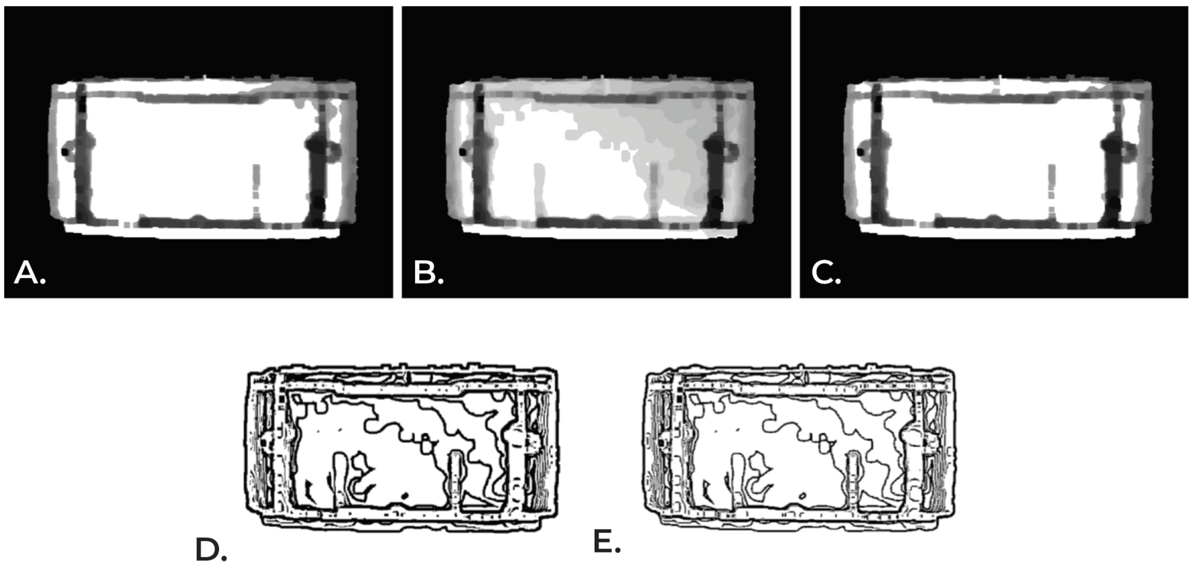


Figure 4.44: Methods for limit calculation (own illustration). A) Mean, B) Median, C) 4-Zone threshold with mean value, D) adaptive threshold with mean value, E) Adaptive threshold with Gauß.

there is the function cv2.adaptiveThreshold() for this. This can be performed using two different methods of threshold value calculation: "cv2.adaptiveThreshMeanC", "cv2.adaptiveThreshGaussianC". The first method uses the mean value of the environment. The second method uses the weighted sum of environmental values. (c) shows a simplified adaptive threshold method. The region of the box was divided into four squares. The individual mean value per square was then calculated and used as the threshold value for the region. Figure 4.44 shows that for this application, the mean value and the 4-zone threshold provide the best result. Since the 4-zone threshold does not make a significant difference, it was omitted, and only a simple mean value was used.

Edges are very distinctive image features, so an edge detection filter is applied according to the threshold method. Edges are relatively small areas in which the intensity of the gray values changes significantly. If one takes the derivative of the image function - the slope - these edges can be recognized as local maxima. There are various filters, including the Sobel operator, the Laplace filter, and the Canny algorithm [Pet06].

The Sobel operator [Ope18] is a first-order edge detection filter. The gradient image is generated by folding the matrix and the original image. In this image, high-intensity areas represent a significant change in the original image. The higher the intensity in the gradient image, the stronger the edge was in the original image. The Sobel operator can be used to determine edges in the x and y directions.

The Laplace filter [Ope18] is a second-order method. The extremes of the first derivative are not considered, but the zeros of the second derivative. If the derivative of a function is formed, the extremes of the function become zero points of the derivative.

The Canny Edge Detector implementation of OpenCV [Ope18] aims at three main objectives:

- low error rate: good detection only of existing edges
- good localization: minimal distance between the calculated and real edge
- minimum resonance: only one detection edge per real edge

First, a Gaussian filter is used to remove any noise. In the second step, the intensity gradient of the image is found analog to the Sobel filter.

Here the angles are rounded to the four angles 0, 45, 90 or 135. To ensure that only the real edge is determined, all pixels next to the edge are then removed. This leaves only a single, thin edge. This method is called non-maximum suppression. The final step is hysteresis. Thresholds are used to filter the edges to determine whether they are accepted as edges or not. An upper and a lower threshold value are used for this purpose. All edges above the upper threshold are accepted as edges. All edges below the lower threshold are discarded. Edges that lie between the two threshold values are only accepted as edges if part of the edge lies above the upper edge. If the edge lies exclusively between the threshold values, this edge is also discarded. The results of the various filters in this application are shown in Figure 4.45. This indicates that the Sobel operator works best. The Sobel operator is calculated in x- as well as in y-direction.

Contrast-limited adaptive histogram compensation is then performed. To exclude weak edges from consideration, a further threshold value method is then applied. The threshold value is again calculated from the mean value of all values not equal to 0. The edges in the resulting image contain a few contiguous lines. To eliminate smaller lines, erosion is applied first. A morphological closure is then used again to close the holes. Finally, the image is traversed in the x and y directions and areas smaller than 30 pixels are filled in. These steps are shown in Figure 4.46.

At this point, the image has been prepared to such an extent that it is possible to search for an area in which the gripper can build up a vacuum. To determine a suitable grip

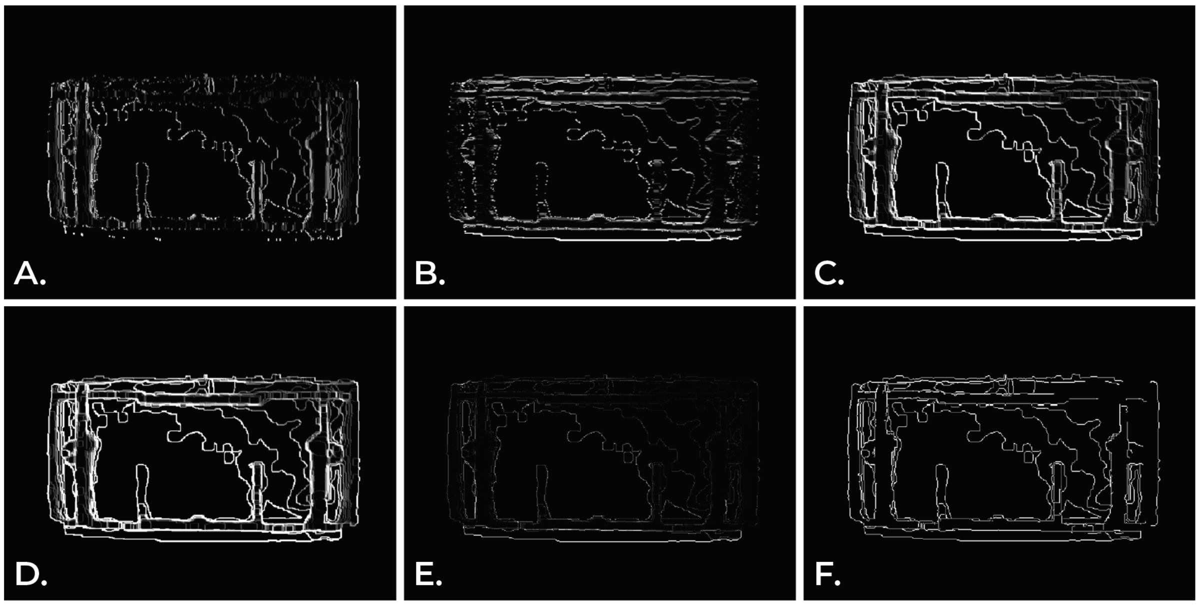


Figure 4.45: Applied filters for comparison (own illustration). A) Sobel in the x-direction, B) Sobel in the y-direction, C) Both Sobel directions combined, D) Enforcement by adding Sobel in the x-direction, E) Laplace-Filter, F) Canny-Filter

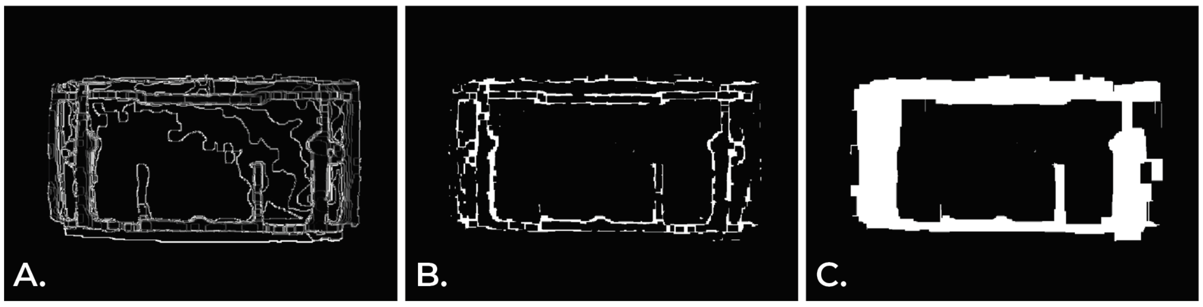


Figure 4.46: Preprocessed images with processing steps to close borders (own illustration). A) Contrast limited adaptive histogram equalization, B) Erosion and morphological closing, C) Closing of smaller gaps.

point, it is necessary to know how large the area on the container should be. The vacuum suction cup has a width of 81 mm and a height of 42 mm. However, since this may be changed in the future, this dimensioning is initially regarded as variable here. A safety tolerance of several millimeters is added to this size. From these data, it is therefore known how large the vacuum suction bell is in the camera coordinate system. Since it can also be assumed that the suction cup is to rest on the container, the mean depth of the container can be used for the calculation. The x-, y- and z-coordinates of the corner points of the suction cup can be found in the camera coordinate system. From this, the size of the suction cup in the image coordinate system can be calculated with the help of the intrinsic camera matrix. It is now known how large the flat surface in the image must be for the suction pad to fit into reality. To ensure stable gripping, the gripping point should be in the middle of the container. Therefore, a rectangle of the required size is first placed in the middle of the container. Based on the edges of this rectangle, you can now decide in which direction the rectangle must be moved so that no white pixels are to be found in the rectangle. Figure 4.47 (a) shows the original rectangle in pink. It can be seen that the rectangle is placed too high and could not grip due to the upper edge of the container. Therefore, the rectangle is moved downwards. In red, the rectangle can be seen after the displacement.

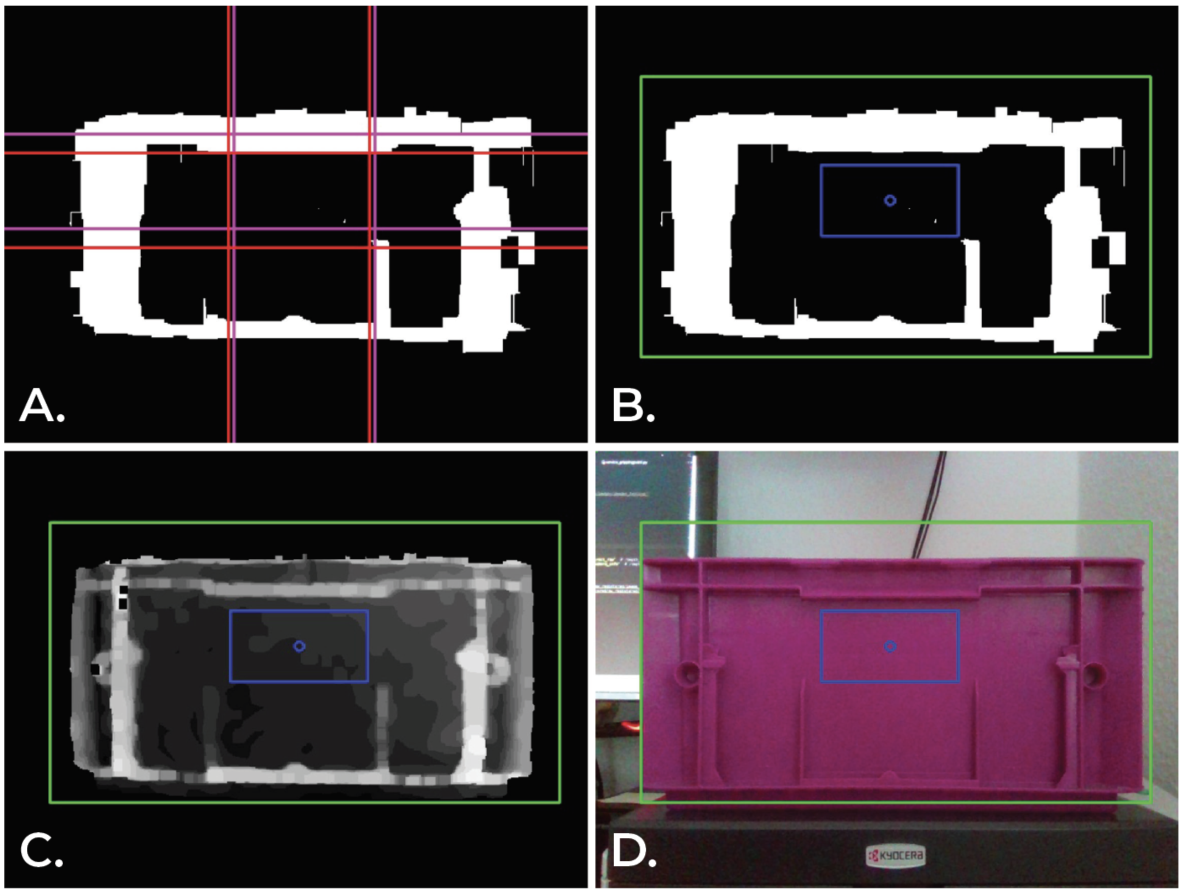


Figure 4.47: Final execution of gripping area analysis (own illustration). A) Searching for a rectangle, B) Suction cup area, C) Suction cup area, drawn in the original depth image, D) Suction cup area, depicted in original RGB image

In summary, Figure 4.48 shows a pictorial representation of the flow logic of the gripping point algorithm.

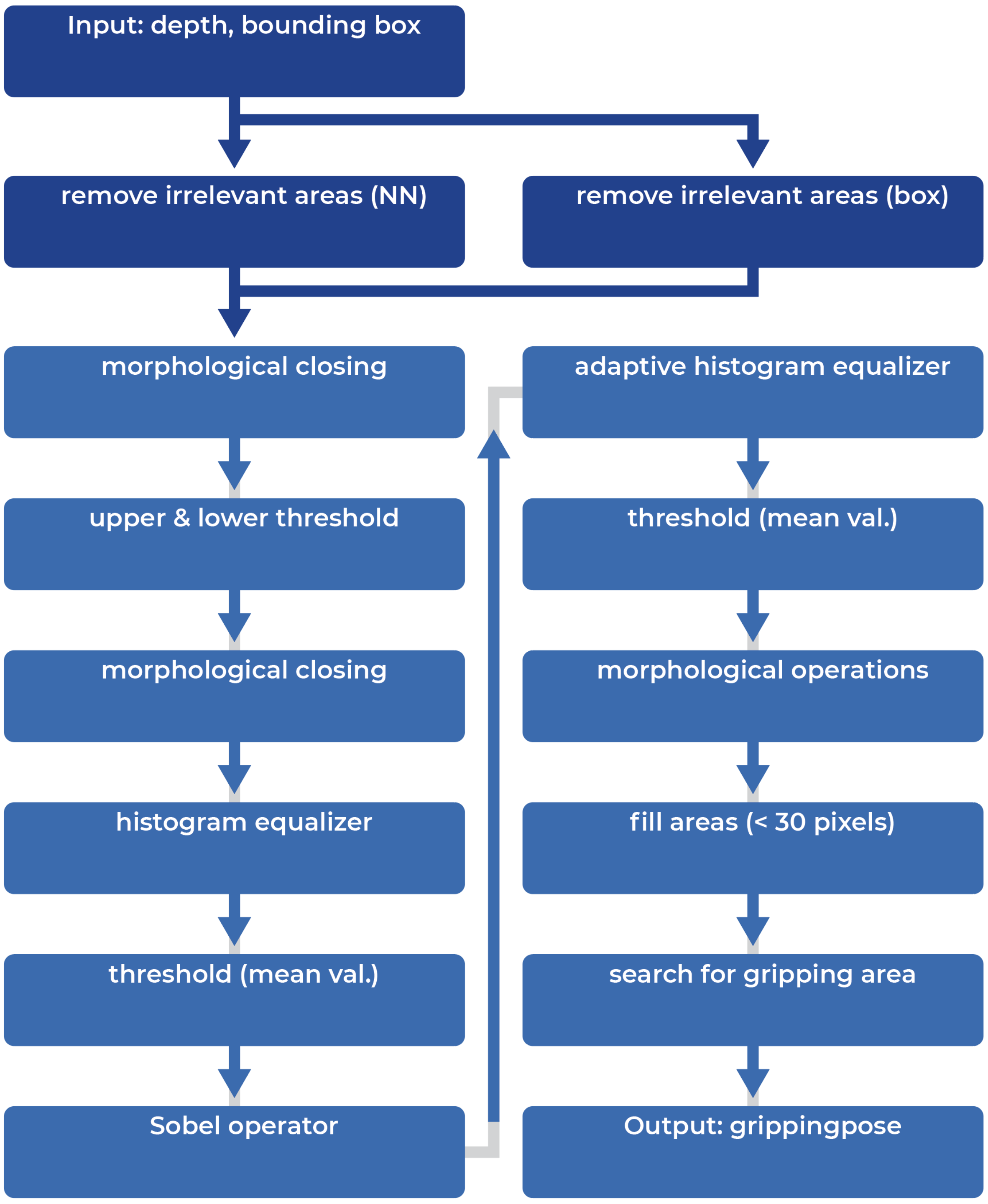


Figure 4.48: Summary of necessary steps for gripping point detection (own illustration)

4.4.6 Implementation and Evaluation

Finally, the presented variation possibilities of the gripping pose determination from the perspective of the top are tested to identify the implementation of the performance. As already mentioned in the objective, the accuracy, as well as the computing time of the algorithm, are the main decision criteria.

Top View

First, the top view gripping pose detection approach is evaluated. The following structure was chosen for the recording of the test data. The camera was mounted at a distance of 0.78 m from the ground with the direction of view. A container was placed on the floor in the middle of the image and rotated by 5 degrees from 0 degrees to 175 degrees. Since the container is symmetrical, a rotation of 360 degrees is not necessary. For all images, the same container was used, so that in this experiment, only the angle was changed. For each angle, 50 images were taken, and the angle of the container was determined. An evaluation image with an angle of 45 degrees is shown in the Figures 4.49, 4.50 and 4.51.

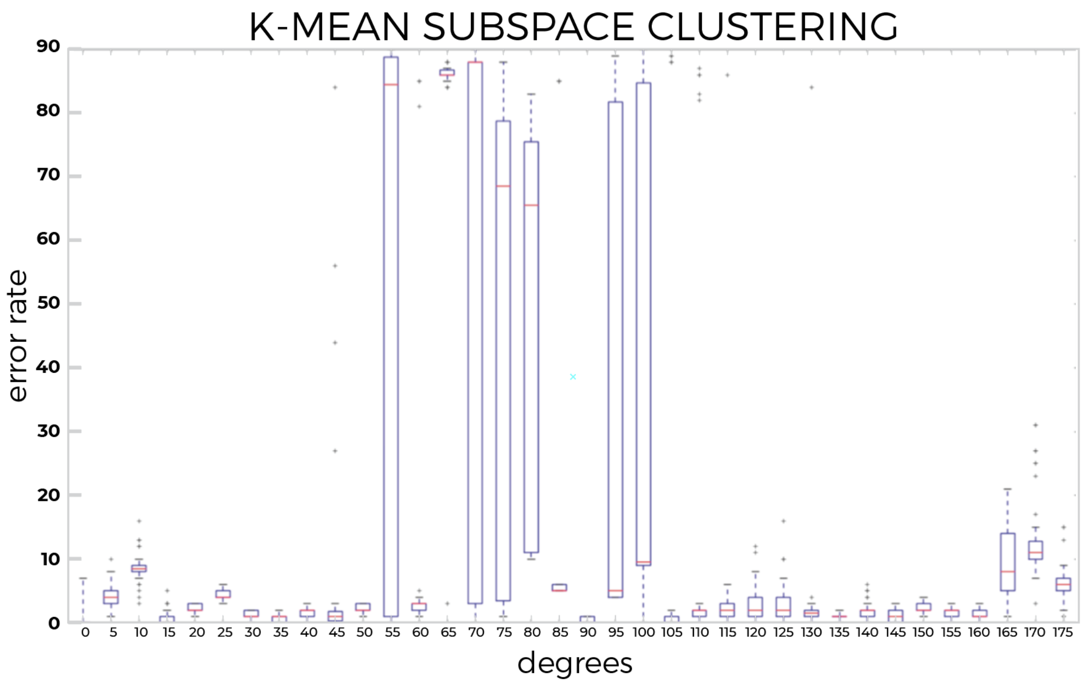


Figure 4.49: Results comparison k-mean subspace clustering(own illustration)

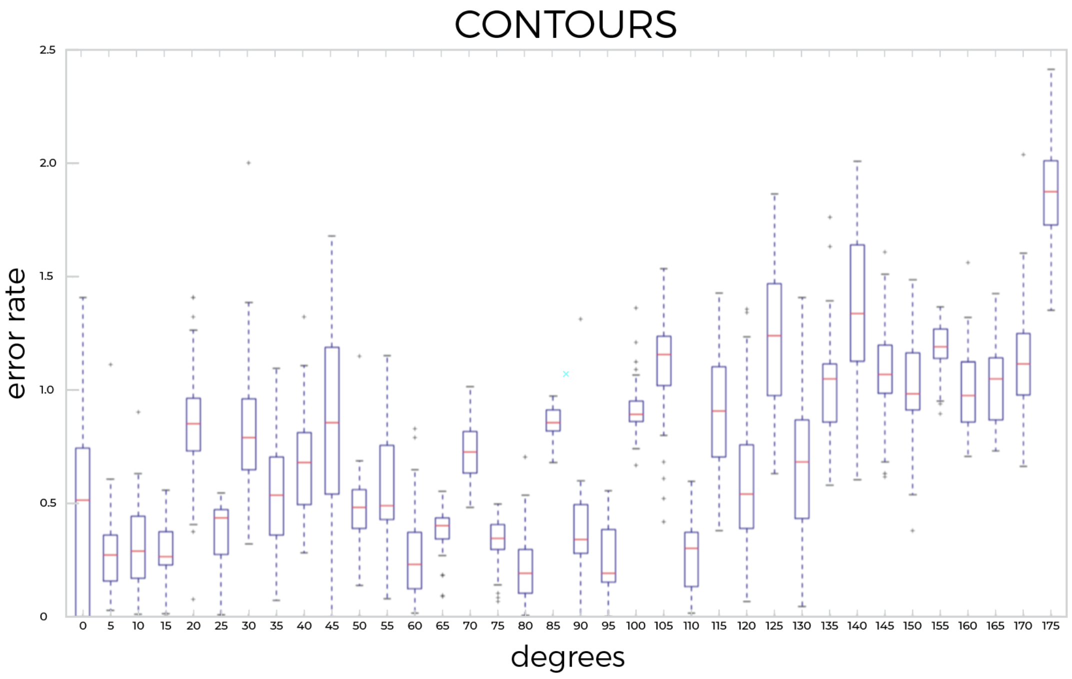


Figure 4.50: Results comparison contours (own illustration)

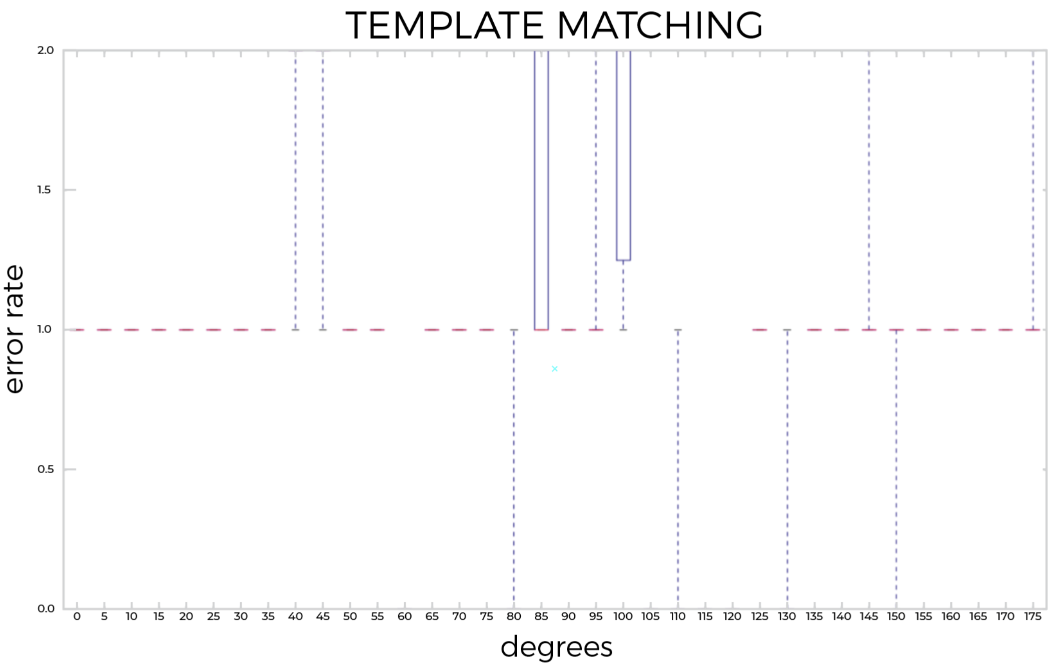


Figure 4.51: Results comparison template matching (own illustration)

To assess the accuracy of the angle detection, Figures 4.49, 4.50 and 4.51 show the absolute error between the real angle and the calculated angle. On the x-axis, the angles between 0 degrees and 175 degrees and on the y-axis the absolute error in degrees are plotted. The red line represents the median. Within the box - of the upper and lower quartiles - lie 50 % of the errors. The length of the box corresponds to the interquartile distance (IQR). The maximum length of the whisker is 1.5 times the IQR. The lower whisker is calculated as 2.5% quantile and the upper whisker as 97.5% quantile so that all data within the whiskers are considered mild outliers and all data outside are considered extreme outliers.

Figures 4.49, 4.50 and 4.51 are showing the absolute error in degrees for the calculation of the orientation using k-Mean Subspace Clustering. The median, as well as the quartiles and whiskers, are at most angles between 0 degrees and 5 degrees. This is an acceptable area. Striking however are angles between 0 degrees and 10 degrees and between 165 degrees and 175 degrees. Here the algorithm shows larger errors. The error range is between 5 degrees and 13 degrees. Figure 4.52 (a) shows an image with a real angle of 10 degrees. The error in this image is 16 degrees, but there is no obvious error in the algorithm. Figure 4.50 shows an image with a real angle of 165 degrees. The error in this image is 20 degrees. Here you can see that the edge of the container is not recognized correctly, and therefore, the angle calculation is also incorrect. Far more striking are the angles 55 degrees, 70 degrees, 75 degrees, 80 degrees, 95 degrees, and 100 degrees. Here, the box extends from 1 degree up to 89 degrees and therefore shows a high dispersion of the error. Figure 4.52 (c) shows an image with a real angle of 55 degrees. An angle of 141 degrees is calculated, so the error is 86 degrees. It can be seen that the container was not recognized correctly here and the aspect ratio is reversed so that the Bounding Box is turned by about 90 degrees. This is also the case at 75 degrees (d), 95 degrees (e) and 100 degrees (f). Sometimes the bounding box is not even detected within the original container, as shown in Figure 4.52 (g) to (i). Probably parts of the surrounding objects will be included in the calculation of the bounding box. Figure Figures 4.49, 4.50 and 4.51 shows, however, that the algorithm can deliver the correct result even with many interfering factors. Since the algorithm has to cope with several containers in the image, the method of k-Mean Subspace Clustering is not robust enough.

Figure 4.51 shows the absolute error in degrees for calculating the angle using template matching. Since the template is only rotated by 1 degree at a time and then compared, only natural numbers can come out as angles. Thus the error is also a natural number. For most angles, the error is constant at 1 degree. Rarely is the median at 0 degrees or 2 degrees. Outliers up to 2 degrees can occur. This is within the quality requirements for

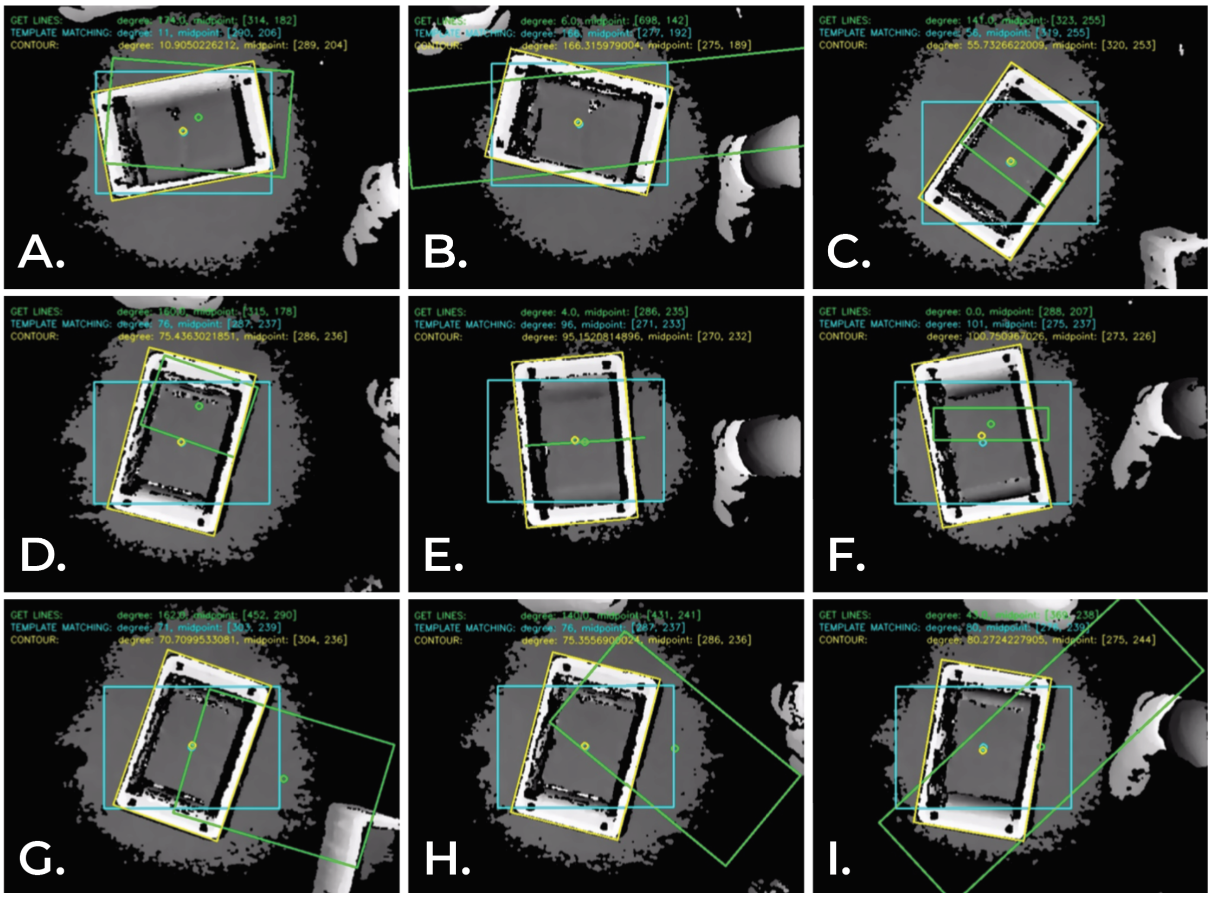


Figure 4.52: *Visualization of different errors in top view angle calculation (own illustration). A) 10 deg. container with an error of 16 deg., B) 165 deg. container with an error of 20 deg., C) 55 deg. container with an error of 86 deg., D) 75 deg. container with an error of 85 deg., E) 95 deg. container with an error of 89 deg., F) 100 deg. container with an error of 80 deg., G) 70 deg. container with an error of 88 deg., H) 75 deg. container with an error of 65 deg., I) 80 deg. container with an error of 37 deg.*

all angles.

The absolute error in degrees for calculating the angle using contours is shown in Figures 4.49, 4.50 and 4.51. The median of all different angles is between 0.2 degrees and 1.4 degrees with the upper quantiles also in the range below 2 degrees. Only the angle of 175 degrees is slightly different from the other angles since the median here is 1.8 degrees. After checking the evaluated images, no particular irregularity is discernible. The quality requirements are also fulfilled here so that this method of angle calculation can also be used for the robots.

A further criterion for the evaluation of the angle calculation is the speed, since for example with the depalletizing robot, the process of the container grasping and processing may amount to maximally 19 s. Most of the time is needed for the container so that the image processing should work as fast as possible.

Table 4.53 shows the median of the statistical values, mean and standard deviation of the three methods and pre-processing in terms of processing time. The standard deviation of the template matching is 0.19 s. Compared to 0.07 s and 0.04 s of the other two methods, template matching is much worse. If the mean values of the processing times of the various techniques are combined with those of the preprocessing, the following times result for the entire angle calculation.

	PREPROCESSING	K-MEAN SUB-SPACE CLUSTER-ING	TEMPLATE MATCH-ING	CONTOURS
MEDIAN [S]	0.22	0.13	5.20	0.08
MEDIUM VALUE [S]	0.25	0.14	5.30	0.08
STANDARD DEVIATION [S]	0.10	0.07	0.19	0.04

Figure 4.53: Time analysis - gripping point detection

It should be noted that the k-Mean Subspace Clustering method should not be used due to the inaccuracy and the Template Matching method due to the processing time. This leaves the contour method.

Side View

Since there are no variations within the algorithm when calculating the gripping point from the side, its performance is validated in the following chapter within the framework of the overall implementation of the Perception Algorithm. In contrast to the extracted experiments at this point, the performance of the robot can be tested there under real conditions in series production.

4.5 Evaluation

For the evaluation of the Perception Algorithm designed in this chapter, consisting of the modules Detection, Selection, and Localization, this algorithm is implemented using the exemplary use case of the depalletizing robot and checked about the fulfillment of requirements. For this purpose, it is first integrated into the existing robot prototype according to the robot structure described in Chapter 3. The robot is then tested in the series production process under real conditions in a vehicle factory. The following evaluations refer exclusively to the Perception Algorithm. Since the tested robot hardware was a prototype, isolated hardware-related errors occurred in the test series, such as a sudden interruption of CAN communication. Since new engines were continuously tested on the tested prototypes, for example, the components used were not perfectly matched to each other. However, this vulnerability will be fixed after the design freeze when the first series versions are procured. All conceptual peculiarities described in Chapter 3 fulfilled their purposes. The values listed below, therefore, refer exclusively to the errors caused by the software in connection with the perception module.

To make a precise statement regarding the accuracy and reliability of the Perception Algorithm, an attempt is made to simulate the real application scenario as well as possible for the validation. For this purpose, the complete robot was attached to a single bearing stitch, as described in Chapter 2. It should be mentioned that such a prototype cannot be used in the real process as long as negative effects due to process errors for vehicle assembly cannot be ruled out. To enable a large variety of containers, these were manually fed to the robot via the existing conveyor technology. The tests were carried out throughout one week. Each gripping process was documented. The results of the individual modules were also analyzed and archived. The lighting situations were identical to real applications. The only difference to the serial process was the manual container selection. In this way, scenarios could be simulated more frequently in which the containers had to be gripped despite heavy soiling or damage. The results presented in the following subchapter are, therefore, the representative for the use of the robot in a real environment. Without the manual complication of the test by manipulated containers, a significant improvement of the results in the series process can also be assumed.

During the experiments at the plant, 241 containers were depalletized and documented. Since the test was performed on prototype hardware of the depalletizing robot, the perception errors had to be separated from the general (hardware or control) errors during the error analysis. Of the 241 containers, a total of 140 could be handled without manual intervention. With 70 containers, it was necessary to intervene after perception-related software errors. The next step was to analyze in which module of the Perception Algorithm the errors occurred. Those results are shown in Table 6.2.

Table 4.9: Overview of error rates in the respective modules of the Perception Algorithm

Module	Error Rate
Detection-Module	8,4%
Selection-Module	0%
Localization-Module	19,6%

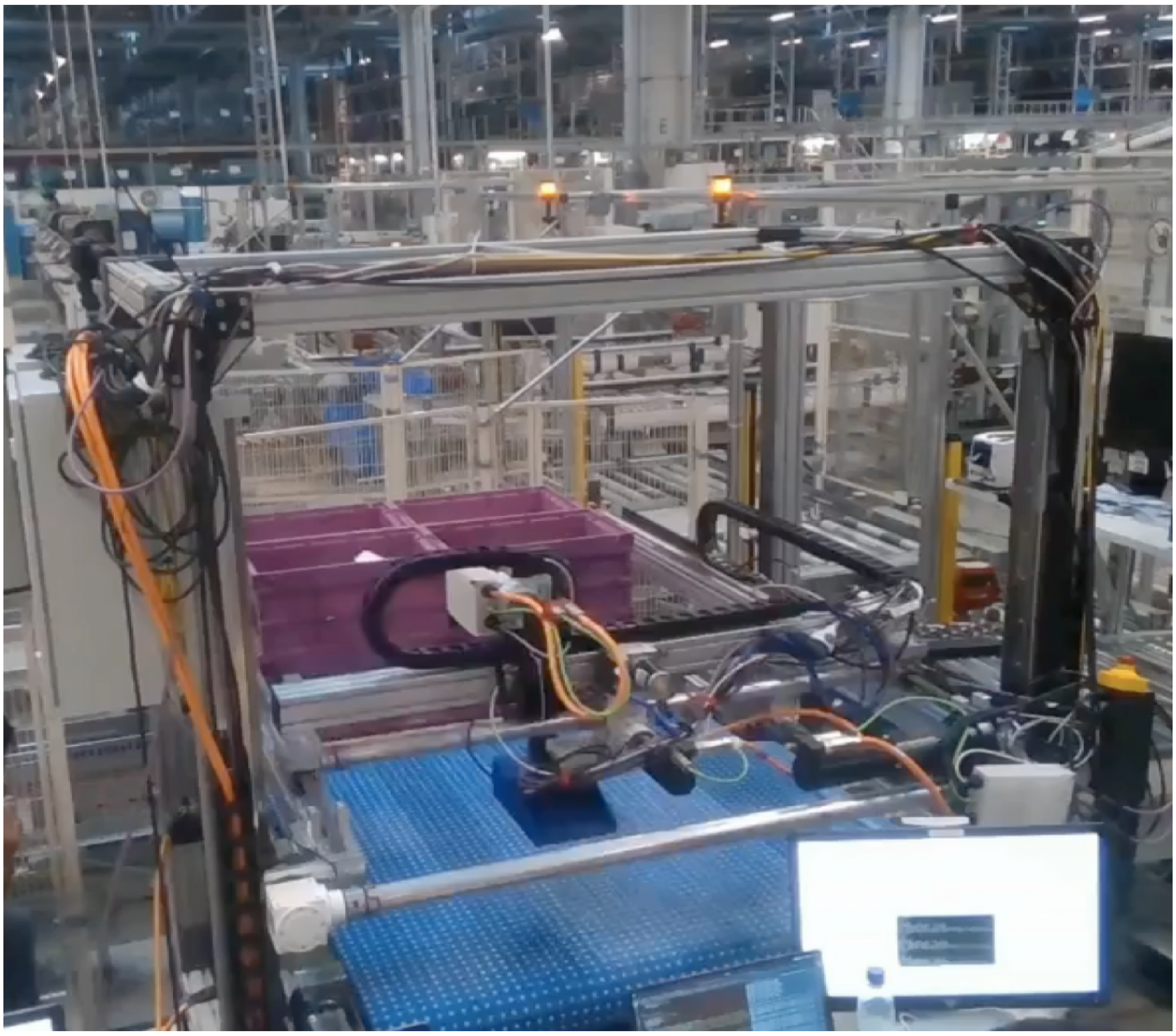


Figure 4.54: Image of robot and Perception Algorithm evaluations in the real plant environment (Leipizig) (own illustration)

4.6 Conclusion

In this chapter, the functional modules of the Perception Algorithm were systematically developed, validated, and the underlying scientific relationships explained.

Object detection takes place in the first module. All objects relevant for the application are recognized using neural networks based on two-dimensional color images. In the context of the robots to be developed in this thesis, Yolo is used based on in-depth comparative tests and trained with logistically individual data sets.

In the second module, the relevant objects for the respective execution of the process are selected based on the detected objects. For example, criteria such as available workspaces or sequences to be adhered to are taken into account during object processing.

Furthermore, the gripping points of the selected objects are calculated in the third module. For this purpose, a depth image is taken, which is then processed in various processing and analysis steps. The prepared image can then be used to search for physically suitable gripping surfaces for positioning the robot gripper.

Finally, in this chapter, the described software modules were implemented in the robots and tested under real conditions in the following area of application. The results obtained are summarized in Figure 4.55. The Perception Algorithm will be extended in the following chapter to eliminate the weak points visible in it and to enable the robots to be used in series production in the industry.

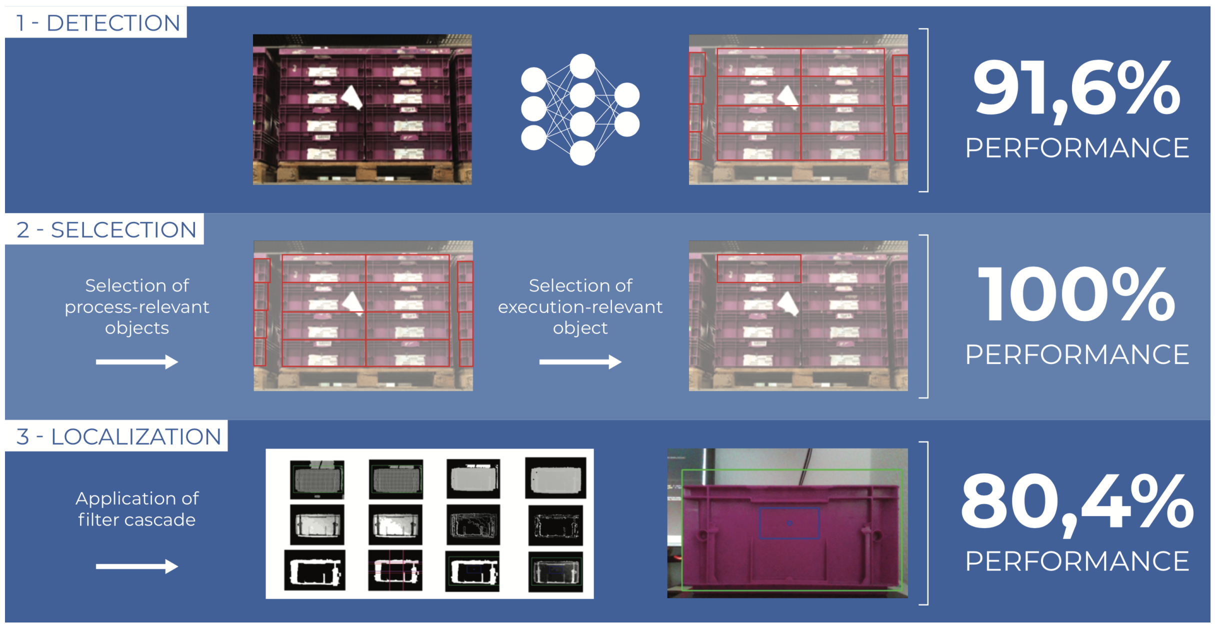

1 - DETECTION
91,6%
PERFORMANCE
2 - SELCECTION
Selection of process-relevant objects
Selection of execution-relevant object
100%
PERFORMANCE
3 - LOCALIZATION
Application of filter cascade
80,4%
PERFORMANCE

Figure 4.55: *Summary of the evaluation of the Perception Algorithms functional modules (own illustration)*

5 Robust Perception Algorithm

As the experiments in the previous chapter in the series process under real conditions have shown, the Perception Algorithm implemented there does not yet fully meet the requirements.

To improve the performance of the entire module, this chapter first analyzes the errors that occur and divides them into error clusters to be solved. The generated error patterns are then compared with holistic solution strategies. These are then implemented, parameterized, and validated again.

5.1 Error Analysis

This chapter analyzes the occurring errors. This analysis is divided into two parts: error pattern caused by the Detection-Module and error pattern created by the Localization-Module.

5.1.1 Detection

To introduce this subchapter, the current state of the art will first be analyzed at the beginning. Subsequently, the mobile application specially designed for the performance analysis of neural networks for object recognition, will be presented. The insights gained through the use of the application into the weak spots of the networks are finally documented.

Related Work

The most common method for evaluating the performance of deep neural networks for object detection is the calculation of statistical indicators, such as mean average precision mAP [Dai16] [Gir+12], which compares the predictions of the networks with the ground truth normally marked manually. Although this procedure provides an objective and comparative assessment of performance, it does not provide information on the exact nature of the errors. Also, this approach is very time-consuming when applied to all logistics applications in an automotive assembly plant or, for large manufacturers, even to all of their assembly plants. Furthermore, the key figures obtained in this way only reflect the performance of a network at a specific point in time. Accordingly, the continuous changes already described in the logistics processes, e.g., new containers, are not taken into account.

The continuous use and real-time performance monitoring of the robot would be the best approach for such investigations. The impracticability of this technology is evident due to the high financial and logistical costs and possible disruptions in the flow of plant material.

Mobile Application

To counter this problem and to execute longterm validation tests of the created Perception Algorithm in the real plant environments, a mobile application was developed for error analysis within the scope of this doctoral thesis. These can be used to call up and execute the Perception Algorithm from the specific robots that have been adapted to the respective applications. In addition to the essential functions, this also includes the execution of the trained networks for object recognition and thus enables validation of the Perception Algorithms in the real logistics environment under the relevant industrial conditions without the need for complex integration of the robots into the respective processes.

The application is structured as follows. At the beginning of the test run, an existing parameterization must be selected from the list. For this purpose, there is the possibility of either a logical selection based on the application case characterization as well as the local distribution of the robots over the plants. If you are also close to a relevant robot application, you can also select the Perception Algorithm by reading out the Quick Response (QR) codes on the robots. If the application to be examined does not yet exist in the data backbone on which the app is based, it must be created. Application area, network, and application-specific parameters can be selected.

After selecting the Perception Algorithm to be analyzed, the user reaches the execution screen. The Fotolivestream, which is visible through the camera, is shown there in the large window. The detected objects are displayed in these boxes in the form of bounding boxes as it is shown in Figure 5.1. At the top of this box, the corresponding class and the confidence of the prediction are also displayed. As already mentioned several times, the two decisive evaluation variables of these authentication modules are performance (reliability) and the computing time required for robust detection. Additional information regarding the calculation speed in frames per second can be read at the bottom of the screen. For the subjective evaluation of the performance, there is the possibility of assessment (1 - 5). Besides, comments can be left behind. To document the weak points and defect images identified, it is also possible to take screenshots, including the results calculated by the neural network for object recognition.

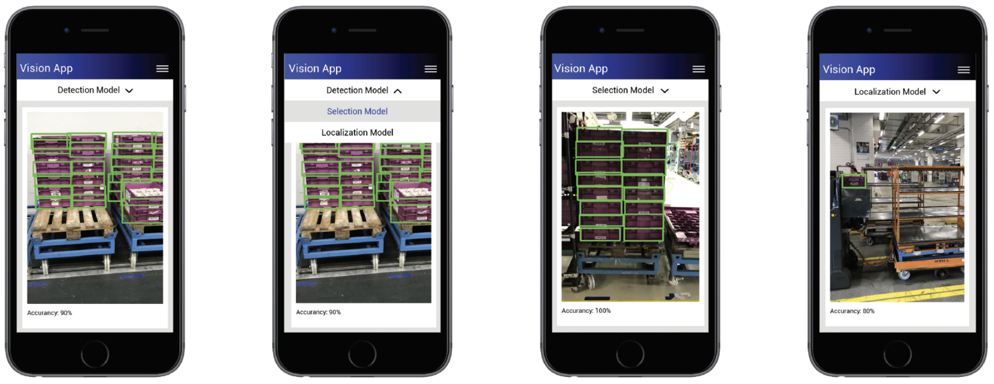


Figure 5.1: App functions (own illustration)

In this execution screen, three different modes can be selected according to the conception of the entire Perception Algorithm to be able to make statements regarding the performance of the respective individual modules. The following module combinations can be selected for this purpose:

- Detection-Module
- Detection-Module + Selection-Module
- Detection-Module + Selection-Module + Localization-Module

Separate tests of the individual Selection- and Localization-Modules are not possible since the results of the previous modules are required as input variables for these.

Before going into the error patterns to be observed, it should be mentioned at this point that a performance comparison of application and robot has shown that in the first two modules for Detection and Selection no difference between the algorithm on the robot and the algorithm of the mobile application is discernible. This is reasonable because on the one hand side of the similar weights of the converted DNN and the other hand the identical code functionalities only this time in Swift and not in Python.

However, the third module, which is necessary to determine the gripping pose, is subject to deviations. This is because, in the application, the depth data are concluded via the two camera lenses of the iPhone as a kind of stereo camera. Besides, the raw data recorded by the iPhone is subjected to numerous processing steps in iOS [9to19]. This does not happen in this form in the robot. Therefore, the mobile application is initially only used to validate the first two modules.

Results

For validation and analysis of the Detection-Module, the mobile application was used in the relevant logistics areas of the plant Leipzig. In this test series, two overarching phenomena could be observed:

- Error in object detection itself
- Temporal changes of detected objects across multiple frames

After an in-depth analysis, the documented errors in object recognition could be assigned to the following three categories:

- Error pattern 1: Objects are not recognized
- Error pattern 2: Objects are recognized with wrong dimensions
- Error pattern 3: Objects are recognized as wrong class

These error patterns are shown schematically in Figure 5.2 and supplemented with examples of real plant validation. In this image, different colors are used. Blue bounding boxes are representing the manually labeled ground truth. Correctly detected objects are

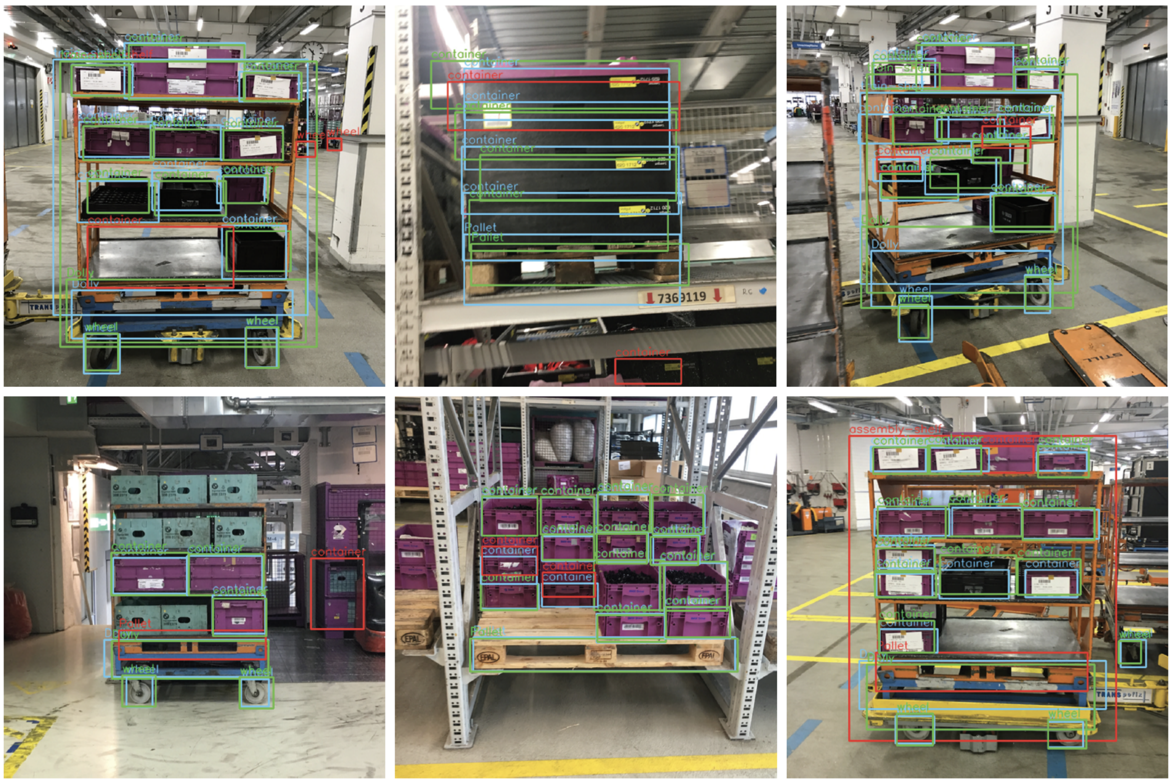


Figure 5.2: error clusters: blue - ground truth, green - correct detections, errors - red (own illustration)

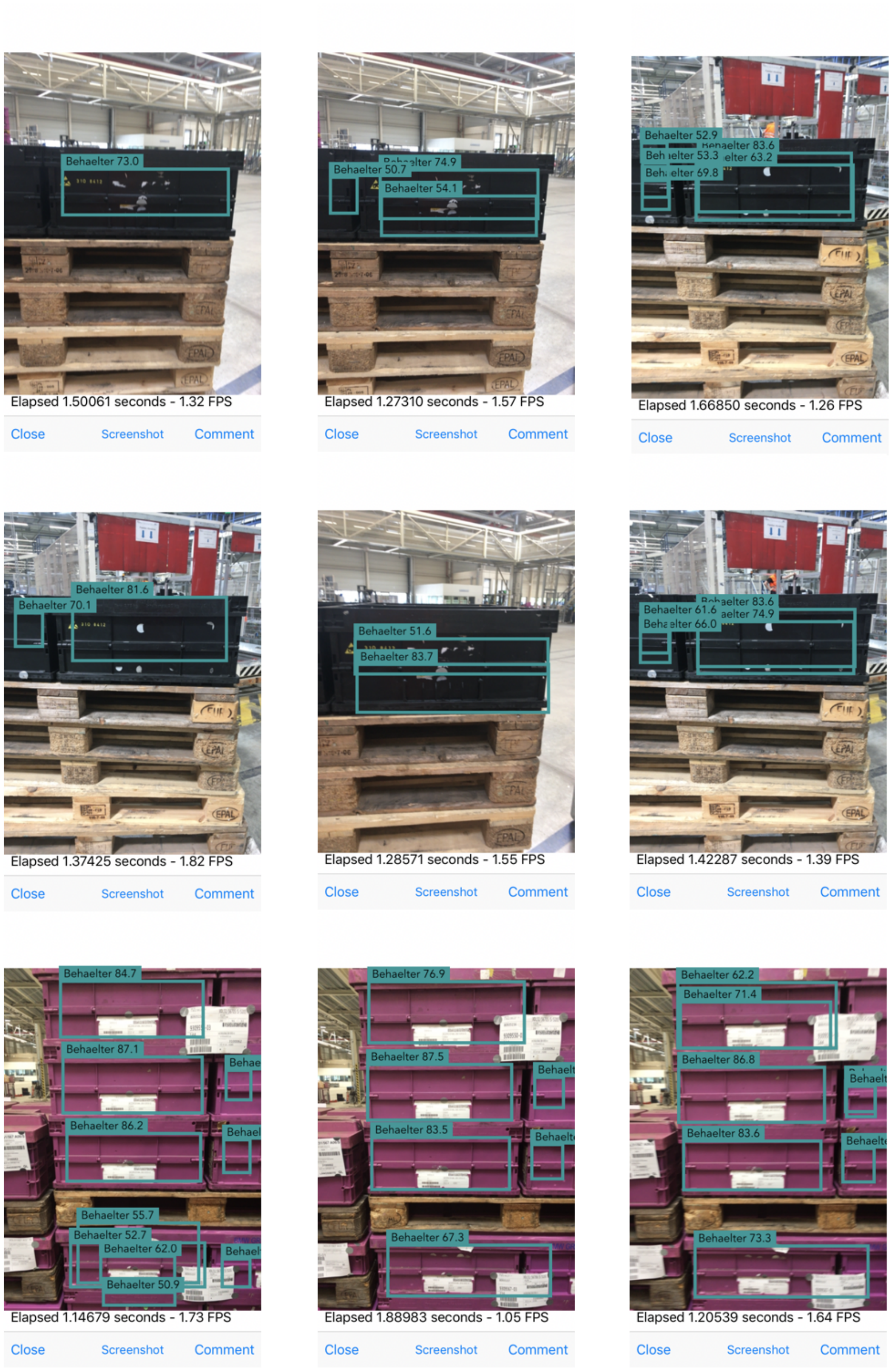
marked with green boxes whereas errors accordingly to the error as mentioned above clusters are visualized in red.

Figure 5.3 shows examples for the occurring temporal variances. For these images, the Detection-Modules results were directly taken for a duration containing several frames.

It should also be noted here that these error patterns usually occur in combinations. Depending on their characteristics, this can have different effects on the process execution of the robot.

If the robot's targets are not detected, as shown in Figure 5.2, this can lead to an interruption in process execution. If multiple objects are visible to the robot, this would mean that the robot would go to the selection step with the wrong set of objects, and would, therefore, be more likely to choose to grab the wrong object. This can not only lead to an interruption of further process execution but can also damage components.

Correctly detected objects with incorrect geometric dimensions influence, in particular,



```
Elapsed 1.50061 seconds - 1.32 FPS
Elapsed 1.27310 seconds - 1.57 FPS
Elapsed 1.66850 seconds - 1.26 FPS
Elapsed 1.37425 seconds - 1.82 FPS
Elapsed 1.28571 seconds - 1.55 FPS
Elapsed 1.42287 seconds - 1.39 FPS
Elapsed 1.14679 seconds - 1.73 FPS
Elapsed 1.88983 seconds - 1.05 FPS
Elapsed 1.20539 seconds - 1.64 FPS
```

Figure 5.3: Errorclusters (own illustration)

the final determination of the gripping point. If a solid gripping surface is found for the wrong bounding box, there is a high probability that the handle will distort the container and thus damage other parts.

There are two possible variants for the third error pattern. On the one hand, incorrect detection of a container, e.g., as a lattice box, would have comparable effects to undetected objects in error pattern 1. On the other hand, if other objects are incorrectly detected as containers, this can lead to severe damage to components and robot parts or even to injuries.

This list of effects on the already formulated requirements, the meaning, the Perception Algorithm is robust and error-prone to get.

The error images described in this paragraph also have the property that can occur from Frame to Frame in various forms. Errors in the Perception Algorithm for a frame do not necessarily happen with the following frames.

5.1.2 Localization

As already mentioned in the previous section, the Localization-Module can also be represented via the app, but primarily due to the different camera hardware and integrated image processing technology, the findings observed in the mobile application cannot be transferred entirely to the gripping pose determination in the respective robots. For this reason, a further test series is carried out for the error analysis of the gripping pose determination.

Experimental Setting

To be able to reproduce the real framework conditions in the test series best, those experiments were carried out at the plant Munich in various areas of incoming and outgoing goods. Thus the environment in which the containers are located is always different. Care was also taken to ensure that scenarios were recorded in which, for example, separate containers are stacked or in which different containers are located in the background. When selecting the containers, care was taken to ensure that the diversity of the material flow was representative in terms of geometric dimensions, materials, and colors. The gripping algorithm was evaluated in 19 different container types. To reflect the unplanned variance that exists in addition to the planned deviation as described in Chapter 2, several different containers were used for each of the 19 container types. These were taken directly from the real material flow of the plant. A list of the tested containers can be found in the appendix.

Rating Scheme

To evaluate the accuracy of the results of the Localization-Module, the detected gripping point is drawn into the RGB and depth images. Figure 5.4 shows examples of possible result regions of the gripping point algorithm. As already discussed in the previous chapter, a valid gripping point can be located on all smooth surfaces that are large enough to place the suction pad and build up a vacuum. If besides, it is taken into account that the container is subsequently drawn in, for example during belt supply or depalletizing, the gripping point must be as central as possible ("optimal") on the container, to avoid any skewing. Touchpoints that do not meet these criteria are considered as "not valid". If the specific gripping point is also outside the container itself, this is evaluated as "false". The respective defined gripping zones for all examined container types can be viewed in the appendix, and some of them are shown in Figure 5.4.

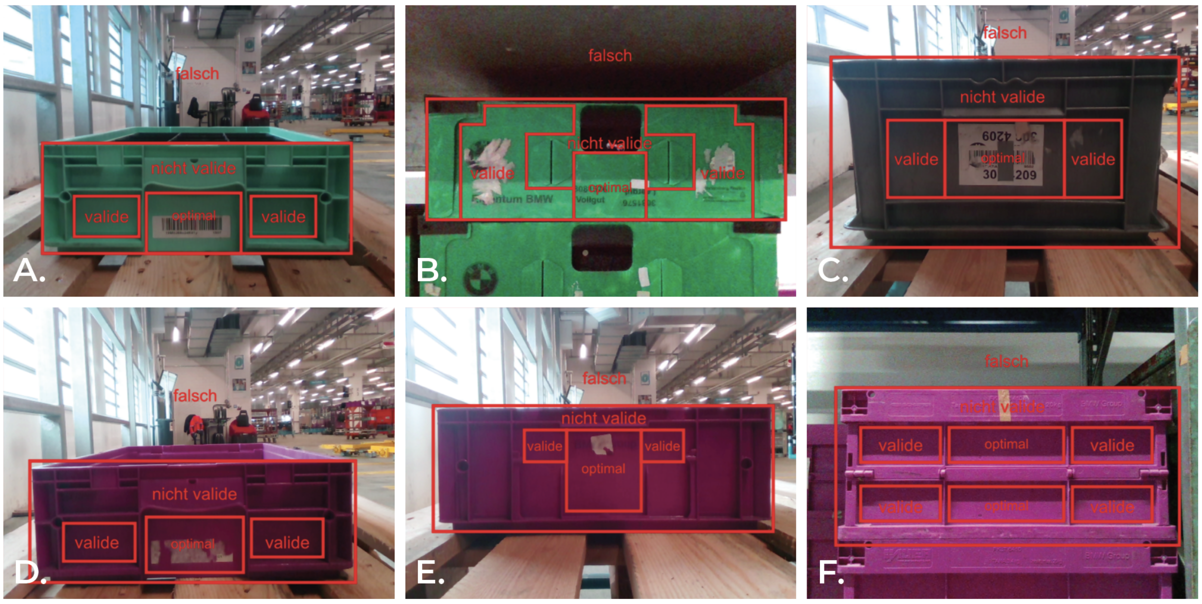


Figure 5.4: Grippping area classifications (own illustration). A) Container-Type 2, B) Container-Type 3, C) Container-Type 8, D) Container-Type 10, E) Container-Type 11, F) Container-Type 18.

Throughout the entire test series, 7300 images were recorded, processed, and evaluated.

Following the execution of the algorithm, all gripping points of the categories "optimal" and "valide" and further analyze the remaining test samples to identify general error causes.

Results

Of a total of 7300 test images, 59% of the gripping points were optimally determined, and a further 3% were valid. The remaining 38% are divided into 34% non-valid and 4% false. Compared to the implementation in the real robot, this can be explained in this case by the larger container variance used. In the real tests, the containers were pre-sorted according to their possible tangibility depending on the presence of labels and statistically according to the distribution of the container volume in the real-logistics material flow. This explanation is supported by the breakdown of the error rates of the analyses carried out in these test series into the individual container types. This can be seen in Figure 5.5.

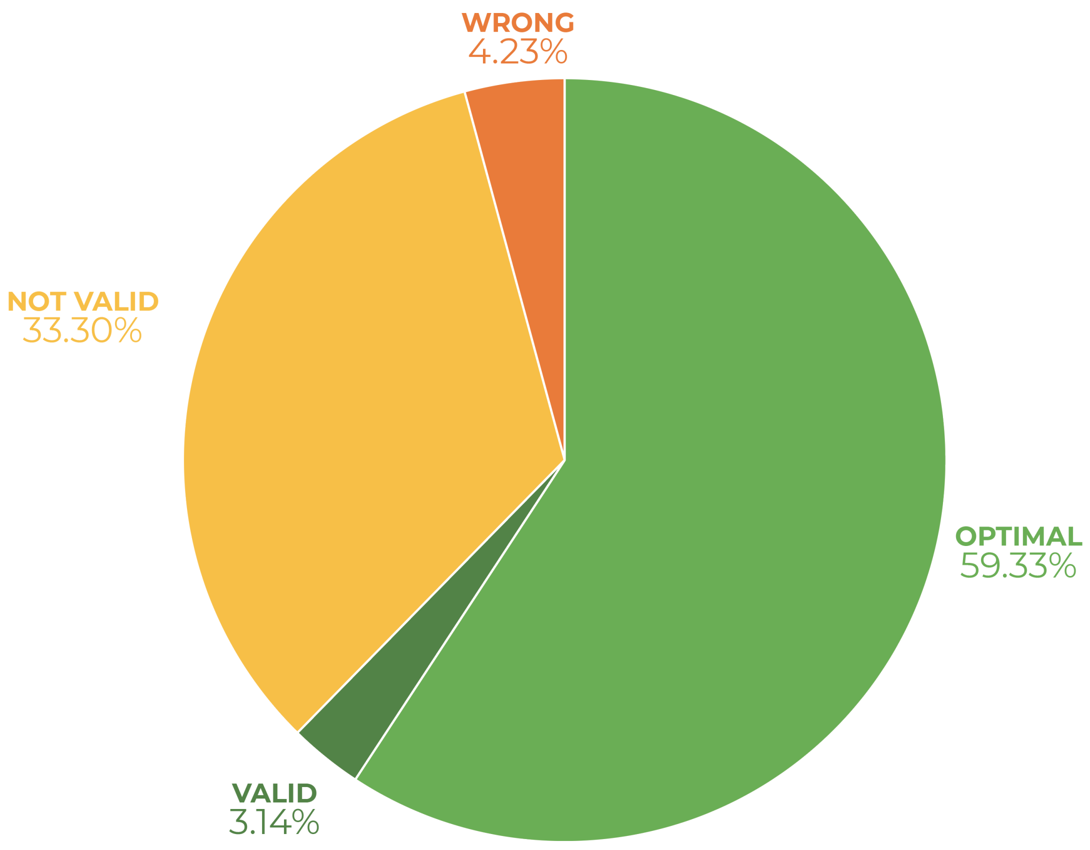


Figure 5.5: Gripping results (own illustration)

To focus the analysis of this chapter exclusively on the weaknesses of the gripping pose determination, the influences of the previous perception modules must first be removed. For this purpose, the processed RGB images are viewed, and those test series for which the error can be assigned to Detection (Module 1) or Selection (Module 2) are sorted out. Sample images of these error sources are shown in Figure 5.6 for better comprehensibility.

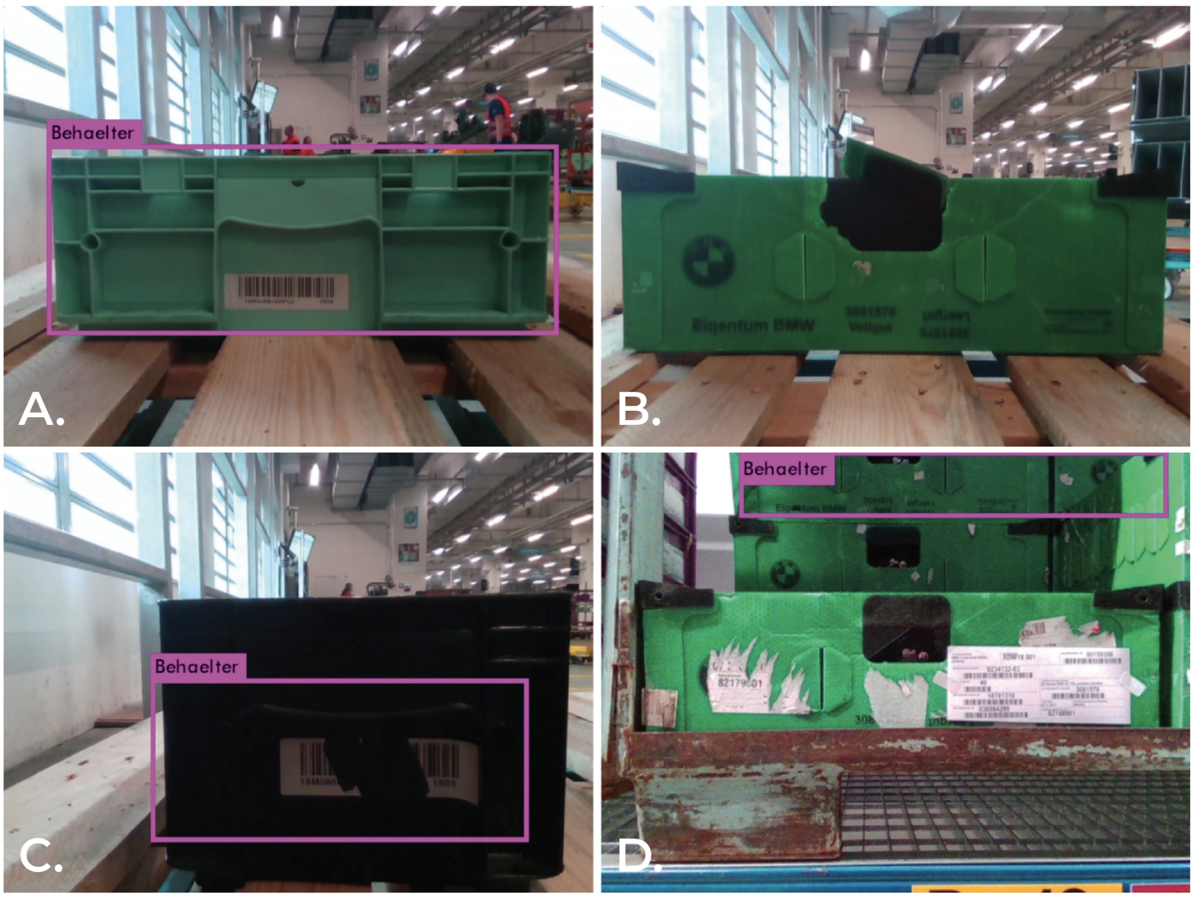


Figure 5.6: Neural Network errors that influence the gripping performance (own illustration)

By excluding sources of error from previous process steps, the quantity of incorrectly defined gripping points can be reduced by 22.85%.

The depth images are then analyzed for the remaining faulty gripping poses. This shows another dominant error pattern in the form of broken depth images. These errors can take on a variety of styles. Three of these are shown as examples in Figure 5.7. The first error pattern shows up in the form of inconsistent depth images. Several adjacent pixel columns of an image are faulty. It can also happen that the depth image is composed incorrectly. Finally, partially completely erroneous images are observed. A closer analysis of the camera hardware used has shown that these phenomena are independent of the scene to be recorded and the camera used. Instead, these error patterns are characterized by a specific random factor. With a total of 40% of the original 38% of the gripping point faulty, the depth image file can thus be determined as the cause of the error.

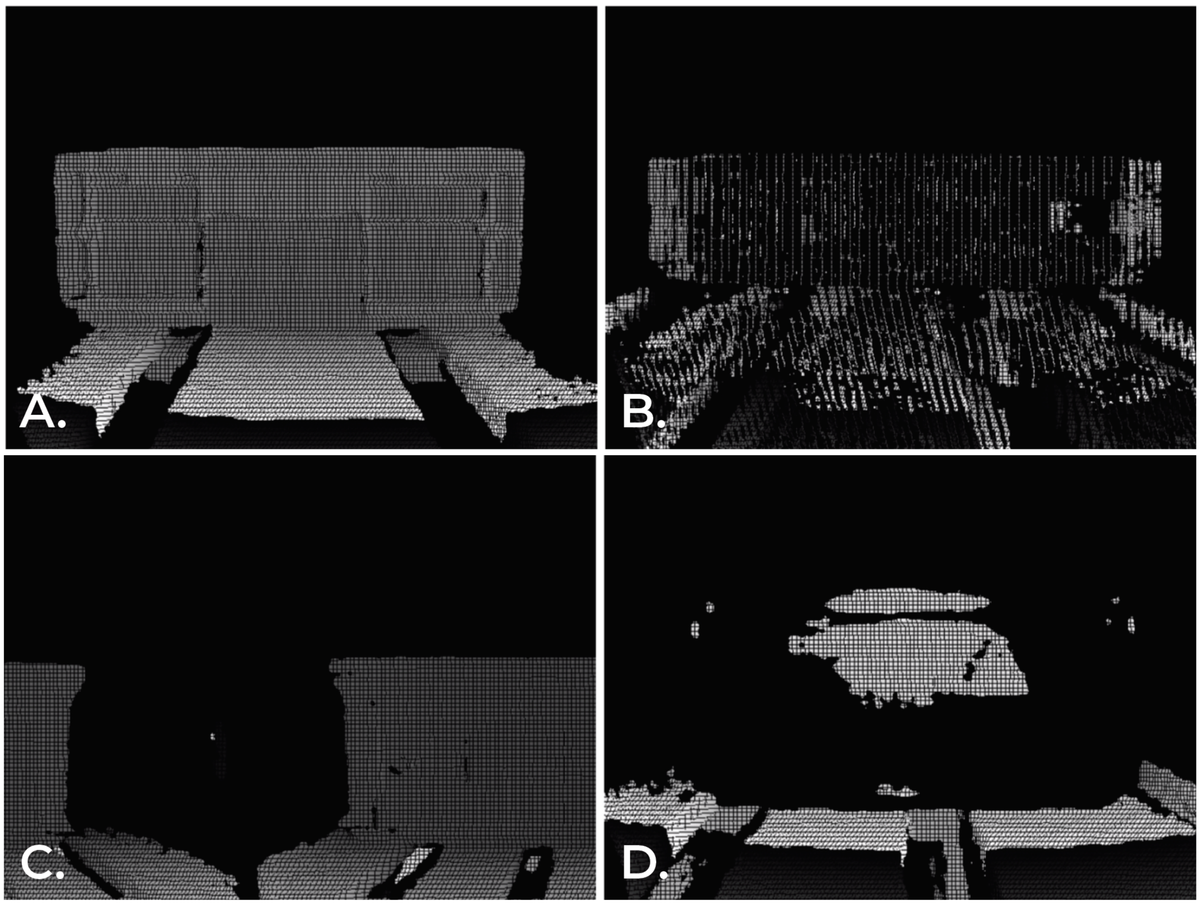


Figure 5.7: Depth errors (own illustration). A) Correct depth map, B) Not consistent depth map, C) Wrong depth map, D) Container only partly visible in-depth map.

The remaining errors can now be analyzed concerning the algorithm itself. The error patterns that can be seen can be assigned to two fundamental causes. These are shown in Figure 5.8 and are briefly explained below. The first cause may be the crooked shooting of the images, for example, if the container is slightly rotated in front of the camera. This will contaminate the depth data, and the algorithm will fail. The second cause can be attributed to faulty containers. For example, holes or several layers of labeling are potential sources of error.

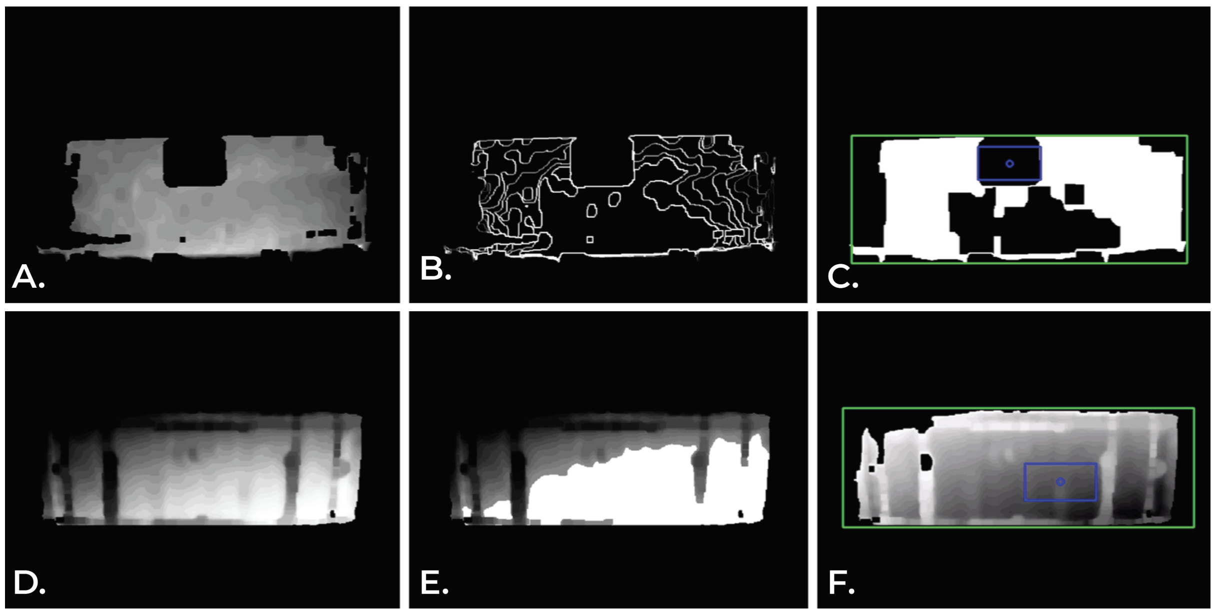


Figure 5.8: Remaining errors (own illustration). A) Depth image with a hole, B) Gradient of the container from A), C) Grippingpoint gets found in the black area of A), D) Container is captured tilted, E) Thresholding removes important information from the depth image, F) wrong area for the gripping point gets selected.

5.1.3 Clusters

In summary, the errors of the entire Perception Algorithm, i.e., all three implemented modules, can be split into four categories:

- Faulty object detection
- Temporal different object recognition results
- Time-dependent depth image quality
- Wrong basic conditions (damaged containers, wrong camera angle).

For these four error categories, solution strategies are examined in the following subchapter to increase the overall robustness of the Perception Algorithm to be able to satisfy the industrial requirements fully.

5.2 Concept

As already mentioned errors in the Perception Algorithm can have severe effects on the production system, but above all, also on human health. Therefore, it is imperative to increase the system's inherent robustness as much as possible on the one hand and to give the overall algorithm a certain degree of awareness on the other hand to at least recognize uncertainties or calculated errors and thus be able to initiate special measures without causing damage.

For this purpose, the previous Perception Algorithm, consisting of the three modules "Detection", "Selection" and "Localization", is supplemented by the two further modules "Validation" and "Aggregation".

The first is used to validate the detected objects to exclude the influence of faulty or incorrectly identified purposes on the following two modules. The results are evaluated and averaged over several frames to determine the impact of time on both the recognition of objects and the determination of gripping poses. A schematic sequence of the Perception Algorithm, now consisting of five modules, is shown in Figure 5.9. The exact structure and contents of the two robustness enhancing modules are shown in the following sections.

An algorithm-based solution of the fourth error class is not possible. Instead, the necessary framework conditions must be ensured throughout the entire process. As these efforts are organizational, they will not be considered further here.

5.3 Validation Module

To exclude the error patterns described in the previous section when using the robot in series processes, strategies are developed. The aim of the proposed approaches is either to prevent the occurrence of errors or, if this is not possible, to detect them reliably to avoid any damage. For this purpose, state of the art will first be examined in more detail bevor selecting and implementing the final module.

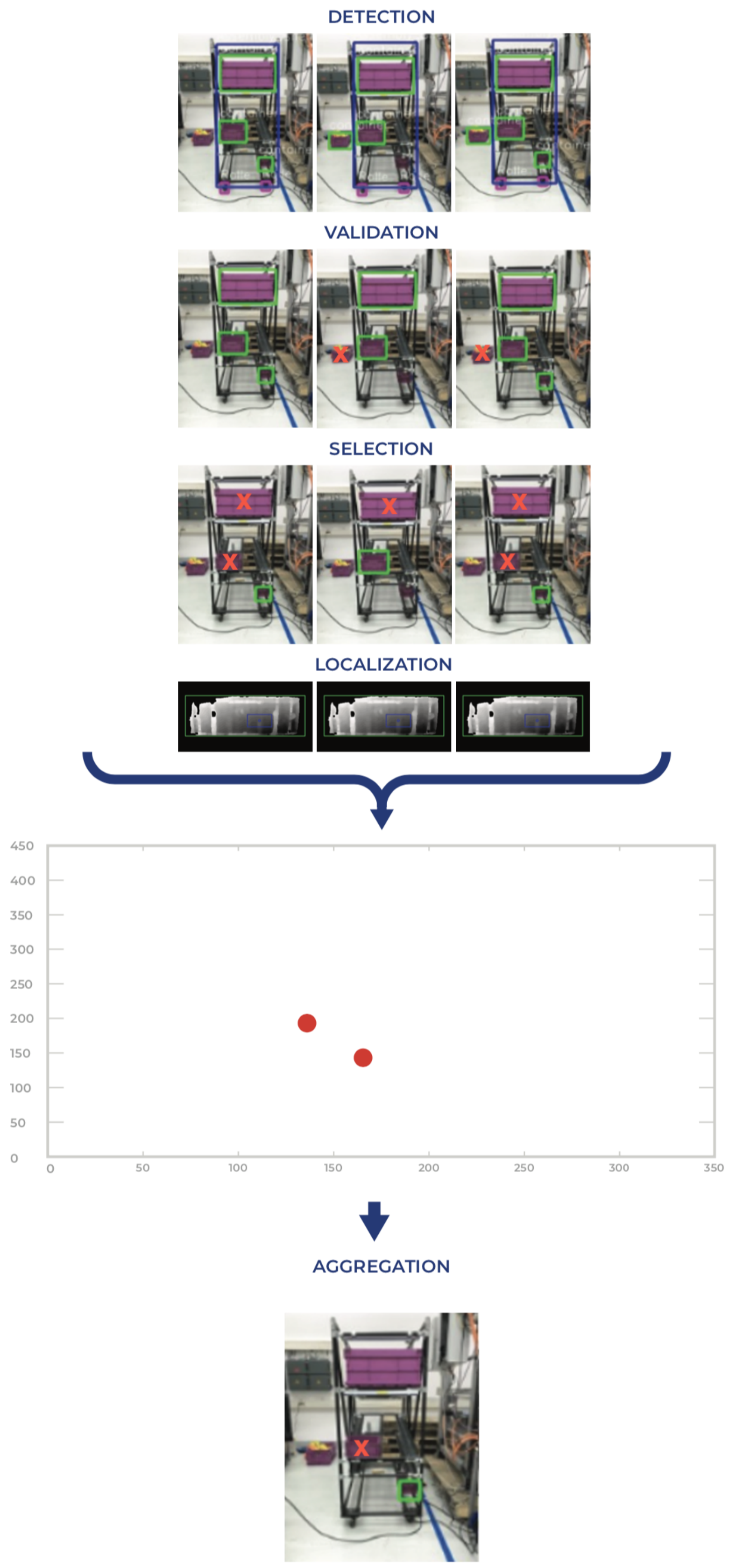


Figure 5.9: Final Perception Algorithm (own illustration)

5.3.1 Target

An important difference between a person and a computer with vision capabilities in detecting objects is the ability to decide if the observed object is task-relevant directly. On this ground, a Task False Positives (TFP) rate is defined that combines the network's false positives (FP) and the task-irrelevant predictions.

$$TFP = FP + Task_{irrelevant} \tag{5.1}$$

Thus, the goal of the validation module is to reduce the TFP rate. Specific information regarding the detection task such as objects dimensions and condition occurrences can be used to validate the detection process.

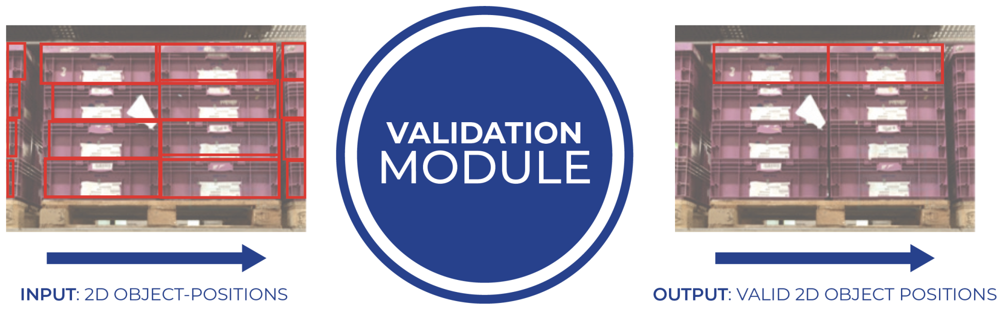


Figure 5.10: Target-specication for the Validation-Module (own illustration)

Target-specication for the Detection-Module (own illustration The aim of the Validation-Module is, therefore, to pass on only valid positions, i.e., realistic positions of objects which also occur in the real scene, to the Selection-Module.

5.3.2 Related Work

Object detection models can achieve high performance in detecting multiple objects in RGB images. In a completely different environment, however, they often produce false predictions. Especially in industrial applications, the reduction of the value of incorrect predictions is decisive for the safe functionality of the robot. The fact that state-of-the-art detectors ignore contextual and scene-related information increase the number of incorrectly captured objects. In related studies, two different approaches to context and object information can be identified. In this area, previous knowledge about the probabilities of object events was used, and correct predictions based on these hypotheses were validated. Gierad Laput [Lap12] introduced the zone assignment algorithm to reduce the occurrence of false alarms in a vehicle detection system. The frequency distribution is mapped from the fundamental truth. Based on this map, the regions with high object occurrences are known and used to validate a detection if it falls into this region. The second approach focuses on the modification of the architecture of the applied neural network. Further layers and networks aim to learn more about scene analysis and to arrive at a robust detector. Therefore [GL13] introduced a novel technique called Context Forest to fathom the connections between global manifestations in the image. Some other works [YPC14] [Vez15], implemented object detectors with modified architecture to learn more about the observed scene and the relationship between different objects. The neural network itself is trained with additional information. Another technique uses object properties to reject false candidates, with other classifiers trained to recognize specific features of the object. In this context, Arandjelovic et al. [Ara+05] add a skin color classifier based on Gaussian models to reject false faces. To apply such techniques, a modified neural network must be implemented and trained with large amounts of data. To the best of our knowledge, validation as a second step, taking into account specific and global context information, has not yet been discussed in the literature. To avoid building a new object detector with a modified architecture adapted to our vision system, a different and flexible approach is proposed.

5.3.3 Concept

The Validation-Module is an error handling procedure that requires the definition of a set of contextual criteria to distinguish not only between true and unrealistic recognitions. This method aims to validate only predictions that comply with predefined rules, taking into account contextual indications in a given environment.

The criteria for the structure of the Validation-Module in autonomous container handling are divided into three groups and described as follows:

- Object-specific criteria
- Class-specific criteria
- Scene-specific criteria.

These three groups are explained in more detail below.

5.3.4 Object-specific Criteria

Object-specific conditions contain information about object-specific properties of real estate. The predicted objects are only validated if they meet all conditions. For example, the dimensions of the captured objects can be checked here. Therefore, all correct dimensions are stored in a database. The predicted dimensions (height-width ratio) are then compared with the stored real dimensions.

5.3.5 Class-specific Criteria

Class-specific criteria are the validation conditions that describe the occurrence of objects in a particular class. One of the conditions is, for example, that overlap between objects of the same class is not allowed. Further criteria describing the placed objects in the working environment of the robot are implemented. For instance, a recognized object within the class repository cannot be under or within another object of the same class, independently of other classes. In other applications, such as depalletizing, all containers on a full pallet must meet the requirements of entire pallet layers. This means that the containers must maintain the permissible vertical distance and not the horizontal distance rule.

5.3.6 Scene-specific Criteria

The scene-specific conditions represent a validation of the relationship between the classes. Real objects appear together with other objects and usually have certain relative positions. For example, a predicted bounding box of a shelf may not be under a predicted container or pallet. In addition to the logical scene conditions, task-related conditions are also applied. Many examples can be described in this context. When a container is unloaded from a shelf in the assembly line, only containers that are actually on the shelf are validated. Only containers on a pallet are validated for depalletizing. For example, there is a maximum of eight containers in one loading unit. For the validation of tugger train frames, the detection of a dolly with all-wheel drive is required. Besides, some characteristics must take into account like perspective distortions. Since only two wheels are usually visible from a four-wheel platform, two detected wheels are a sufficient validation state.

5.3.7 Test

By implementing a specific Validation-Module for each robot, the module would gain accuracy, but the vision system would lose its general aspect. For this reason, a decision algorithm is developed based on a tree of configurations. A generic implementation with basic functions such as size comparison (between real dimensions and predicted dimensions) and relative position (between two objects) will be developed. These basic functions have adjustable parameters and threshold values that can be adapted to the respective robot task. Each robot only has to parameterize relevant functions taking into account its specific context information. To evaluate the effectiveness of the proposed Validation-Module, the algorithm is applied to different images of randomly placed objects in a complex environment where the object detector was performed with low accuracy. Tolerances and threshold values are adapted to the changed perspective. For example, the location and size perspective is taken into account. False predictions of the detector and detections that are not relevant for the process before and after validation are calculated. The result is shown Figure 5.11.

The result shows how the module can significantly reduce the number of incorrectly captured objects by various combinations of validation criteria. The number of incorrectly captured containers in the test images is reduced from 101 to 46. Also, the Validation Module rejected about half of the incorrectly captured pallets and reduced the number of incorrectly captured shelves from 5 to 3.

By reducing the false and irrelevant detection rates, the vision system gains in precision and robustness. Therefore, the Validation Module proves to be a promising approach to improve the vision system of the robot to meet the requirements of the application.

Whether the degree to which the Validation Module can reduce the error detection rate is sufficient for a robust process flow must be evaluated on the robot itself in the series process. The relatively high error rate in this test results in particular from comparatively difficult images and, as mentioned above, a network with low accuracy.

5.4 Aggregation Module

As already analyzed, both in the Detection-Module and in the Localization-Module frame-selective algorithm results due to the highly variable environment and partially

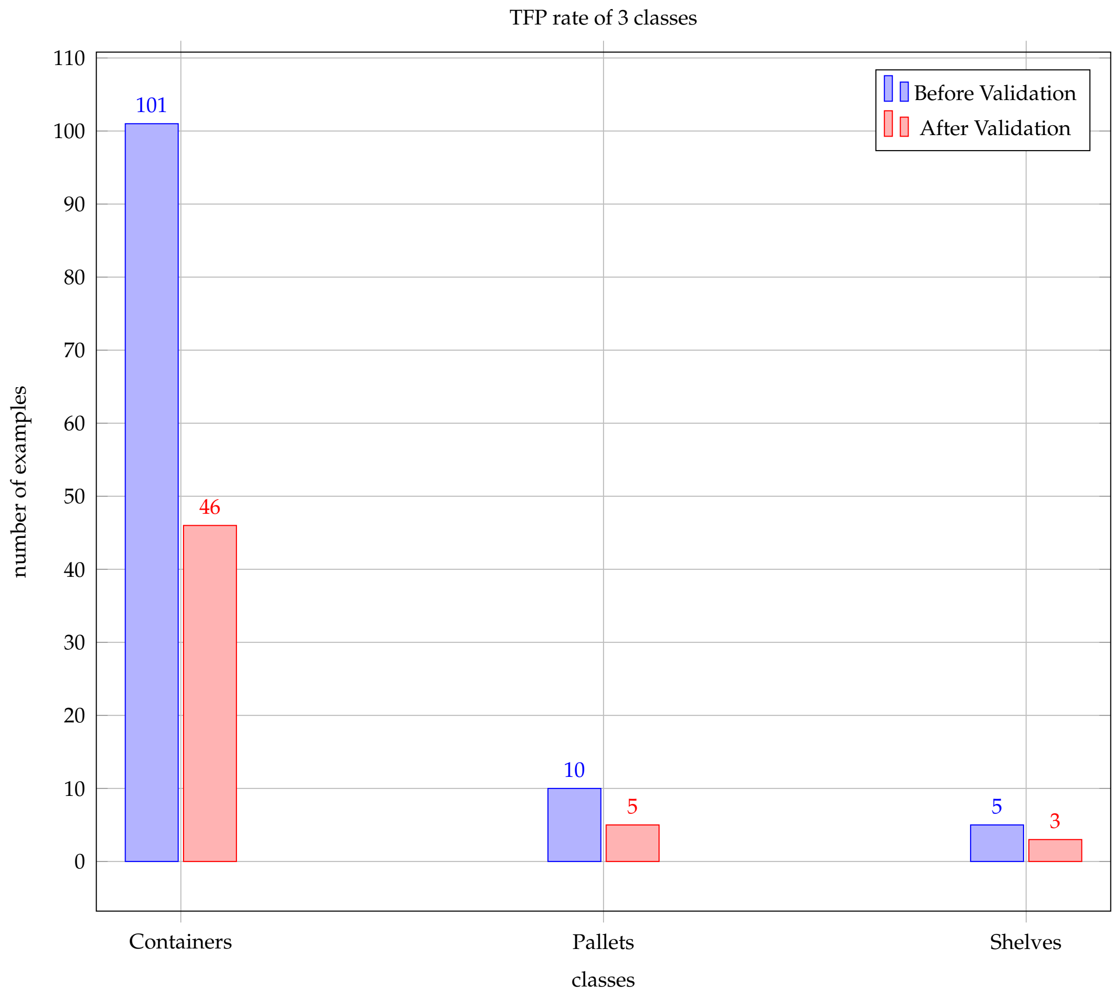


Figure 5.11: TFP rate before and after validation on a mixed test dataset (own illustration)

random sensor noise. Dealing with those variances play an essential role in getting the Perception Algorithm robust and reliable.

5.4.1 Target

The Aggregation-Module, therefore, defines two main objectives. On the one hand, the occurrence of damaged images or depth images is to be avoided by using several frames. On the other hand, by processing several frames per gripping point determination process, these temporary differences are to be used for the detection of anomalies and the determination of an average gripping pose.

To illustrate this objective, the pixel values of the real target and the predicted target values are shown in Figure 5.13. These values were derived from an analysis of ten

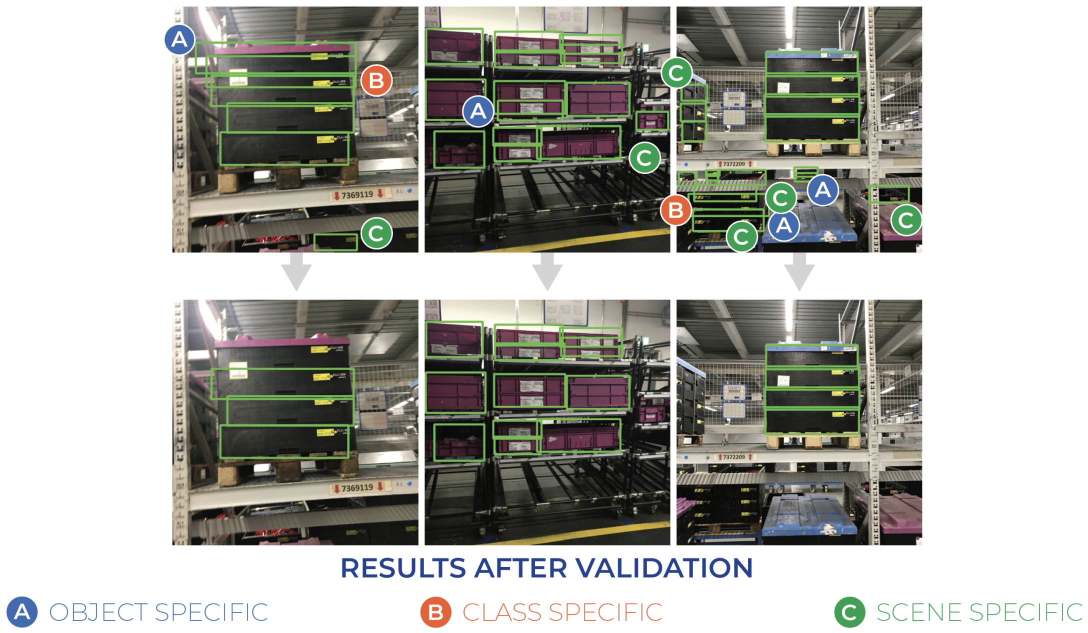


Figure 5.12: Validation results (own illustration)

successive frames at the plant Leipzig in the area of assembly supply. The scene was not changed. Neither the camera setup nor the robot position, to best represent the temporal sensor noise as well as the influence of the environment (in this case vehicles passing on the assembly line).

5.4.2 Concepts

To find the best suitable approach for the aggregation of the requested results in the following different approaches is introduced.

Approach 1

The intuitive solution is a verification process applied to successive results of the vision system. The confidence that a detection of a static object is a correct detection increase with the number of times it was detected in a row, despite the changes in the environment.

The problem could be formalized as validating the selected target if this target was detected R times in a row. For instance, if a failure occurs caused by a temporary

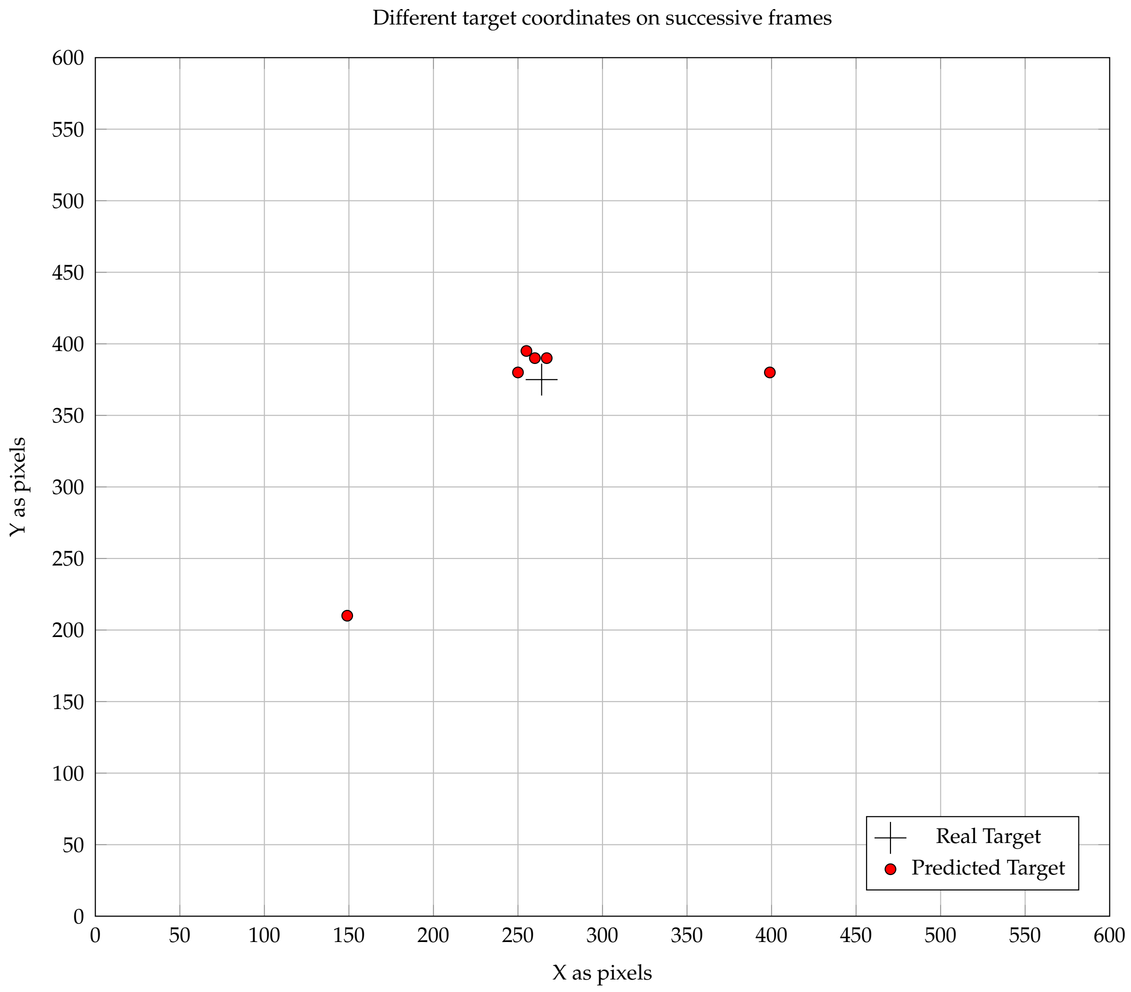


Figure 5.13: Pixel coordinates of a static target on 10 successive frames with fixed camera setups under variant environment conditions (own illustration)

occlusion, the vision system will find a difference between the current result in time t and the previous one in time t_{-1}. When a deviation in the selected target is observed in one frame compared to the previous one, this result is considered invalid, and the process for determining the target is restarted.

The $Target_t$ selected in the $frame_t$ is validated if it satisfies the condition in 5.2.

$$Target_t = Target_{t-1} = Target_{t-2} = ... = Target_{t-R} \tag{5.2}$$

Thus, R is the number of times that the vision system needs to generate the same result in a row to be considered as a valid output.

The main limitation, in this case, is, that each time an outlier occurs, the entire process has to restart without taking the previous detections into consideration. This solution is robust but time-consuming.

Approach 2

The second solution is to apply the Perception Algorithm on a fixed set of frames N. For each frame, the vision system predicts the pixel coordinates of the target. Before validating the output, the outliers are rejected, and only persistent points are considered in the pose estimation of the target. To find the outliers among all predicted targets, the density-based spatial clustering of applications with noise, shortly DBSCAN, is used. Clustering is a technique for partitioning data into groups where members of a group are similar in some way and dissimilar to members outside the group [TSK06]. DBSCAN is a density-based algorithm for discovering clusters [Est+96]. As the name suggests, the key idea of this algorithm is based on how dense the data points are located. Thus, DBSCAN clusters together a minimum number of points (minPts) that are in the same neighborhood with radius ϵ, while measuring the distance between them [Est+96]. Additionally, it identifies the points that are in low-density regions as outliers. This procedure works on a parametric approach involving two parameters, namely minPts and ϵ.

Procedure:

1. Index all points,and label all as '0'
2. foreach point:
    a) If it is labelled already, goto next point.
    b) Get points (neighbours) within 'ϵ' distance from the chosen point.
    c) If # of neighbours $<$ 'minPts', label as NOISE(-1) and goto next point.
    d) Else, label it as a new cluster (c_{new}).
    e) select the neighbouring points as a new set 'S', for each point in 'S':
        i. If labelled as -1, relabel to new cluster number (c_{new}), and goto next point in the set.

ii. If point is already labelled, skip it.

iii. If it is unlabelled, then label it (c_{new}),
Get newighbours in ϵ boundary and if count $>$ minPts add to the set 'S'

iv. continue to next point in the set, if set is empty, the go back to step (a)

Result of DBSCAN applied on the same sequence in 5.13 with minPts equal to 4 selected targets and ϵ to 20 pixels is illustrated in 5.14. Once the outliers are identified, they can

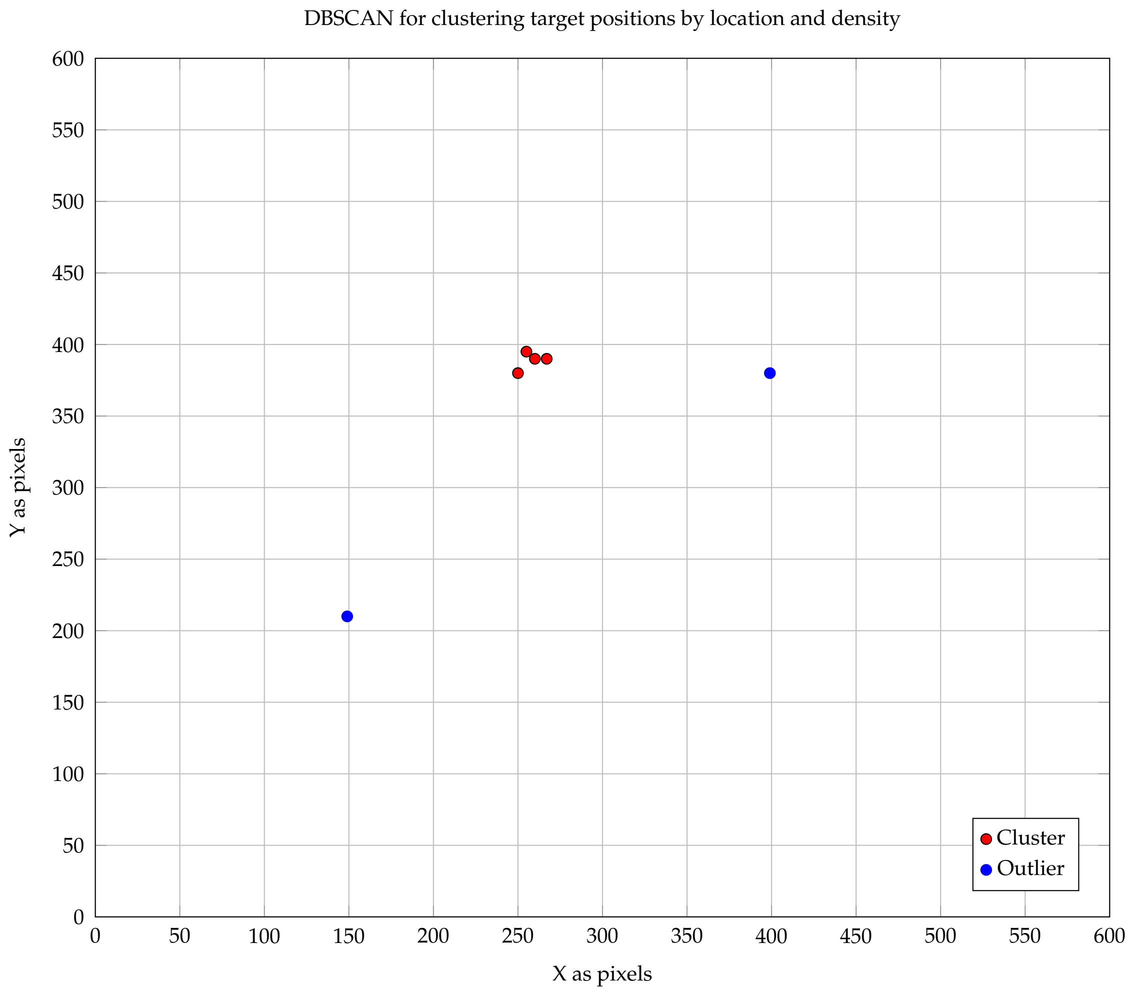


Figure 5.14: Result of DBSCAN applied on the same frame sequence as in 5.13 (own illustration)

be discarded, and the validated output is computed as the median of the most dense cluster.

The main limitation is that the number N of considered frames is fixed. Thus, the verification process is not dynamic. For instance, if a cluster is not found among the N predicted targets, the process is interrupted or restarted from the beginning.

Combination

To overcome the mentioned limitations and keep the number of sensitive parameters to a minimum, a combination between the first and the second solution is proposed. In this case, the aggregation method is a dynamic clustering, where only the validating repetition number R needs to be set initially. In other words, a selected target is validated if it was selected at least R times before. This condition does not specify that the R selections must to be in a row. Targets are detected along with successive frames until the minimum number of repetition R is reached. During the process, if a deviation occurs, it will be automatically discarded. The algorithm used for clustering is DBSCAN. The total number of frames needed for verification is equal to:

$$N = R + n_{outliers} \tag{5.3}$$

The number of frames N depends on the performance of the vision system and the validating number of repetition R. Since, computational time increases with the increase of several frames N, finding the optimal value of R should not be underestimated. A generic value is hard to determine since it strongly depends on the vision system performance in a specific environment. For this reason, experiments in the working environment of each robot have to be carried out, and the performance of the Perception Algorithm have to be recorded for different values of R, in order to define a suitable value.

5.5 Conclusion

This chapter aimed to increase the robustness of the Perception Algorithm to establish the industrial suitability of the logistics robots. For this purpose, the three functional modules (Chapter 4) were supplemented by two further modules based on thorough error analysis and their clustering.

In the Validation-Module, the recognized objects are validated in a multi-stage procedure. Criteria are first applied to the object itself, the same objects within a class, and the appearance of objects of different classes. This can effectively reduce unrealistic detections.

In the Aggregation-Module, the results of the four already explained modules are summarized. This result set is then clustered using a machine learning algorithm. Finally, the Centroid of the largest cluster is determined as the new gripping point. This means that outliers can be completely minimized.

After the effectiveness of each of these steps has been individually tested in the specific chapters on generic test data, the overall function is systematically validated in the following chapter.

6 Validation

In this chapter, the Perception Algorithm, which finally consists of five modules, as developed in the two previous main chapters, is evaluated in its entirety. For this purpose, the performance in the three main areas of application mentioned above, palletizing, provisioning, and depalletizing, is examined.

Due to the already advanced project status of the palletizing robot, the performance of the Perception Algorithm can be thoroughly tested in series production at the Leipzig plant. The performance of the Perception Algorithm when used in a depalletizing robot could also be tested in a real environment. Due to the prototype hardware used, however, not in series operation, but a separate test mode under comparable conditions. Since the provisioning robot requires additional functionalities that were not considered in the present paper, such as the omnidirectional approach of the robot, and because of its technical complexity, tests in the real environment were only possible to a minimal extent. Permanent use of such a test platform would have a negative influence on productivity by influencing the processes on the assembly line. Therefore, the tests were first carried out in a laboratory environment and then exemplarily on an assembly line section.

6.1 Palletizing

Since the end of 2018, four palletizing robots have been used in series production at the BMW plant in Leipzig. As shown in Figure 6.1, three different container types are palletized. The fourth robot is mainly used for training and demonstration purposes or as a fallback strategy in the event of a defect in one of the three other robots. Due to the advantages of serial application, no specific experiments were carried out to validate the object recognition algorithms. Instead, the real material flow was used to evaluate the performance of the devices under real industrial conditions.

Together with an availability of 95 to 97 %, the three systems proved to be sufficiently

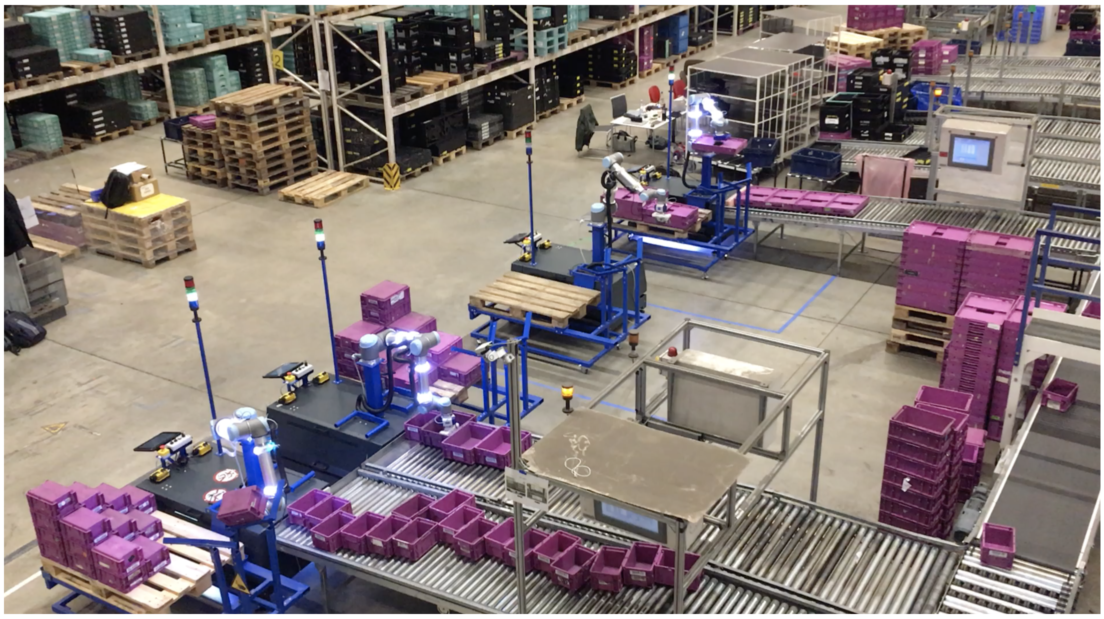


Figure 6.1: Palettizing robots working productively in serial processes of the plant Leipzig (own illustration)

robust for series use. No errors occurred in the Perception Algorithm itself. In some cases, however, containers that were not entirely taught were delivered to the retrieval stitch. These were also correctly not recognized by the Perception Algorithm since gripping such containers would lead to damage to the gripper. However, to prevent such process incidents from blocking the entire process, two additional detection points were integrated into the robot's path planning. If no container can be detected at the first point, it is tried again at the second point - with a view to other containers. As a result, the cycle time could be reduced in addition to the already sufficiently high availability.

6.2 Depalletizing

In this experiment, the two modules were implemented in the same experimental setup as in the first validation of the original Perception Algorithm. After it could be shown in the previous subchapters that the new modules were able to increase performance in extracted experiments, their influence can now be demonstrated by comparing them with the earlier results.

The robot was transported to the goods receiving area of a vehicle plant and integrated into the production process there. The depalletization of the container type with the highest proportion of the total material flow was in the foreground.

The results shown in the tables below illustrate how the proposed validation and Aggregation Module improved the perception of the robot while depalletizing in the plant.

Table 6.1: Depalletizing - real environment - Perception Algorithm with added Aggregation-Module

Module	**Performance Increase**
Detection-Module	+2%
Selection-Module	+-0%
Localization-Module	+10%

These results show that pure averaging has only an insignificant influence on the detection of objects. The impact of averaging on the determination of the gripping point is much more significant. This often fails if the depth data is not too noisy or damaged. Since an Intel RealSense D435 [Int18], which is a consumer segment 3D camera, was chosen for the application, this phenomenon was frequently observed.

Table 6.2: Depalletizing - real environment - Perception Algorithm with added Validation-Module

Module	**Performance Increase**
Detection-Module	+8%
Selection-Module	+-0%
Localization-Module	+-0%

This table shows that the Validation-Module eliminates the error susceptibility of the Detection-Module almost wholly. Since it only starts with the detection, the non-existent influence is evident with the other modules.

Table 6.3: Depalletizing - real environment - Perception Algorithm with added Aggregation- and Validation-Module

Module	**Performance Increase**
Detection-Module	+9%
Selection-Module	+-0%
Localization-Module	+10%

6.3 Assemblyline Supply

For depalletizing, the Perception Algorithm has to be tested for the provisioning robot as well.

The implemented vision-based container localization system aims to assist the robot positioning in allowing a successful gripping process. The vision module gives feedback about the position and orientation of a target container, which enables the mobile platform to reach the desired location. Once the new position is reached, the container can be successfully unloaded from the shelf.

6.3.1 Environment 1

Different experiments are conducted to establish the feasibility of the proposed localization system. To investigate the performance of the vision system, the first series of experiments were conducted under optimal conditions in the laboratory with a complete rearrangement of the scene.

The robot is placed in the laboratory in front of a similar shelf, that can be founded in the assembly line (Figure 6.2). Different robot start positions and target locations are tested (a total of 100 combinations). These experiments are designed to analyze the target detection of the vision system and positioning capabilities of the robot in a static and simple environment.

6.3.2 Environment 2

In the second series of experiments, the conditions are different. This experiment is designed to analyze the vision and positioning capabilities of the robot in a dynamic environment. Therefore, the robot is placed in the assembly line under real conditions, where the environment is highly changing (Figure 6.3). For instance, workers are walking close to the robot, and objects are constantly moving like cars in the background, and objects are additionally placed in the environment close to the target. A total of 100 tests are carried out.

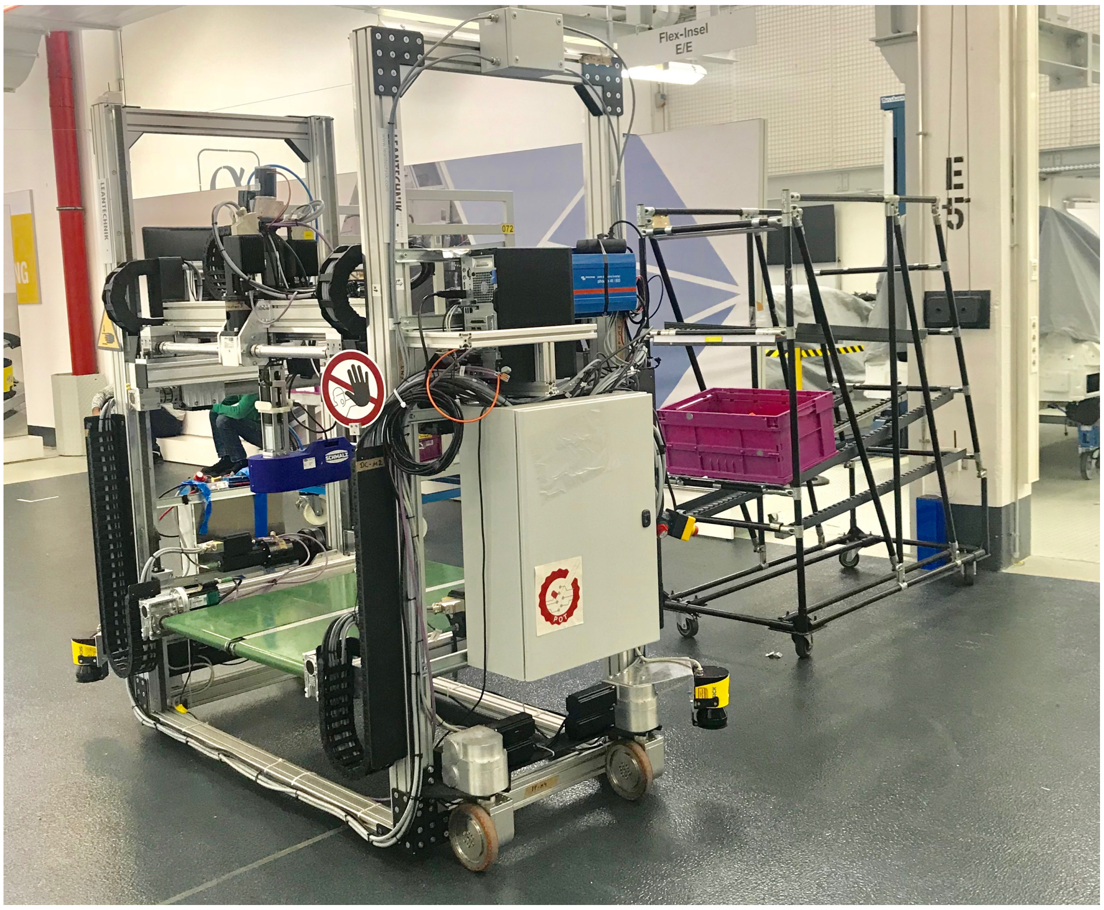


Figure 6.2: Provisioning robot in experimental development area (Munich) (own illustration)

6.3.3 Evaluations

Evaluation 1

To evaluate the effectiveness of the Detection-Module, results of the object detector are recorded, and the error rate is computed based on the following definitions of success and failure.

- Detection-Module
  - Success: Whenever the real target is among the detected objects, and its predicting bounding box overlaps with the target ground truth bounding box with an IoU over 90%.
  - Failure: Otherwise.

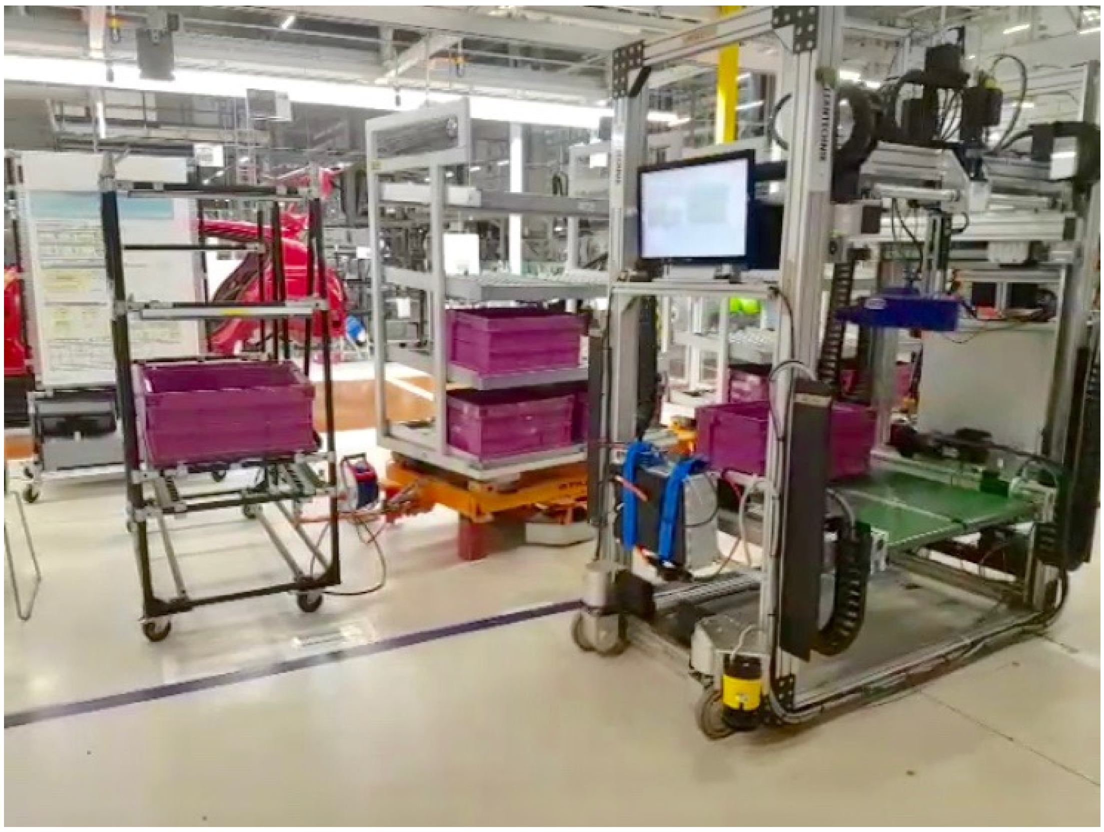


Figure 6.3: Provisioning robot in real plant logistics area (Leipzig) (own illustration)

Additionally, the second definition of error rate is used to evaluate the Selection-Module without and with the application of Validation-Module:

- Selection-Module
  - Success: Whenever the selected bounding box overlaps with the target ground truth bounding box with an IoU over 90%.
  - Failure: Otherwise.

The first results reporting the error rates in the simple environment (Innovation Lab) are summarized in Table 6.4. In this case, the background is static. Although illumination changes always occur, and few foreign objects are randomly placed close to the target.

Modules	Detection	Selection	Validation + Selection
Error Rate	4%	9%	5%

Table 6.4: Static environment: Error rate of the vision system in the laboratory

The object detector performs with a low error rate of 4%, following the definition of “Success / Failure” mentioned before. However, the Selection Module applied directly after the detection increases the error rate by 5%. This result shows that, despite the correct detection of the target, a wrong object can be selected. Finally, the application of the Validation Module before the target selection improves the result and allows the vision system to perform with an overall error rate of 5%.

During the second experiments, the error rate increases compared to the static environment. The object detector performs with an error rate of 11%. Furthermore, the target selection without validation succeeds only in 77% of the cases. Thanks to the validation, that reduces the task false positives rate, an overall error rate of 14% is recorded, despite the complexity of the environment.

Modules	Detection	Selection	Validation + Selection
Error Rate	11%	23%	14%

Table 6.5: Dynamic environment: Error rate of the vision system in industrial environment

Experimental results show that once the Validation-Module is employed, the target detection algorithm performs more accurately, particularly in indoor complex and dynamic scenes. A success rate of performance (86% in real conditions) is then achieved. When analyzing the observed errors over different tests, two main factors are depicted. First, the validation was not able to reject all the TFPs. Secondly, if a failure occurs in the Detection-Module (the target is not detected), another object with the same properties can be validated and thus selected, which yields an error.

The vision system has to achieve high performance in terms of precision and speed. Therefore, the computation time for different modules of the positioning system is additionally provided in Table 6.6.

The recorded time shows that one loop needs for selecting a target from one frame approximately 3.65*s*. The most time-consuming step is loading the weights of the detector, which has to be executed once after the startup of the robot.

Module	Average Time
Detection	3.64s
Loading Yolo weights	3.59s
Prediction	0.05s
Validation	0.8ms
Selection	0.1ms
Total (+ Processing)	3.65s

Table 6.6: Average computation time

Evaluation 2

The purpose of the second evaluation is to investigate the robustness of the system against temporal changes and dynamics in the assembly line environment.

Therefore, the robot activates the vision system several times from the same start position. The location of the target remains the same along one sequence (10 frames). The resulting predictions of the target's coordinates in each frame are recorded. Despite the unchanged settings, outliers among the predictions are observed.

To overcome this problem and improve the robustness of the vision system, the aggregation method, derived in the section, is applied. The proposed aggregation method refers to a dynamic clustering process, that rejects these outliers and groups together similar results in a cluster with a minimum number R of points. Once a cluster is formed, the output can be validated. In other words, it validates the output that was at least predicted R times. Therefore, several validating repetitions R has to be defined. To this end, the average error rate of the target detection algorithm is computed for different values of R.

During the experiments, when outliers in a sequence of frames occur, they are observed on an average of 2 frames and in most cases on 2 successive frames.

Thus, the correlation of the error rate and R in Figure 6.4 can be justified. Based on this result, R equal to 3 is a suitable value for a robot working in the assembly line, that reduces the error rate to 11.5%.

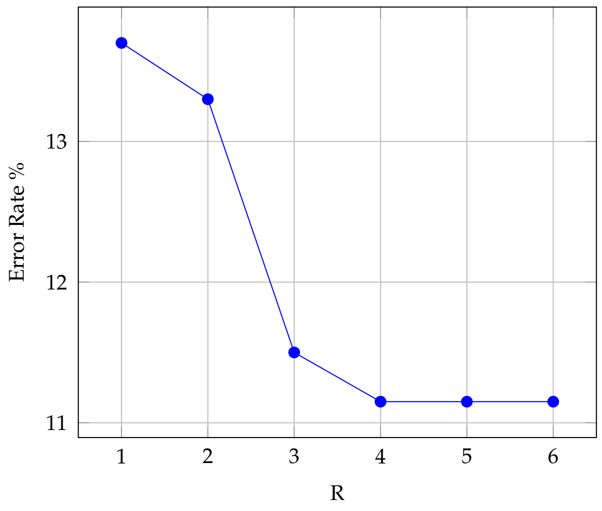


Figure 6.4: Correlation of average error rate and the required repetition number of predicted targets for validation (own illustration)

Evaluation 3 To evaluate the entire vision system, including target pose estimation module using depth images and camera intrinsic parameters, the last recorded error rates represent the failure rate of the robot in reaching the desired position.

- Vision System
  - Success: Whenever the robot reaches the desired relative position with a small tolerance of $3cm$ allowing the gripping of the container.
  - Failure: Whenever the robot does not reach the desired relative position.

Table 6.7 depicts the results of these experiments. In the laboratory, the average failure rate of 9.3% is recorded. However, the positioning of the robot was performed with an error rate of 16.7%, in the assembly line.

Environment	Static	Dynamic
Error Rate	9.3%	16.7%

Table 6.7: Positioning error rate

The increase of the error rate, compared to the target detection performance, is mainly due to the errors in-depth information provided by the camera. This is measured with the depth root mean square error (RMSE) rate. The accuracy of the depth data given by the camera tends to decrease in the distant part of the frame. For Realsense D435 used the RMSE is estimated to be 20mm for an object in a distance of 2m from the camera.

6.4 Conclusion

Overall, the performance of the Perception Algorithm can be significantly increased by adding the two new modules, as confirmed in Table 6.3.

Having a look to the stated requirements, the Perception Algorithm can now be considered as production-ready to be deployed onto the robots as long as it can be assured that a certain amount of process factors can be kept.

7 Summary and Outlook

In this final chapter, the doctoral thesis is summarized, and an outlook into future research areas is given.

7.1 Summary

The thesis of the present doctoral thesis, which was presented in the introductory chapter, was that automation of complex dynamic processes in logistics under aggravated industrial conditions could be made possible by the use of intelligent robots.

In the previous chapter, it could already be shown that the Perception Algorithm - the core component of intelligent robots - can successfully meet the requirements placed on it. If this is implemented in the respective robots, it could also be shown that the robots can or will be used in the processes as soon as the final series hardware is used.

In order to achieve this objective, the following thematic areas were touched upon.

For this reason, automation solutions already used in industry were initially presented. Building on this, the properties of the underlying processes were analyzed. This analysis of the process characteristics was then carried out for the logistics in the high-variant assembly and compared with the results of the first analysis. From the deviations derived from this, it was shown on the one hand that it is not possible to automate logistics processes with standard automation technology. On the other hand, requirements for automation solutions were derived from the differences in the processes and generic, and modular robot concepts were presented for palletizing empties and handling full containers in incoming goods as well as for belt preparation on the assembly line. These robot concepts were described according to their functional clusters concerning perception, decision, action, interaction, and safety.

Intelligent robots are necessary to overcome the challenges of the high planned and unplanned variance of the objects to be handled, their dynamic changes, and the general environmental influences of the industrial logistics environment. The central part of the work, therefore, focused on the conception, development, implementation, and validation of Perception Functions. Based on literature analysis of the current state of the art, the concept of the Perception Algorithm was derived. A modular approach was chosen to enable the best possible accuracy and robustness as well as to maintain adaptability to different applications and framework conditions. In this case, the objects are first detected, the relevant ones selected, and then their gripping position determined.

A systematic procedure was also used to create the functions of the respective modules. Based on literature results, various possible solutions were compared, adapted, and integrated. These were subsequently supplemented with new developments. Single-shot detectors such as Yolo are used to implement object detection in the Detection-Module. This was chosen because of its high accuracy and short computing time. The selection steps in the subsequent Selection-Module were created according to the use cases. For the final determination of the gripping pose, different possibilities were implemented, compared, and finally, the use of a filter cascade applied to the depth images was selected.

Initial integration tests in the robots on the several application cases and abstract tests on synthetic data sets revealed deficits in reliability. For a more in-depth analysis of these error pattern, a mobile application was developed, which allows intuitive testing of the respective modules of the Perception Algorithm. A total of four error patterns were identified: Errors in object recognition, temporal variances due to changing scenarios, incorrect input data, and non-compliance with framework conditions in the processes.

To solve these problems, the Perception Algorithm was extended by the modules Validation and Aggregation. The former investigates the predictions achieved by neural networks at the object level concerning plausibility. The interaction of these objects in the same class is then evaluated before the overall voices of all recognized objects in a scenario are finally checked. To eliminate the temporary invariances, these modules are executed repeatedly. Their results are then summarized in the Aggregation-Module, clustered, and the most plausible target passed on.

Thanks to these additions, the robustness of the Perception Algorithm could be significantly increased in the final validation using the concrete examples of assembly line supply and depalletizing in incoming goods. Remaining errors were entirely due to wrong processes in the plants. Work instructions must remedy these.

Based on the robot concepts developed within the framework of the dissertation, three palletizing robots are in series production at the BMW plant in Leipzig as it can be seen in Figure 6.1. The depalletizing robot is currently being finalized in terms of hardware and will be integrated into the BMW plant in Dingolfing in the middle of the year after the endurance tests already carried out in the real environment. The provisioning robot will be tested in the middle of the year for further endurance tests at the plants in Regensburg and Leipzig. There, the goal is to optimize the interaction with the other process participants (route trains or other road users) and to certify the entire system with regards to safety standards.

7.2 Outlook

The already mentioned roll out of the mentioned robots shows the practical suitability of the created systems. This is particularly noteworthy as it represents the first deployment of autonomous systems based mainly on artificial intelligence in the real industrial environment under very challenging conditions.

The increasing use of these systems will open up further possibilities in the future, such as continuous learning through the possibility of generating more massive data sets. This would further increase the accuracy of the Detection-Module.

In addition to the necessary embedding of the systems in intelligent infrastructure, the problem of unbalanced datasets must be solved, and further investigation carried out to determine which robots can or may learn from which and to what extent.

A final potential that should be addressed in future work is the handling of errors in the processes. Due to the increasing degree of automation, it can be assumed that these can be significantly reduced.

If the systems created are supplemented by the ideas mentioned here, the potential for an economically sensible increase in the degree of automation in dynamic logistics processes with a wide range of variants under industrial conditions will increase increasingly. This can reduce complexity, cut costs, and relieve people from today's monotonous activities and assign them to more demanding tasks.

Bibliography

[10211] E. I. 10218-1: Industrieroboter - Sicherheitsanforderungen - Teil 1: Roboter (ISO 10218- 1:2011); Deutsche Fassung EN ISO 10218-1:2011. 2011 (cit. on p. 48).

[10212] E. I. 10218-2: Industrieroboter - Sicherheitsanforderungen - Teil 2: Robotersysteme und Integration (ISO 10218-2:2011); Deutsche Fassung EN ISO 10218-2:2011. 2012 (cit. on p. 48).

[13811] E. I. 13850: Sicherheit von Maschinen - Not-Halt - Gestaltungsleitsatze. 2011 (cit. on p. 48).

[15017] D. T. 15066: Roboter und Robotikgeraete - Kollaborierende Roboter (ISO/TS 15066:2016) Deutsche Fassung 2017-04. 2017 (cit. on p. 48).

[34914] E. 349: Sicherheit von Maschinen - Mindestabstände - zur Vermeidung des Quetschens von Koerperteilen. 2014 (cit. on p. 48).

[70111] E. I. 7010: Graphische Symbole - Sicherheitsfarben und Sicherheitszeichen - Registrierte Sicherheitszeichen (ISO 7010:2011); Deutsche Fassung EN ISO 7010:2012. 2011 (cit. on p. 48).

[9to19] 9to5Mac: iPhone XS sensor and image processing makes video 'vodo' good. 2019. URL: `https://9to5mac.com/2018/10/02/iphone-xs-low-light/` (cit. on p. 130).

[Aba+15] M. Abadi et al.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/` (cit. on p. 83).

[ABB19] ABB: Industrial Robots. 2019. URL: `https://new.abb.com/products/robotics/industrial-robots` (cit. on p. 9).

[AC15] M. V. Andulkar and S. S. Chiddarwar: Incremental approach for trajectory generation of spray painting robot. In: Industrial Robot 42.3 (2015), pp. 228–241. ISSN: 0143991X. DOI: `10.1108/IR-10-2014-0405` (cit. on p. 13).

[Ade+84] E. H. Adelson et al.: Rca84. In: (1984) (cit. on p. 57).

[al14] Z. C. L. et al: Optimal Thresholding of Classifiers to Maximize F1 Measure. In: Machine Learning and Knowledge Discovery in Databases. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 225–239 (cit. on p. 78).

[AL18] F. Altenberger and C. Lenz: A Non-Technical Survey on Deep Convolutional Neural Network Architectures. In: CoRR abs/1803.02129 (2018) (cit. on pp. 66, 67).

[Ara+05] O. Arandjelovi et al.: This is the published version : Available from Deakin Research Online : However , permission to reprint / republish this material for collective works for resale or redistribution to servers or lists , or Face Recognition with Image Sets Using Manifold Density Divergence. In: (2005) (cit. on p. 142).

[Arn+17] D. Arnold et al.: *Handbuch Logistik*. VDI Buch - Springer Verlag, 2017. ISBN: 978-3-540-72929-7 (cit. on pp. 2, 16, 18, 19, 30).

[B M14] P. B. Maind: Research Paper on Basic of Artificial Neural Network. In: International Journal on Recent and Innovation Trends in Computing and Communication 2 (2014) (cit. on p. 61).

[Bau14] T. Bauernhansl: Die Vierte Industrielle Revolution - Der Weg in ein wertschaffendes Produktionsparadigma. In: Industrie 4.0 in Produktion, Automatisierung und Logistik. Wiesbaden: Springer Fachmedien Wiesbaden, 2014, pp. 3–35 (cit. on pp. 1, 2).

[BBV00] J. Bruce et al.: Fast and inexpensive color image segmentation for interactive. In: Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000) (Cat. No.00CH37113) 3 (2000), pp. 2061–2066. DOI: `10.1109/IROS.2000.895274` (cit. on p. 53).

[BGW03] C. von Bank et al.: A Visual Quality Inspection System Based on a Hierarchical 3D Pose Estimation Algorithm. In: 2003, pp. 179–186. DOI: `10.1007/`

978-3-540-45243-0_24. URL: http://link.springer.com/10.1007/978-3-540-45243-0%7B%5C_%7D24 (cit. on p. 15).

[BMW18] BMW: BMW Intranet. 2018 (cit. on pp. 22, 23).

[Boh15] J. Bohannon: Artificial intelligence bests humans at classic arcade games. 2015. URL: https://www.sciencemag.org/news/2015/02/artificial-intelligence-bests-humans-classic-arcade-games (cit. on p. 3).

[BS04] T. Brosnan and D. W. Sun: Improving quality inspection of food products by computer vision - A review. In: Journal of Food Engineering 61.1 SPEC. (2004), pp. 3–16. ISSN: 02608774. DOI: 10.1016/S0260-8774(03)00183-3 (cit. on p. 14).

[Che+05] S. B. Chen et al.: Acquisition of weld seam dimensional position information for arc welding robot based on vision computing. In: Journal of Intelligent and Robotic Systems: Theory and Applications 43.1 (2005), pp. 77–97. ISSN: 09210296. DOI: 10.1007/s10846-005-2966-6 (cit. on p. 13).

[Chi89] T. Chin: On the Detection of Dominant Points on Digital Curves. In: (1989) (cit. on p. 110).

[Cut06] V. Cutsuridis: The perception- ... -action cycle cognitive architecture and autonomy : the view from the brain. In: (2006), pp. 36–38 (cit. on pp. 31, 32).

[CZ13] W. Chen and D. Zhao: Path Planning for Spray Painting Robot of Workpiece Surfaces. In: Mathematical Problems in Engineering 2013 (2013), pp. 1–6. ISSN: 1024-123X. DOI: 10.1155/2013/659457 (cit. on p. 14).

[Dai+16] J. Dai et al.: R-FCN: Object Detection via Region-based Fully Convolutional Networks. In: CoRR abs/1605.06409 (2016). arXiv: 1605.06409. URL: http://arxiv.org/abs/1605.06409 (cit. on p. 73).

[Dai16] J. Dai: R-FCN : Object Detection via Region-based Fully Convolutional Networks. In: Nips (2016) (cit. on p. 128).

[Dal+10] N. Dalal et al.: To cite this version : Histograms of Oriented Gradients for Human Detection. In: (2010), pp. 886–893 (cit. on p. 60).

[DBG14] S. DoÇ§an et al.: Web application testing: A systematic literature review. In: Journal of Systems and Software 91.1 (2014), pp. 174–201. ISSN: 01641212. DOI: 10.1016/j.jss.2014.01.010 (cit. on p. 47).

[Dee19] DeepMind: The history of AlphaGo so far. In: (2019). URL: https://deepmind.com/research/alphago/ (cit. on p. 3).

[Den+09] J. Deng et al.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR09. 2009 (cit. on p. 80).

[Die14] W. Diez: Produktpolitik in der Automobilwirtschaft. 2014 (cit. on p. 1).

[DMS18] N. Dvornik et al.: Modeling Visual Context is Key to Augmenting Object Detection Datasets. In: CoRR abs/1807.07428 (2018) (cit. on p. 81).

[Dyk94] H. van Dyke: Applications of Distributed Artificial Intelligence in Industry. In: Foundations of Distributed Artificial Intelligence Dove 92 (1994), pp. 1–18 (cit. on p. 3).

[Est+96] M. Ester et al.: A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD'96. Portland, Oregon: AAAI Press, 1996, pp. 226–231. URL: http://dl.acm.org/citation.cfm?id=3001460.3001507 (cit. on p. 148).

[FD04] W. Fischer and L. Dittrich: *Materialfluß und Logistik*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. ISBN: 978-3-642-62260-1. DOI: 10.1007/978-3-642-18707-0. URL: http://link.springer.com/10.1007/978-3-642-18707-0 (cit. on p. 16).

[GB13] I. Göpfert and D. Braun: *Internationale Logistik in und zwischen unterschiedlichen Weltregionen*. Wiesbaden: Springer Fachmedien Wiesbaden, 2013. ISBN: 978-3-658-02603-5. DOI: 10.1007/978-3-658-02604-2. URL: http://link.springer.com/10.1007/978-3-658-02604-2 (cit. on p. 16).

[Geh+11] D. Gehrig et al.: Combined intention, activity, and motion recognition for a humanoid household robot. In: IEEE International Conference on

Intelligent Robots and Systems (2011), pp. 4819–4825. DOI: 10.1109/IROS.2011.6048716 (cit. on p. 51).

[Ger03] C. Gershenson: Artificial Neural Networks for Beginners. In: arXiv April (2003), p. 8. ISSN: 0001-5652. DOI: 10.1093/icesjms/fsp009. arXiv: 0308031 [cs] (cit. on p. 60).

[Gir+12] R. Girshick et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: (2012), pp. 2–9 (cit. on p. 128).

[Gir+13] R. B. Girshick et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CoRR abs/1311.2524 (2013) (cit. on p. 71).

[Gir15] R. Girshick: Fast R-CNN. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). ICCV '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1440–1448. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.169. URL: http://dx.doi.org/10.1109/ICCV.2015.169 (cit. on p. 70).

[GL13] J. Gall and V. Lempitsky: Class-Specific Hough Forests for Object Detection. In: (2013) (cit. on p. 142).

[Goo19] I. Goos: Wirtschaftsförderung Dingolfing. 2019. (Visited on 03/17/2019) (cit. on p. 2).

[Gra17] R. Gray: How automation will affect you - the experts view. 2017. URL: http://www.bbc.com/future/story/20170522-how-automation-will-affect-you-the-experts-view (cit. on p. 2).

[Gsp+12] S. Gspandl et al.: A dependable perception-decision-execution cycle for autonomous robots. In: 2012 IEEE International Conference on Robotics and Automation. IEEE, May 2012, pp. 2992–2998. ISBN: 978-1-4673-1405-3. DOI: 10.1109/ICRA.2012.6225078. URL: http://ieeexplore.ieee.org/document/6225078/ (cit. on p. 31).

[Gu+17] S. Gu et al.: Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In: Proceedings - IEEE International Conference on Robotics and Automation (2017), pp. 3389–3396. ISSN: 10504729. DOI: 10.1109/ICRA.2017.7989385. arXiv: arXiv:1610.00633v2 (cit. on p. 53).

[Gün11] W. Günthner: Effiziente Logistik - zentraler Erfolgsfaktor im Nutzfahrzeugbau. In: fml - Lehrstuhl für Fördertechnik Materialfluss Logistik - Technische Universität München (2011) (cit. on p. 1).

[GW17] R. Gonzalez and R. Woods: *Digital Image Processing*. 2017. ISBN: 9780133356724 (cit. on p. 109).

[Haa08] B. Haasdonk: Digitale Bildverarbeitung Einheit 8 Lineare Filterung. In: (2008), pp. 1–11 (cit. on pp. 104, 105).

[Had+14] N. N. Hadidi et al.: Functional Magnetic Resonance Imaging as Experienced by Stroke Survivors. In: Research in Gerontological Nursing 7.5 (2014), pp. 200–205. ISSN: 1940-4921. DOI: 10.3928/19404921-20140820-01. arXiv: arXiv:1312.2249v1. URL: http://www.healio.com/doiresolver?doi=10.3928/19404921-20140820-01 (cit. on p. 60).

[Hau13] M. Haun: *Handbuch Robotik*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. ISBN: 978-3-642-39857-5. DOI: 10.1007/978-3-642-39858-2. URL: http://link.springer.com/10.1007/978-3-642-39858-2 (cit. on pp. 8–13, 29).

[Hei16] Heidrive: Heidrive Manual. In: (2016) (cit. on p. 36).

[Hel+09] S. Helmer et al.: Semantic Robot Vision Challenge: Current State and Future Directions. In: Arxiv preprint arXiv09082656 (2009). arXiv: arXiv:0908.2656v1. URL: http://arxiv.org/abs/0908.2656 (cit. on p. 51).

[HST13] K. He et al.: Guided Image Filtering. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 35.6 (June 2013), pp. 1397–1409. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.213. URL: http://ieeexplore.ieee.org/document/6319316/ (cit. on p. 103).

[Hu97] J. S. Hu: Stream-of-Variation Theory for Automotive Body Assembly. In: 46.1 (1997), pp. 1–6 (cit. on p. 12).

[HW07] M. Hülsmann and K. Windt: Understanding Autonomous Cooperation and Control in Logistics. In: Understanding Autonomous Cooperation and Control in Logistics (2007). DOI: 10.1007/978-3-540-47450-0 (cit. on p. 19).

[HY12] J. Huang and S. You: Point cloud matching based on 3D self-similarity. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (2012), pp. 41–48. ISSN: 21607508. DOI: `10.1109/CVPRW.2012.6238913` (cit. on pp. 53, 54).

[Ihm06] J. Ihme: *Logistik im Automobilbau*. München: Carl Hanser Verlag GmbH & Co. KG, Mar. 2006. ISBN: 978-3-446-40221-8. DOI: `10.3139/9783446408623`. URL: `http://www.hanser-elibrary.com/doi/book/10.3139/9783446408623` (cit. on pp. 16, 17).

[Int18] Intel: Intel RealSense Depth Camera D435. 2018. URL: `https://click.intel.com/intelr-realsensetm-depth-camera-d435.html` (cit. on p. 154).

[J R17] A. F. J. Redmon: YOLO9000: Better, Faster, Stronger. In: CVPR. IEEE Computer Society, 2017, pp. 6517–6525 (cit. on pp. 76, 79).

[J R18] A. F. J. Redmon: YOLOv3: An Incremental Improvement. In: CoRR abs/1804.02767 (2018) (cit. on pp. 75–77).

[K O15] R. N. K. OShea: Introduction to Convolutional Neural Networks. In: 2015 (cit. on p. 66).

[KB14] D. Kingma and J. Ba: Adam: A Method for Stochastic Optimization. In: CoRR abs/1412.6980 (2014) (cit. on pp. 64, 65).

[KET07] C. C. Kemp et al.: Challenges for robot manipulation in human environments [Grand challenges of robotics]. In: IEEE Robotics and Automation Magazine 14.1 (2007), pp. 20–29. ISSN: 10709932. DOI: `10.1109/MRA.2007.339604` (cit. on p. 15).

[Kio+01] T. K. Kiong et al.: *Precision Motion Control*. Advances in Industrial Control. London: Springer London, 2001. ISBN: 978-1-4471-3693-4. DOI: `10.1007/978-1-4471-3691-0`. URL: `http://link.springer.com/10.1007/978-1-4471-3691-0` (cit. on p. 44).

[KKS18] A. Koster et al.: Five trends transforming the Automotive Industry. In: PwC 1.1 (2018), pp. 35–45 (cit. on p. 1).

[Klu18] F. Klug: *Logistikmanagement in der Automobilindustrie*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018. ISBN: 978-3-662-55872-0. DOI: 10.1007/978-3-662-55873-7. URL: http://link.springer.com/10.1007/978-3-662-55873-7 (cit. on p. 22).

[KPL10] J. Katrasnik et al.: A Survey of Mobile Robots for Distribution Power Line Inspection. In: IEEE Transactions on Power Delivery 25.1 (Jan. 2010), pp. 485–493. ISSN: 0885-8977. DOI: 10.1109/TPWRD.2009.2035427. URL: http://ieeexplore.ieee.org/document/5345712/ (cit. on p. 14).

[Kra04] F. Kramer: Rauschunterdrückung in der Theorie & Praxis. In: (2004) (cit. on p. 105).

[Kro+15] J. B. Krolczyk et al.: Material flow optimization â a case study in automotive industry. In: Tehnicki vjesnik - Technical Gazette 22.6 (2015). ISSN: 13303651. DOI: 10.17559/tv-20141114195649 (cit. on p. 19).

[KT81] R. Kruger and W. Thompson: A technical and economic assessment of computer vision for industrial inspection and robotic assembly. In: Proceedings of the IEEE 69.12 (1981), pp. 1524–1538. ISSN: 0018-9219. DOI: 10.1109/PROC.1981.12199. URL: http://ieeexplore.ieee.org/document/1456467/ (cit. on p. 14).

[Küb+16] C. Küber et al.: Planning Method for the Design of Flexible as Well as Economic Assembly and Logistics Processes in the Automotive Industry. In: Procedia CIRP 41 (2016), pp. 556–561. ISSN: 22128271. DOI: 10.1016/j.procir.2015.12.038. URL: http://dx.doi.org/10.1016/j.procir.2015.12.038 (cit. on p. 7).

[Kuk16] Kuka: Roboter-Achterbahn. 2016 (cit. on p. 16).

[KUK19] KUKA: Industrieroboter von KUKA. 2019. URL: https://www.kuka.com/de-de/produkte-leistungen/robotersysteme/industrieroboter (cit. on pp. 9, 11, 12).

[La 94] B. J. La Londe: Evolution of the Integrated Logistics Concept. 1994 (cit. on p. 19).

[Lap12] G. Laput: Reducing False Alarm Occurrences in Vehicle Detection Systems. In: (2012) (cit. on p. 142).

[Lit+18] G. Litzenberger et al.: Automation boom in electrical/ electronics industry drives 30% increase in sales of industrial robots. 2018. URL: https://ifr.org/post/automation-boom-in-electrical-electronics-industry-drives-30-increase-in-sa (visited on 03/18/2019) (cit. on p. 2).

[LLS15] I. Lenz et al.: Deep learning for detecting robotic grasps. In: International Journal of Robotics Research 34.4-5 (2015), pp. 705–724. ISSN: 17413176. DOI: 10.1177/0278364914549607. arXiv: 1301.3592 (cit. on p. 53).

[MAD04] V. C. Moulianitis et al.: A model for concept evaluation in design - An application to mechatronics design of robot grippers. In: Mechatronics 14.6 (2004), pp. 599–622. ISSN: 09574158. DOI: 10.1016/j.mechatronics.2003.09.001 (cit. on p. 10).

[Mah+17a] J. Mahler et al.: Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics. In: (2017). ISSN: 0929-5593. DOI: 10.15607/RSS.2017.XIII.058. arXiv: 1703.09312. URL: http://arxiv.org/abs/1703.09312 (cit. on pp. 53, 55).

[Mah+17b] J. Mahler et al.: Dex-Net 3.0: Computing Robust Robot Vacuum Suction Grasp Targets in Point Clouds using a New Analytic Model and Deep Learning. In: (2017). ISSN: 2577-087X. DOI: 10.1109/ICRA.2018.8460887. arXiv: 1709.06670. URL: http://arxiv.org/abs/1709.06670 (cit. on pp. 53, 55).

[Mar11] C. Martin: *Logistics & Supply Chain Management*. Vol. 48. 4. 2011, p. 276. ISBN: 9780273731122. DOI: 10.1007/s12146-007-0019-8. arXiv: arXiv:1011.1669v3. URL: http://www.springerlink.com/openurl.asp?id=doi:10.1023/B:BTTJ.0000047119.22852.38 (cit. on p. 22).

[MH10] H. Martínez-Barberá and D. Herrero-Pérez: Autonomous navigation of an automated guided vehicle in industrial environments. In: Robotics and Computer-Integrated Manufacturing 26.4 (Aug. 2010), pp. 296–311. ISSN: 07365845. DOI: 10.1016/j.rcim.2009.10.003.

URL: https://linkinghub.elsevier.com/retrieve/pii/S0736584509000994 (cit. on p. 8).

[MH11] J. Miemczyk and M. Holweg: Building Cars To Customer Order - What Does It Mean for Inbound Logistics Operations? In: Journal of Business Logistics 25.2 (2011), pp. 171–197. ISSN: 0735-3766. DOI: 10.1002/j.2158-1592.2004.tb00186.x (cit. on pp. 8, 18).

[Muz+12] I. W. Muzan et al.: Implementation of industrial robot for painting applications. In: Procedia Engineering 41.Iris (2012), pp. 1329–1335. ISSN: 18777058. DOI: 10.1016/j.proeng.2012.07.318. URL: http://dx.doi.org/10.1016/j.proeng.2012.07.318 (cit. on p. 14).

[Nei12] B. Neilson: Five theses on understanding logistics as power. In: Distinktion 13.3 (2012), pp. 322–339. ISSN: 21599149. DOI: 10.1080/1600910X.2012.728533 (cit. on p. 28).

[Ope18] OpenCV: OpenCV Tutorial. 2018. URL: https://opencv-python-tutroals.readthedocs.io/en/latest/ (cit. on pp. 110, 113, 114).

[Ouy+16] W. Ouyang et al.: Factors in Finetuning Deep Model for Object Detection with Long-Tail Distribution. In: CVPR. IEEE Computer Society, 2016, pp. 864–873 (cit. on pp. 79, 83).

[P Y17] Y. Z. P. Yuan: Faster RCNN with Region Proposal Refinement. In: (2017) (cit. on pp. 70–73).

[Par+07] C. Parlitz et al.: Intuitive Human-Machine-Interaction and. In: 01 (2007) (cit. on p. 51).

[Par06] E. Parlament: Richtlinie 2006/42/EG des Europaaeischen Parlaments und des Rates vom 17. Mai 2006 ueber Maschinen und zur Aenderung der Richtlinie 95/16/EG. 2006 (cit. on p. 48).

[Par14a] E. Parlament: Richtlinie 2014/30/EU des Europaaeischen Parlaments und des Rates vom 26.Februar 2014 zur Harmonisierung der Rechtsvorschriften der Mitgliedstaaten über die elektromagnetische Vertraeglichkeit. 2014 (cit. on p. 48).

[Par14b] E. Parlament: Richtlinie 2014/35/EU des Europaaeischen Parlaments und des Rates vom 26.Februar 2014 zur Harmonisierung der Rechtsvorschriften der Mitgliedstaaten ueber die Bereitstellung elektrischer Betriebsmittel zur Verwendung innerhalb bestimmter Spannungsgrenzen auf dem Markt. 2014 (cit. on p. 48).

[Pel+14] S. Pellegrinelli et al.: Multi-robot spot-welding cells: An integrated approach to cell design and motion planning. In: CIRP Annals - Manufacturing Technology 63.1 (2014), pp. 17–20. ISSN: 17260604. DOI: 10.1016/j.cirp.2014.03.015. URL: http://dx.doi.org/10.1016/j.cirp.2014.03.015 (cit. on p. 12).

[Pet06] J. Peterwitz: Grundlagen Bildverarbeitung und Objekterkennung. In: (2006) (cit. on p. 113).

[PIT16] C. Poss et al.: Sauggreifsystem sowie Verfahren zum Betreiben eines Sauggreifsystems. 2016 (cit. on p. 40).

[Qia99] N. Qian: On the Momentum Term in Gradient Descent Learning Algorithms. In: Neural Netw. 12.1 (Jan. 1999), pp. 145–151. ISSN: 0893-6080. DOI: 10.1016/S0893-6080(98)00116-6. URL: http://dx.doi.org/10.1016/S0893-6080(98)00116-6 (cit. on p. 63).

[RD01] N. Roy and G. Dudek: Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations. In: Autonomous Robots 11.2 (2001), pp. 117–136. ISSN: 09295593. DOI: 10.1023/A:1011219024159 (cit. on p. 8).

[Red16] J. Redmon: Darknet: Open Source Neural Networks in C. http://pjreddie.com/darknet/. 2013–2016 (cit. on p. 83).

[RN05] F. Rosenblatt and C. Nonr-: Frosenblatt. In: 65.6 (2005), pp. 1–23. ISSN: 1939-1471(Electronic);0033-295X(Print). DOI: 10.1098/rspb.1976.0087. arXiv: arXiv:1112.6209. URL: papers://d471b97a-e92c-44c2-8562-4efc271c8c1b/Paper/p322 (cit. on p. 3).

[ROS14] ROSWiki: SMACH ROS StateMachine. 2014. URL: http://wiki.ros.org/smach (cit. on p. 39).

[RS02] C. Rose and M. D. Smith: Mathematical Statistics With Mathematica:Mathematical Statistics With Mathematica. In: Journal of the American Statistical Association 97.460 (2002), pp. 1202–1203. DOI: `10.1198/jasa.2002.s230`. URL: `http://www.amazon.com/dp/0534377416` (cit. on p. 27).

[Rud16] S. Ruder: An overview of gradient descent optimization algorithms. In: CoRR abs/1609.04747 (2016) (cit. on pp. 63, 64).

[Sch08] M. J. Schedlbauer: *Adaptive Logistikplanung auf Basis eines standardisierten, prozessorientierten Bausteinkonzepts*. 2008, p. 200. ISBN: 9783981181975 (cit. on pp. 16–19, 30).

[Sch19] Schmalz: Spezialgreifer. 2019. URL: `https://www.schmalz.com/de/vakuumtechnik-fuer-die-automation/vakuum-komponenten/spezialgreifer` (visited on 03/19/2019) (cit. on p. 10).

[Soi98] P. Soille: *Morphologische Bildverarbeitung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. ISBN: 978-3-642-72191-5. DOI: `10.1007/978-3-642-72190-8`. URL: `http://link.springer.com/10.1007/978-3-642-72190-8` (cit. on pp. 105, 106).

[Soo14] S. Soo: Object detection using Haar-cascade Classifier. In: Institute of Computer Science, University of Tartu 2.3 (2014), pp. 1–12 (cit. on pp. 58, 59).

[SP17] F. Spenrath and A. Pott: Gripping Point Determination for Bin Picking Using Heuristic Search. In: Procedia CIRP 62 (2017), pp. 606–611. ISSN: 22128271. DOI: `10.1016/j.procir.2016.06.015`. URL: `http://dx.doi.org/10.1016/j.procir.2016.06.015` (cit. on p. 53).

[Spy17] M. Spyros: The forthcoming artificial intelligence (AI) revolution: Its impact on society and firms. In: Futures 90 (2017), pp. 46–60 (cit. on p. 3).

[Sta+14] A. Staranowicz et al.: Easy-to-use and accurate calibration of RGB-D cameras from spheres. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 8333 LNCS (2014), pp. 265–278. ISSN: 16113349. DOI: `10.1007/978-3-642-53842-1_23` (cit. on pp. 52, 54).

[SWJ17] K. T. Song et al.: CAD-based Pose Estimation Design for Random Bin Picking using a RGB-D Camera. In: Journal of Intelligent and Robotic Systems: Theory and Applications 87.3-4 (2017), pp. 455–470. ISSN: 15730409. DOI: `10.1007/s10846-017-0501-1` (cit. on p. 55).

[Sze+17] V. Sze et al.: Efficient Processing of Deep Neural Networks: A Tutorial and Survey. In: CoRR abs/1703.09039 (2017) (cit. on pp. 61, 62, 80).

[TB18] J. Tremblay and S. Birchfield: Falling Things: A Synthetic Dataset for 3D Object Detection and Pose Estimation Thang To NVIDIA. In: (2018). URL: `http://research.` (cit. on p. 55).

[Tel85] N. Telegraph: Topological Structural Analysis of Digitized Binary Images by Border Following. In: 46 (1985), pp. 32–46 (cit. on p. 110).

[TSK06] P. Tan et al.: *Introduction to Data Mining*. Pearson Education, 2006 (cit. on p. 148).

[Tsu17] Tsung: A Framework for Assessing the Sustainability of Soil and Groundwater Remediation. In: [Pdf] March (2017), pp. 1–25. ISSN: 08895406. DOI: `10.1109/ICCV.2017.324`. arXiv: `1708.02002` (cit. on p. 75).

[TSV05] H. Trussell et al.: Color image processing [basics and special issue overview. In: IEEE Signal Processing Magazine 22.1 (2005), pp. 14–22. ISSN: 1053-5888. DOI: `10.1109/msp.2005.1407711` (cit. on pp. 52, 104).

[TZW16] B. Türk et al.: Roboter im deutschen Maschinenbau. In: PwC - PricewaterhouseCoopers AG (2016) (cit. on pp. 2, 29).

[Unk18] Unknown: Market and supplier study on European robotics , service robotics. In: ipa - Fraunhofer February 2017 (2018), pp. 1–15 (cit. on p. 10).

[UR18] UR: Universal Robot UR10. 2018 (cit. on p. 41).

[Vez15] M. Vezhnevets: Context Forest for Object Class Detection. In: University of Edinburgh (2015) (cit. on p. 142).

[VJ01] P. Viola and M. Jones: Rapid object detection using a boosted cascade of. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1 (2001), pp. 511–518. ISSN: 00396028. DOI: `10.1016/S0039-6028(97)00710-3` (cit. on pp. 58, 59).

[Wei+16] L. Wei et al.: SSD: Single Shot MultiBox Detector. cite arxiv:1512.02325Comment: ECCV 2016. 2016. URL: http://arxiv.org/abs/1512.02325 (cit. on pp. 73, 74).

[WL93] B. Widrow and M. a. Lehr: Artificial Neural Networks of the Perceptron, Madaline, and Backpropagation Family BT - Neurobionics: An Interdisciplinary Approach to Substite Impaired Functions of the Human Nervous System. 1993. URL: http://isl-www.stanford.edu/%7B~%7Dwidrow/papers/c1992artificialneural.pdf%7B%5C%%7D5Cnpapers2://publication/uuid/60B788D9-9574-46C5-B369-C173D1D7410C (cit. on p. 3).

[WO14] A. Wilhelm and E. Oztop: Reinforcement learning to adjust parametrized motor primitives to new situations. In: Springer Tracts in Advanced Robotics 97 (2014), pp. 119–147. ISSN: 1610742X. DOI: 10.1007/978-3-319-03194-1_5 (cit. on p. 53).

[Woj+08] C. Wojek et al.: Sliding-Windows for Rapid Object Class Localization: A Parallel Technique. In: Pattern Recognition. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 71–81. DOI: 10.1007/978-3-540-69321-5_8. URL: http://link.springer.com/10.1007/978-3-540-69321-5%7B%5C_%7D8 (cit. on p. 57).

[YPC14] J. Yang et al.: Context Driven Scene Parsing with Attention to Rare Classes. In: (2014) (cit. on p. 142).

[Zen+17] A. Zeng et al.: Multi-view self-supervised deep learning for 6D pose estimation in the Amazon Picking Challenge. In: Proceedings - IEEE International Conference on Robotics and Automation (2017), pp. 1386–1393. ISSN: 10504729. DOI: 10.1109/ICRA.2017.7989165. arXiv: 1609.09475 (cit. on pp. 55, 93, 94).

[ZS09] B. Zhang and S. B. Skaar: Robotic de-palletizing using uncalibrated vision and 3D laser-assisted image analysis. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, Oct. 2009, pp. 3820–3825. ISBN: 978-1-4244-3803-7. DOI: 10.1109/IROS.2009.5354054. URL: http://ieeexplore.ieee.org/document/5354054/ (cit. on p. 16).

Selbstständigkeitserklärung

Name: Poss
Vorname: Christian

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Dissertation wurde in gleicher oder ähnlicher Form noch in keinem früheren Promotionsverfahren eingereicht.

Datum: Unterschrift: