Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Mechanical and Industrial Engineering

Henrik Duus Berven

Unstructured bin picking of multiple shiny objects using machine learning and 3D computer vision

Master's thesis in Robotics and Automation
Supervisor: Lars Tingelstad
June 2021

NTNU
Norwegian University of
Science and Technology

Henrik Duus Berven

Unstructured bin picking of multiple shiny objects using machine learning and 3D computer vision

Master's thesis in Robotics and Automation
Supervisor: Lars Tingelstad
June 2021

Norwegian University of Science and Technology
Faculty of Engineering
Department of Mechanical and Industrial Engineering

NTNU
Norwegian University of
Science and Technology

Title of Report

Subtitle of Report

Henrik Duus Berven

2021-06-24

Preface

This thesis marks the final chapter of my Master of Science in Engineering at the Department of Mechanical and Industrial Engineering at NTNU. The project has been a collaboration between Siemens Energy AS and the Norwegian University of Science and Technology(NTNU).

I want to thank my project supervisor Lars Tingelstad for his help and guidance the last two semesters. Jan Petersen and Lars Tore Gellein at Siemens Energy AS for their help with this project, the have provided valuable insight and resources. Martin Ingvaldsen at Zivid AS for his assistance and input on their cameras. NTNU and Siemens Energy for the opportunity. Working on this has been a great experience. When I applied for the master's program at NTNU, it was exactly these types of challenges I wanted to work on.

Sammendrag

Denne oppgaven presenterer utviklingen av et automatisert system for ustrukturert "bin picking" av blanke deler. Hensikten med systemet er at Siemens Energy AS skal bruke det for automatisk kitting av materialstativvogner som brukes til robotproduksjon av batteripakker. Metodene og teorien som brukes i systemet er presentert. Etterfulgt av hvordan de er brukt.

Delene er blanding av metallbraketter, kobling og festemateriell. Et Zivid One 3D-kamera er satt opp for skinnende gjenstander og brukes til å fange punktskyer av høy kvalitet. Punktskyen behandles i MvTec HALCON ved først å beskjære den for kun å inneholde esken med deler av interesse, deretter blir overflatematching gjort ved hjelp av "point pair features". Matchene som blir funnet blir evaluert basert på evalueringsverdi og posisjon, og den beste matchen blir videresendt. Delspesifikke plukkregler blir deretter brukt på den videresendte match "posen", og de nødvendige transformasjonene er gjort for å skape et gyldig "pose" mål for plukking. Posen sendes til en kalibrert ABB YuMi-robot som plukker delen fra esken. Systemet er programmert til å kjøre en sløyfe til alle de spesifiserte delene er plukket.

De utviklede systemprosedyrene og det komplette systemet ble testet for å evaluere ytelsen. Innledende prosedyretester viste at nøyaktigheten "hand-eye" kalibreringen og kvaliteten på den fangede punktskyen var god til å plukke med den nødvendige presisjonen. Posisjonsestimeringen ved hjelp av "point pair features". var i stand til å finne gode treff i scenen, og de anvendte plukkreglene forvandlet matchen til et gyldig robotmål. Basert på dette målet genererte robotkonfigurasjonsprosedyren den riktige konfigurasjonen for at roboten skal kunne plukke delene. Testing av der komplette systemet visste at den er i stand til å plukke flere deler. Selv om den ikke var i stand til å gjøre dette kontinuerlig over lengre perioder uten operatørhjelp.

Summary

This thesis presents the development of an automated system for unstructured bin picking of shiny parts. It is intended to be used by Siemens Energy AS for automatic kitting of material rack trolleys used in the robotic production of battery packs. The methods and theory used in the system is presented and followed by how they are implemented in the system.

The parts are mix of metal brackets, connector and fasteners that are placed unstructured in individual bins. A Zivid One 3D camera is setup for shiny objects and used to capture high quality point cloud of the bins. The point cloud is processed in MvTec HALCON by first cropping it to only contain the bin of interest, then surface matching is done using point pair features. The found matches are evaluated based on match score and position, and the best match is forwarded. Part specific picking rules are then applied to the forwarded match pose and the required transformations are done to create a valid target pose for picking. The pose is sent to a calibrated ABB YuMi robot which proceeds to pick the part from the bin. The system is programmed to run a loop until all the specified parts are picked.

The developed system procedures and the complete system were tested to evaluate the performance. Initial procedure tests showed that the accuracy of the hand-eye calibration and quality of the captured point cloud was good enough the pick with the required precision. The pose estimation using point pair feature were able to find good matches in the scene and the applied picking rules transformed the match into a valid robot target. Based on this target the robot configuration procedure generated the appropriate configuration for the robot to pick the object. Testing the complete pipeline showed that the system is capable of picking multiple parts. Though it was not able to do this continuously over longer periods of time without operator assistance.

Contents

List of Figures

List of Tables

Chapter 1.

Introduction

1.1. Project description

1.1.1. Background

This project is in collaboration with Siemens Energy AS, and their offshore marine center located in Trondheim, Norway. Siemens Energy offer products, solutions, systems, and services that covers nearly the entire energy chain to support their customers in the transition to a more sustainable world [1]. In Trondheim they develop and produce an energy storage solution called BlueVault [2] for electric and hybrid applications. A big part of their production line is automated by Intek Engineering AS [3] with hardcoded industrial KUKA robots. Siemens are now looking into further automating the production line, specifically the sorting of various metal parts, such as brackets, connectors, nuts and more from unstructured bins. These parts are used in the assembly of the battery cells and are picked by hardcoded robot in the initial stage of the production line. Today this job is done by hand and Siemens estimate that they spend approximately 2 hours on this per day. How the parts are sorted is shown in Figure 1.1

The batteries consist of battery cells with tubes and connector, for cooling and electric wiring, stacked in a cabinet. The parts that are to be picked in this assignment are going to be used in the baseplate and top plate of the battery cabinet. These plates will not be discussed or presented in detail because they are part of Siemens Energy intellectual property (IP).

This master thesis is a continuation of a specialization project in the course TPK4560 where the theoretical foundation for developing a bin picking system was established. The following summarizes the work done in the specialization project.

Research existing solutions to get an overview of the technology. Then establish

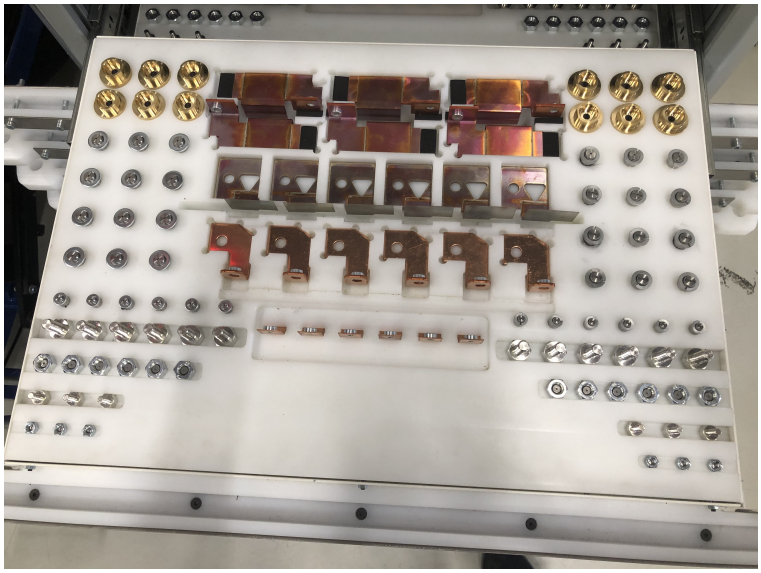


Figure 1.1.: How the parts should be sorted before going into the production line

the theory and fundamentals needed to solve a bin picking problem. Based on this develop a virtual solution that includes the following parts of bin picking: Computer vision (CV) hardware to see the environment, CV software to single out, identify and estimate the pose of objects in the scene. Use the estimated pose and robot software to generate the path a robot must move to reach that pose and move along it. Finally pick up the object and relocate it [4].

1.1.2. Problem description

This project is on the development of a system for automatic kitting of material rack trolleys for use in the robotic production of battery pack. The following objectives were given by NTNU Department of Mechanical and Industrial Engineering, and Siemens Energy AS:

- Bin-picking of shiny parts using 3D-vision and machine learning
- Grasping analysis and gripper development
- Experimental verification in the robot lab at MTP, NTNU

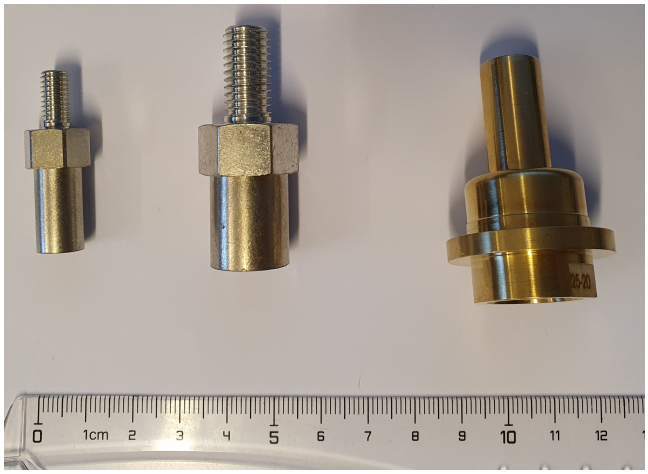


Figure 1.2.: Connectors, **left)** Small female connector (SFC), **middle)** Large female connector (LFC), **right)** brass connector

Figure 1.3.: Nuts and washers

1.1.3. The parts

For this bin picking problem there are a total of 13 individual parts to be picked, 3 connectors, 4 spacers, 2 nuts and 4 brackets.

Connectors:
Two of the connectors have a female end with a full stop, a midsection with the shape of a nut and a threaded bolt at the end. The dimension for the nut and threaded bolt are 6mm and 12mm for the small connector and 10mm and 20 mm for the large connector respectively. The male connector is made of brass, it has a varying circular cross-section, except for the bottom where two cuts have been made giving it flat surfaces used when the connector is wrenched on. The connectors can be seen in Figure 1.2

Brackets:
All of the brackets are made of copper and are shown in Figure 1.6, three of them, brackets 2,3 and 4, have threaded steel fittings fixed on them. Because two of the brackets, 1 and 4, have a very thin plate section they cannot be placed in an unstructured bin, this would cause damage to the part making them unusable. Therefore, they are stacked on a purpose made foam cutout as shown in Figure 1.4 and Figure 1.5. Brackets 2 and 3 are strong enough to be placed in an unstructured bin.

Nuts and washers:
The nuts and washers used can be seen in Figure 1.3, they are used in combination

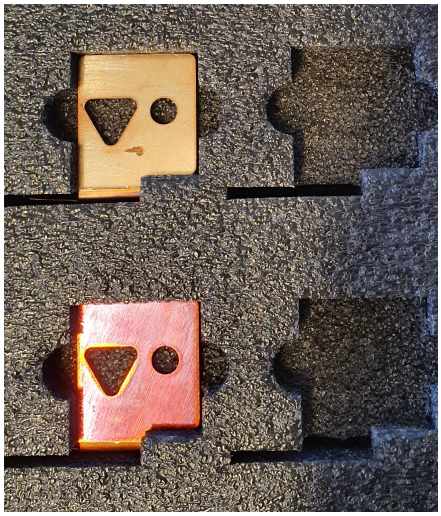


Figure 1.4.: Bracket 1) in foam

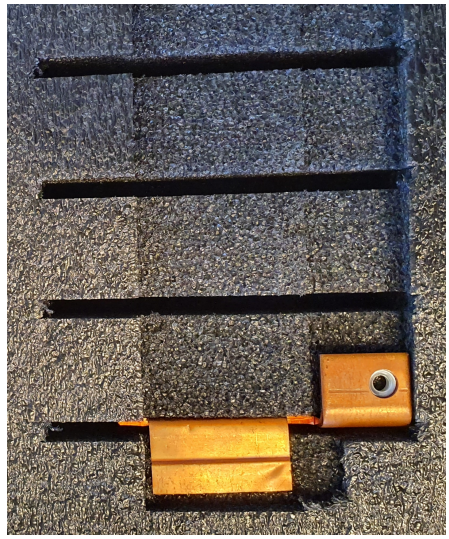


Figure 1.5.: Bracket 4) in foam

with the female connectors. There is one conventional and one split lock washer for each connector. The nuts are M6 for the small connector and M10 for the large.

1.2. Related work

There are numerous approaches to pose estimation and bin picking. Le and Lin [5] propose a deep learning approach to random bin-picking of planar objects in clustered environment. They use instance segmentation based deep learning on 2D images for classifying and localizing objects. Then extract 3D point cloud data from the 2D pixel values to build a coordinate system on the planar object plane. The approach showed promise with a reported successful pickup rate of 99% at an average processing time of 0.9 seconds per step. Lee and Lee [6] present a hybrid deep learning-engineering approach to random bin picking, it used deep learning to detect the parts in the bin and extract the associated features. These features and position are matched with a computer aided design (CAD) model, then the pose of the object is estimated using iterative closest point (ICP). The authors report a recognition rate close to 100% with this approach. Zeng, Yu, Song, *et al.* [7] propose an approach using deep learning to segment and label multiple views of the scene. Then fit pre-scanned 3D models to the resulting segmentation to get the pose of the object. They used the solution to compete in the Amazon Picking challenge and came in 3rd. Li, Liu, Guo, *et al.* [8] propose a detection and pose estimation algorithm based on Partition Viewpoint Feature Histogram

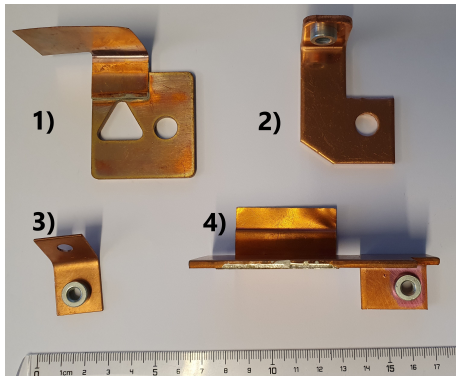


Figure 1.6.: Brackets, **1)** Triangle lask, **2)** Large lask, **3)** Small lask, **4)** Elongated lask

(PVFH) for bin picking. The model is trained offline, then when the point cloud is captured using a 3D sensor it is segmented, and the features are compared to estimate the pose. König and Drost [9] Propose a method using deep learning to segment object instances in RGB images and then using point-pair based voting methods to recover the pose. This approach was the best fast method in the 2020 BOP: Benchmark for 6D Object Pose Estimation challenge [10]. Pickit [11] offers off-the-shelf (OTS) picking solution with camera and the necessary software for machine vision and pose estimation. They offer product with an graphical user interface (GUI) and easy to understand guides, to set up a system with Pickit no coding is necessary.

1.3. Limitations

The robot and camera used for this system is ABBs YuMi and Zivid One, this was decided by the project supervisor and NTNU based on what was available at NTNU laboratories before starting the specialization project. It will therefore not be any in depth theory on different robots and cameras. However, there will be an evaluation, discussion and recommendations based on the performance of the used configuration and hardware. The ABB YuMi has a dual arm configuration. To make the solution non robot specific only one arm will be used. The machine vision software used is MvTec HALCON, this was decided in the specialization project. The project is to develop an industry ready system and using a well-established software developer gives access to a lot of established functionality. It also helps in providing a rugged solution that allows for several programming languages to be used, which also help in achieving the goal of having a non-hardware specific code. As MvTec uses point pair features (PPF) for surface matching only this

method will be described.

1.4. Structure

The structure of this report is as follows:
Chapter 2 presents the preliminary theory of robotics, computer vision, pose estimation and bin picking. Chapter 3 presents the technical information on the Zivid One camera, MvTec HALCON software and ABB YuMi robot used in the system. Chapter 4 presents the method of image acquisition for the camera, surface matching with MvTec HALCON and robot programming in ABB RAPID. Chapter 5 presents the developed system by first describing the different methods developed for setting it up followed by the initialization step that runs once at startup. Then steps in the loop is presented in the following order: image acquisition and evaluation, surface matching and target creation, robot motion, communication, error handler and a summary of the loop. Chapter 6 presents the results of the systems performance with respect to image acquisition, hand-eye calibration, surface matching, gripper design, system time and picking. Chapter 7 presents discussions of the system with respect to methods and Chapter 7 discusses the presented result and evaluates the sub tasks. Chapter 8 concludes on the outcome of the project and describes further work.

Chapter 2.

Theory

This chapter presents the foundational theory of robotics, cameras, images, 3D vision, pose estimation and bin-picking. The majority of the content is from the specialization project delivery in the course TPK4560 [4].

2.1. Fundamentals of robotics

A key concept in robotics and computer vision is being able to represent the position and orientation (pose) of a rigid body. Because, on the most basic level a robot consists of rigid bodies connected by joints. By connecting unit coordinate system to points on the joints and rigid bodies we can get a representation of the six degrees of freedom a rigid body has in 3D space. Then the configuration of the rigid body can be described by the pose relative to a specified fixed frame. This section describes the mathematical foundation and tools used to describe how objects are represented and related to each other. This lays the foundation for the kinematics used to describe robot motion.

2.1.1. Rigid body motions

Kinematics in robotics is described using unit coordinate systems as reference frames attached to points. The pose of these reference frames relative to a specified fixed frame can be represented with transformation T.

Transformation matrix

A homogeneous transformation matrix T is the representation of the combination a translation vector t and rotation matrix R, i.e., the pose.

The rotation matrix R is used to:

- Represent an orientation
- To change the reference frame in which a vector or a frame is represented
- To rotate a vector or a frame

It is in the in the special orthogonal group $SO(3)$, defined by:

$$SO(3) = R = \mathbb{R}^{3x3} : R^T R = I, detR = 1 \tag{2.1}$$

R is defined as:

$$R = \left[\hat{x}, \hat{y}, \hat{z}\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.2}$$

The rotation can also be expressed as $Rot(\hat{\omega}, \theta) = R_{\hat{\omega}}(\theta)$, where $\hat{\omega}$ is the axis of rotation and θ is the rotation amount.

The translation vector $t \in \mathbb{R}^3$ is column vector that describes the x, y, z coordinate of the origin of one frame with respect to a specified reference frame.

$$t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \tag{2.3}$$

The homogeneous transformation matrices in $\mathbb{R}^3$, is the set of all 4x4 real matrices $T = (R, t)$ of the form

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2.4}$$

The three major uses of T is:

- to represent a configuration (pose) of a rigid body
- the change the reference frame in which a vector or frame is represented
- to displace a vector or a frame

In Figure 2.1 two coordinate systems, a fixed frame {s} and the body frame {b}, are shown. The notation for the matrix representing the b in s is ${}^sT_b$, read as: "b given in the coordinates of s". ${}^sT_b$ express {b} relative to {s}

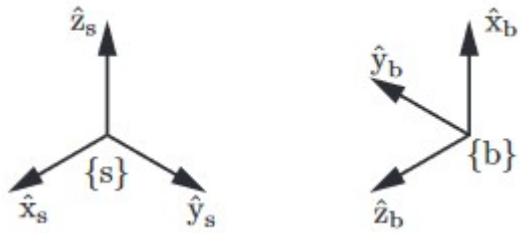


Figure 2.1.: Reference frame transformations [12]

Coordinate transformation

Figure 2.2 is used to illustrate transformations , it shows three reference frames {a}, {b}, {c} and a point v. Reference frame {b} and {c} can be represented relative to {a} by ${}^aT_b = (R_{ab}, t_{ab})$ and ${}^aT_c = (R_{ac}, t_{ac})$ and the point v can be relative to {b} with the vector v_b. The reference frame for the frame or the vector can be change using the subscript cancelation rule. This is shown in the following equations:

$$\begin{aligned} {}^aT_b\,{}^bT_c &= {}^aT_{\not b}\,{}^{\not b}T_c = {}^aT_c \\ {}^aT_b v_b &= {}^aT_{\not b} v_{\not b} = v_a \end{aligned} \tag{2.5}$$

A vector or a frame can be displaced by multiplying with a transformation $T = (R, t)$, the order of multiplication determines whether the $\hat{\omega}$-axis and t are interpreted as in frame {a} or frame {b}. This is shown in the following equations, ${}^aT_{b'}$ displaced with respect to frame {a} and ${}^aT_{b''}$ is displaced with respect to frame {b}:

$$\begin{aligned} {}^aT_{b'} &= T\,{}^aT_b \\ {}^aT_{b''} &= {}^aT_b T \end{aligned} \tag{2.6}$$

Another useful note on transformation matrices is that:

$${}^bT_a = {}^aT_b^{-1} \tag{2.7}$$

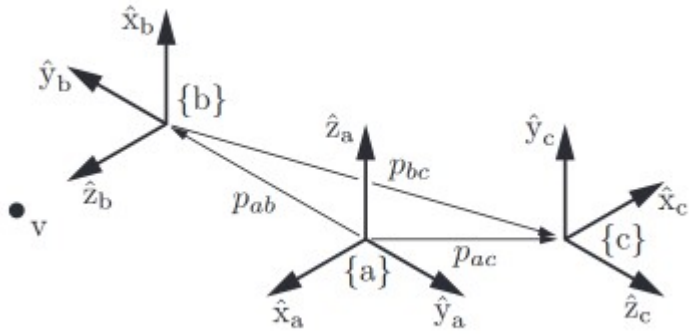


Figure 2.2.: Coordinate system transformations [12]

2.1.2. Robot configuration

When we talk about robot configuration and where it can operate we differ between a robots task space and workspace. Task space is where it task can naturally be explained, e.g. a box for bin picking. The workspace is given by the specifications of the robots, it is the space where the robot end-effector can reach.

2.1.3. Quaternions

Quaternion, q, are an alternative representation of rotations that is commonly used in robotics and CV. q is a four element vector

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \in \mathbb{R}^4 \tag{2.8}$$

given by R as :

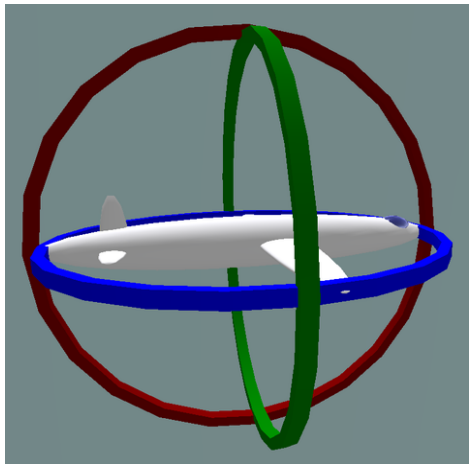
$$q_0 = \frac{1}{2}\sqrt{1 + r_{11} + r_{22}r_{33}} \quad , \quad \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \frac{1}{4q_0}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \tag{2.9}$$

q is composed of one real element and three complex elements. An intuitive description of what the quaternions represent is that the element q_1, q_2 and q_3 can be thought of as the vector that is rotated around and q_0 is the amount of rotation [12], [13].

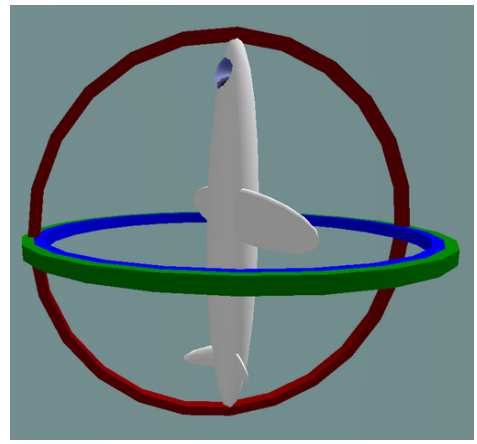
Advantages

Compared to rotational matrices, they are more compact, compared to Euler angles they avoid gimbal lock or "wrist flip" and compared to exponential coordinates they avoid sensitivity and singularities connected to the division by $sin\theta$ in the logarithm formula.

Gimbal lock is a singularity that can happen in wrist joint, a joint where the roll, pitch and yaw axis pass through the same point. In Figure 2.3b the 3 axis of rotation with a corresponding gimbal is shown, blue ring is roll, green is pitch and red is yaw. In this figure the 3 axis are independent and there are no issues. In Figure 2.3a the plan has rotated 90° so that two gimbals are in the same plane and one degree of freedom is lost. For a robot this can result in wrist flip, i.e., the robot tries to spin 180° instantly [14], [15].



(a) "No Gimbal lock" By MathsPoetry - Own work, CC BY-SA 3.0, [16]



(b) "Gimbal lock"By MathsPoetry - Own work, CC BY-SA 3.0, [17]

Figure 2.3.: Gimbal lock

2.1.4. Forward kinematics

Forward kinematics in robotics refers to determining the pose of the end-effector from the joint parameters θ. To calculate this, we attach a reference frame to each link and calculate the end-effector pose given in a specified reference frame. This can be written as product of transformation matrices, using the 3R planar open chain robot in Figure 2.4 as an example. To find the pose of the end effector frame{4} in the reference base frame {0} we need to solve (2.10).

$$ {}^{0}T_{4} = {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{4} \tag{2.10} $$

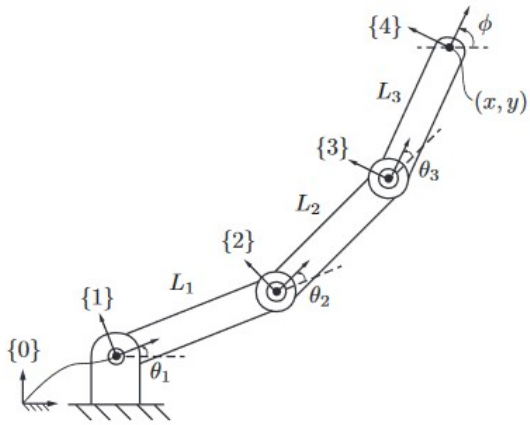


Figure 2.4.: 3R open chain planar robot [12]

2.1.5. Inverse kinematics

Inverse kinematics is the reverse of forwards kinematics, i.e., we want to find the joint parameters θ given a desired end-effector pose. This problem is more complex then forward kinematics because the desired pose can have multiple solutions as demonstrated in Figure 2.5 or no solution if the pose is outside the robots workspace. To solve the inverse kinematics problem, we use analytical methods or iterative numerical methods. It is not always possible to solve the problem using analytical methods or the accuracy of the solution is not good enough, we then use numerical methods.

2.1.6. About high DOF robots

High DOF robots give additional challenges because they are kinematic redundant, i.e., the can reach the same pose with several different configurations. This makes it difficult to program them, they require specialized software, this is a reason of why they are not used more. The reason to use high DOF robots is that they can do a multiple of tasks. Lower axis -robots, i.e., DOF <= 6 have generally been designed and developed to solve a specific task, where as a high-DOF robots can be used in multiple tasks, e.g. bin picking, assemble(screwing etc) and inspection [18].

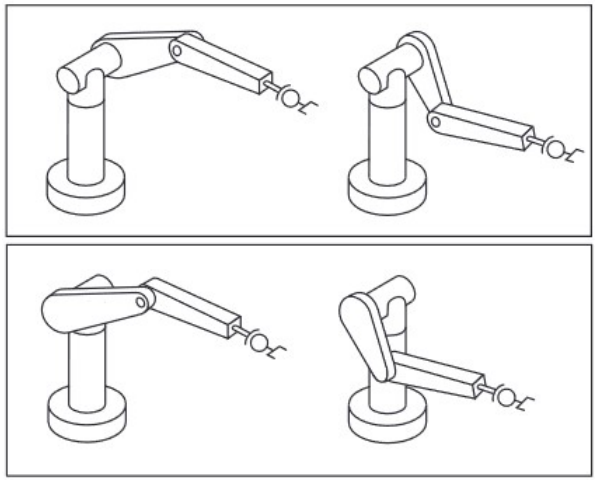


Figure 2.5.: Multiple solutions for the 6R PUMA type arm [12]

2.1.7. Singularities in 7-DOF robots

Below are a description of singularities in 7-DOF robots, they are illustrated in Figure 2.6.
Wrist singularities: This is when axis is 4 and 6 become aligned, then the TCP can be stationary but the axis move rapidly.
Shoulder singularities: The wrist center, i.e., where axis 4,5 and 6 intersect with the line drawn from axis 1.
Elbow singularities: The center of axis 2 and 3 aligns with the wrist center point. Easily avoidable by properly designing the application layout.

The advantage of a 7-DOf robot we can find configurations that can eliminate the wrist and shoulder singularities. Hollerbach [19].

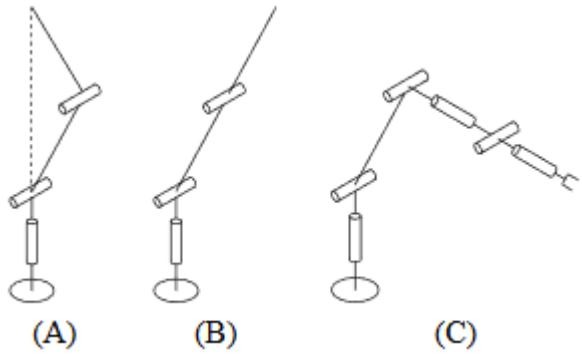


Figure 2.6.: Singularities illustrated, a) shoulder, b) elbow and c) wrist Hollerbach [19]

2.2. Camera

This section describes the mathematical model used for cameras in computer vision.

2.2.1. Pinhole camera model

The pinhole camera model is a camera model that has become the standard in computer vision (CV) to describe the essential geometric transformations used in image formations [20]. It describes the mathematical relationship between a point point $p = (x, y, z) \in \mathbb{R}^3$ and its projection onto the image plane. In the model the aperture is described as a point in the origin of coordinate camera frame {c} called the optical center, the focal plane is the x_c, y_c plane that has it origin in this center. The model does not include lenses used to focus the light.

When light from the point p passes through the optical center it hits the image sensor, i.e., retinal plane, a parallel plane to the focal plane located at a distance f in the negative z_c direction. To simplify the geometry a virtual plane, called the image plane is introduced, it is also parallel to the focal plane but located at distance f in the positive z_c direction and rotated 180° about the z_c axis. This is illustrated in Figure 2.7.

The image coordinates of the point p in the image plane is given by (2.11), where f is the focal length. A common practice is to normalize the image coordinates, i.e., setting $f = 1$, this new plane is called the normalized image plane. The equation for the normalized image coordinates then becomes as shown in (2.12).

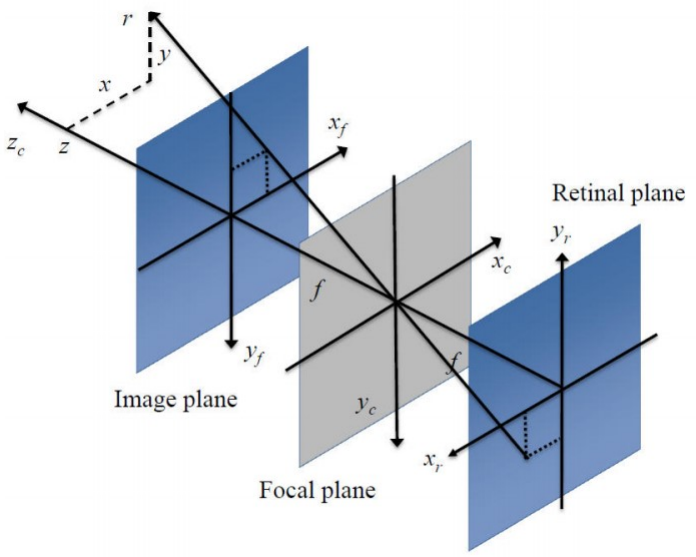


Figure 2.7.: Pinhole camera model [20]

Both planes are shown in Figure 2.8.

$$\begin{aligned} x_f &= f\frac{x}{z} \\ y_f &= f\frac{y}{z} \\ z_f &= f \end{aligned} \tag{2.11}$$

$$\begin{aligned} s_x &= \frac{x}{z} \\ s_y &= \frac{y}{z} \\ s_z &= 1 \end{aligned} \tag{2.12}$$

The vector ${}^c r_p$ describes the position of the point p in frame c, the position in the normalized image is given by the normalized image vector in homogeneous form shown in (2.13). From this we can find the homogeneous pixel coordinates $\tilde{p} = (u, v, 1)^T$ shown in (2.14), where ρ_w, ρ_h are the horizontal width and vertical height respectively and u_0, v_0 are the pixel coordinates of the center of the image plane.

$$\tilde{s} = \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix} = \frac{1}{z}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{1}{z}\,{}^c r_p \tag{2.13}$$

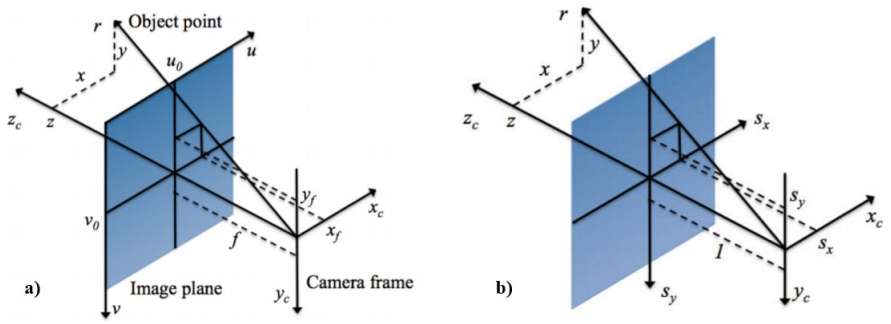


Figure 2.8.: a) image plane b) normalized image plane [20]

$$\begin{aligned} u &= \frac{f}{\rho_w} s_x + u_0 \\ v &= \frac{f}{\rho_h} s_y + v_o \end{aligned} \tag{2.14}$$

We can now describe the transformation from $\tilde{s}$ to $\tilde{p}$ as a linear transformation as shown below:

$$\tilde{p} = K\tilde{s} \tag{2.15}$$

Where $\boldsymbol{K}$ is the camera parameter matrix shown in (2.16), the elements f, ρ_h, ρ_w, u_0 and v_0 are the intrinsic camera parameters that are specific to the camera.

$$\boldsymbol{K} = \begin{bmatrix} \frac{f}{\rho_w} & 0 & u_0 \\ 0 & \frac{f}{\rho_h} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

2.3. Image basics

This section described the theory of captured images and how different camera settings affect the resulting image.

2.3.1. Light Intensity

The number of photons that hit a pixel during the reading phase is output as an intensity from the image sensor. The range of light intensity a typical image sensor can distinguish per pixel typically ranges from 256-4096 (8-12 bits) levels. Figure 2.9 shows a sin-wave signal with increasing amplitude and noise, for a signal to be quantifiable, it needs to be in the low-high SNR range. If the intensity is to

low then there is to much noise and if it is to high the pixels get over saturated, causing clipping and loss of information [21].

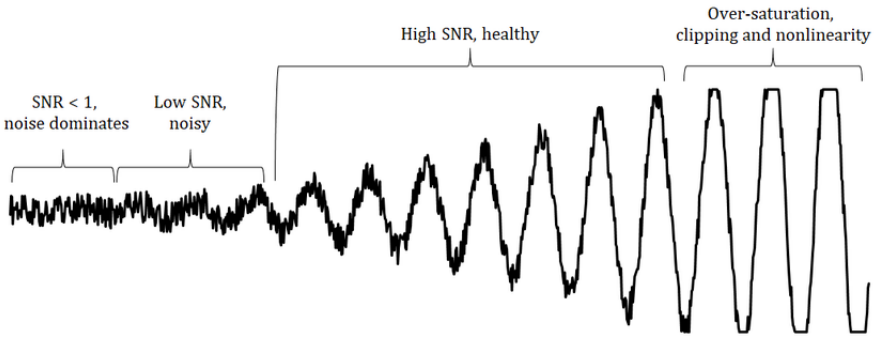


Figure 2.9.: Light intensity [21]

2.3.2. Color Temperature

Color temperature, measured in kelvin (K), is a characteristic of visible light with important applications in photography. The color temperature of a light source is given as the temperature of an ideal black-body radiator that radiates light of similar color to the light source. In a image the color temperature of ambient light affects the appearance of the colors in the image, Figure 2.10 show an image before and after the color is balanced [22], [23].

Figure 2.10.: Before and after color balancing [23]

2.3.3. Signal-to-Noise-ratio

The Signal-to-Noise-ratio (SNR) is measure of the signal strength relative to the background noise [24]. The relationship between signal level, noise and light intensity can be seen in the graph bellow Figure 2.11.

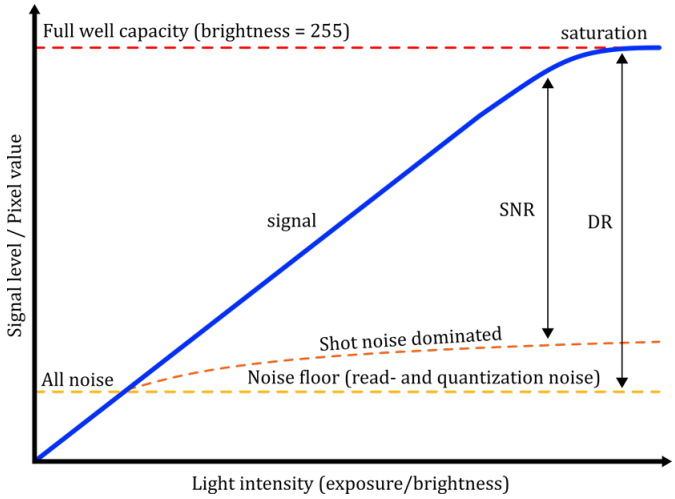


Figure 2.11.: SNR and intensity [21]

2.3.4. Depth of focus

Depth of focus describes the distance between the nearest (d_{near}) and farthest (d_{far}) object in an image that has an acceptable sharpness i.e., is in focus. This is commonly defined by the Circle of Confusion (CoC), which is the area of a point in the image sensor that is covered by the light that has passed trough the camera lens. The general definition for an object to be out of focus is when the CoC cover more than one pixels, as illustrated in Figure 2.12. The first scenario demonstrates this, for the closest and farthest object the focus is in front and behind the senor plane respectively. The effect of this is that the target pixel loses signal power as noise to the neighboring pixels [25], [26].

2.3.5. Exposure Time

Exposure time is the amount of time that an image is exposed to light, also know as shutter speed, i.e., the amount of time the shutter remains open. It has a significant effect on quality of an image, too much light results in an overly pale image, know as over-exposure while too little light results in a overly dark image, know as under-exposure. Another effect of shutter speed is called motion blur, a long exposure time will result in a blurry image while a short exposure time will "freeze" moving objects in an image [27].

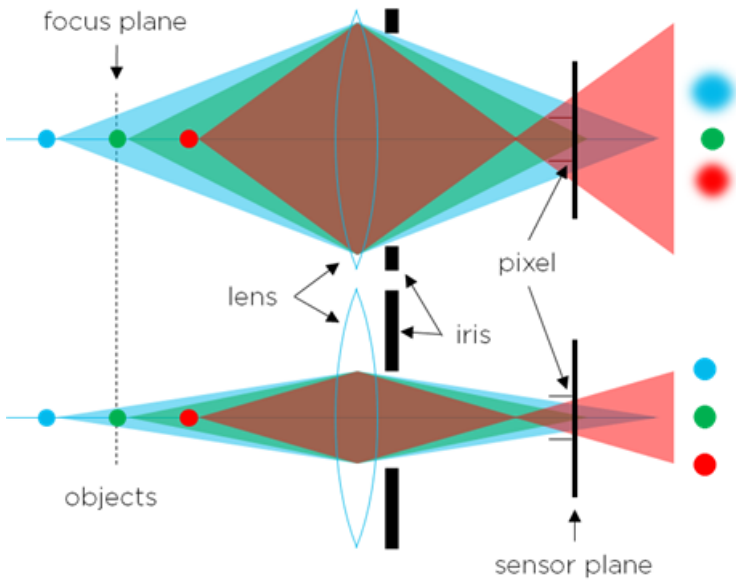


Figure 2.12.: Circle of confusion from [26]

2.3.6. Aperture

The size of the lens opening is called aperture, it can be fixed or adjustable depending on the camera. Aperture is described by the f-number N, given by the following equation.

$$N = \frac{f}{D} \tag{2.17}$$

where f = focal length and D =the diameter of the entrance pupil [28].

2.3.7. High Dynamic Range (HDR)

Dynamic range in an image describes the ratio between max and min tonal values, i.e., the brightness in an image between shaded areas that appears as pure black and brighter areas that appears as pure white. In Figure 2.9 the dynamic range is the low-high SNR region, i.e., where the sensor can distinguish intensity values. HDR images are created by combining several different narrow range exposures of the same object to increase the dynamic range [29].

2.3.8. Exposure stops aka stops

Exposure stops, f-stops or simply Stops in photography is used to describe the brightness of image, i.e., how much light hits the sensor, relative to specified

Figure 2.13.: 2 stops exposure difference between left and right side [30]

reference level (commonly called 0 stops). Moving up on stops is equivalent to doubling the brightness in an image, moving down on stop is equivalent to reduce the brightness by a half, in Figure 2.13 the difference between 2 stops are shown [30]. Modern lenses commonly uses a standard f-stop scale which is a approximation of an geometrical sequence. Each stop represents halving the light intensity which corresponds to reducing the entrance pupil by a factor of $1/\sqrt{2}$ [31].

2.3.9. Gain (ISO)

Gain in photography refers to increasing the brightness in image at a certain exposure. This is done by increasing the pre-amplification of the image sensors which increases its sensitivity to light. The amount of gain used is commonly given as ISO values, e.g. ISO 400, ISO 800 etc. a higher ISO values equals a greater brightness in the image [32]–[34].

2.4. Error effects in images

This section describes errors in images and how they can affect a point cloud.

2.4.1. Ambient light sources

Ambient light sources can be source of issues when working with structured light technology, the frequency of the light can interfere with the light sampling done

by the camera. The effect can demonstrated by filming a 50 Hz light source with a 30 fps camera, the light will appear to be flickering.

2.4.2. Blooming

Blooming occurs when the image sensor is over saturated after been hit with an extremely intense light from a point or a region. When this occurs the pixel gets blinded, i.e., becomes completely white and depth cannot be accurately calculated, and because the light is so intense from the pixels it spills into their neighboring pixels. These effects can be seen in Figure 2.14 where some regions of ocean becomes completely white and it is impossible to distinguish surface features. In Figure 2.15 the light is spilled into neighboring pixels causing the stem of the flower to turn orange. This can also be the case when working with specular objects and active stereo vision camera. The projector light reflection from the object can become so bright that the pixel is oversaturated, this is refered to as a highlight [35].

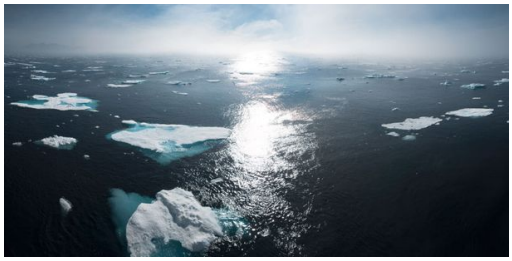


Figure 2.14.: Pixel blinded [35]

Figure 2.15.: Light spill [35]

2.4.3. Reflection points

Reflection points in point clouds are erroneous point regions floating in mid-air, typically seen as “ghost planes” that stretch towards or away from the camera. The cause of these points could be interreflections, excessive movements or alien light sources [36].

Contrast distortion

Contrast distortion in point clouds can happen when there is an abrupt intensity change in a 2D image, such as in checkerboards or on specular surfaces such as round shiny metal objects [37]. This effect occurs in the x-axis of the image sensor, so parts that has their overexposed regions perpendicular to the camera baseline are especially affected by this effect, this is illustrated in figure Figure 2.17. It appears as a ripple-like effect or wave in the point cloud, shown in Figure 2.16.

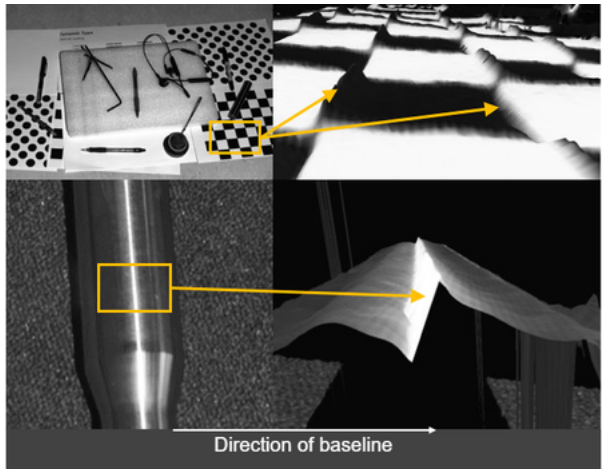


Figure 2.16.: Examples of contrast distortion [37]

Figure 2.17.: Effects of constrast distortion depending on object orientation [38]

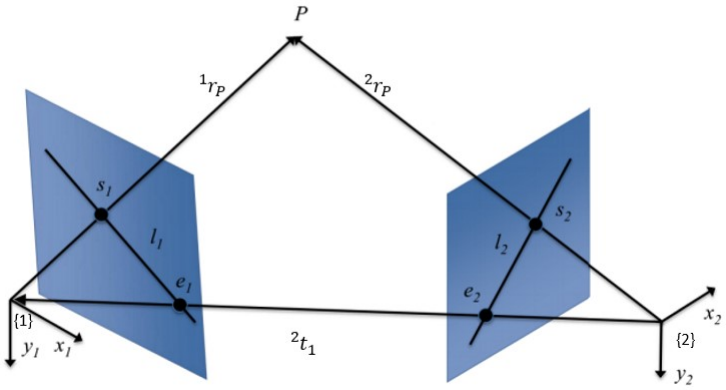


Figure 2.18.: Stereo arrangement [20]

2.5. 3D vision

This section describes the concept of stereo vision, triangulation, the structured light imaging technique used in Zivids cameras and point clouds.

2.5.1. Stereo vision

Stereo vision for computer is based on human vision and is used to capture 3D images of a target scene. It can be divided into the passive and active stereo vision. Passive is when two cameras are used and active is when one of the cameras is replaced with some sort of a light source, i.e., a projector or laser. This section gives a description on how stereo vision can be used to generate 3D images based on a passive vision system.

The setup of a stereo arrangement can be seen in Figure 2.18. The frame of camera 1 and 2 are given as $\{1\}$ and $\{2\}$ respectively. The point $P \in \mathbb{R}^3$ is given in $\{1\}$ and $\{2\}$ as shown in (2.18) and the position of $\{1\}$ relative to $\{2\}$ is given as $^2t_1$. The transformation from $\{2\}$ to $\{1\}$ becomes as shown in (2.19) and finally the point P in camera are related as shown in (2.20) where $\tilde{r}$ are homogeneous vectors.

$$\begin{aligned} {}^1r_p &= [x_1, y_1, z_1] \\ {}^2r_p &= [x_2, y_2, z_2] \end{aligned} \tag{2.18}$$

$${}^2H_1 = \begin{bmatrix} {}^2R_1 & {}^2t_1 \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.19}$$

$${}^2\tilde{r}_p = {}^2T_1 \, {}^1r_p \tag{2.20}$$

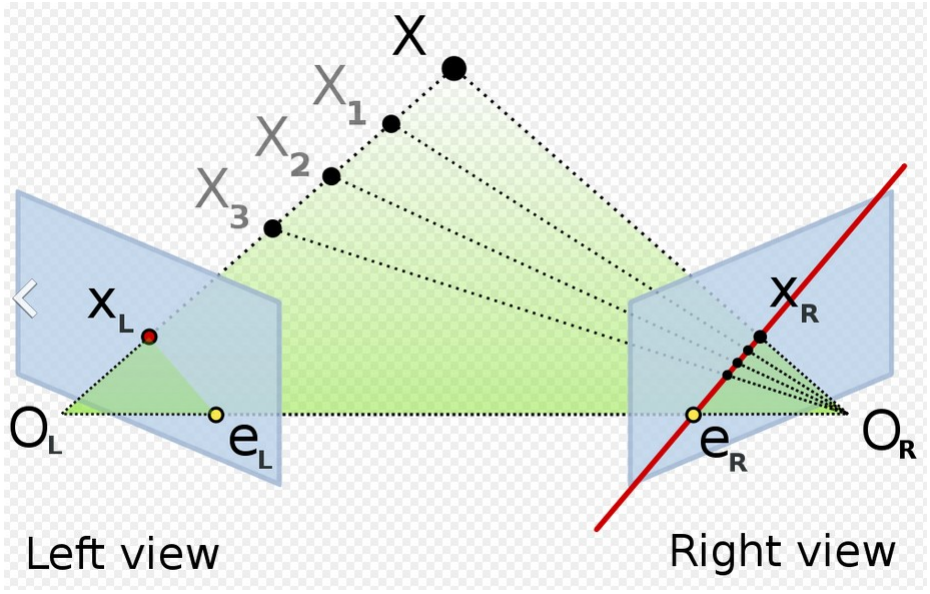


Figure 2.19.: Epipolar lines [39]

The image coordinates s_1 and s_2 follow the same model as shown in (2.13) (2.15). l_1 and l_2 are the epipolar lines defined in (2.21).

$$\begin{aligned} l_1 &= Es_1 \\ l_2 &= E^T s_2 \end{aligned} \tag{2.21}$$

Where E is the essential matrix $E = ({}^2t_1)^{\times}\,{}^2R_1$ used in the epipolar constraint expression in the coordinates of $\{2\}$ shown in (2.22). The constraint follows from the observation that since the vector ${}^1r_p$, ${}^2r_p$ and ${}^21_p$ lies in the same plane the triple scale product of them are zero. The epipolar line and some point $X_i, i = 1, ..., n$ are shown in Figure 2.19. This illustrates how a point X in one image can be seen as a line from the optical center of the other camera to the point X in the other image, this line is the epipolar line [39].

$$({}^2r_p)^T E\,{}^1r_p = 0 \tag{2.22}$$

e_1 and e_2 are the epipoles, which are the corresponding normalized image point when the point to be imaged lies in the origin of one of the camera frames, i.e.,

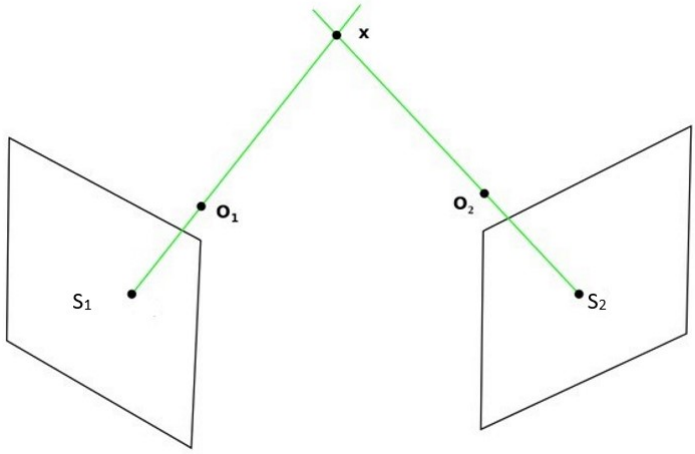


Figure 2.20.: Triangulation ideal case, [41]

${}^2r_p = {}^2t_1$ or ${}^1r_p = -{}^1R_2\,{}^2t_1$. The epipoles are defined by (2.23)

$$\begin{aligned} \lambda_1 e_1 &= -({}^2R_1)^T\,{}^2t_1 \\ \lambda_2 e_2 &= {}^2t_1 \end{aligned} \tag{2.23}$$

2.5.2. Triangulation

Triangulation is process of determining the position of a point in 3D space in CV when we know the position of the point in the two images, i.e., s_1 and s_2, and the pose of the cameras relative to each other i.e., ${}^2T_1$. Using the pinhole camera model and epipolar geometry of a stereo vision setup, illustrated in Figure 2.20, as a demonstration of an ideal case. It can be seen that the projection line, i.e., the line from the point to the image plane through the optical center **O**, for the two cameras intersect at the point x, the triangulation problem is to determine this point. This is straightforward with linear algebra in the ideal case .

In practice however, these points cannot be exactly measured due to noise. So we then get the measured image points s_1' and s_2' shown in Figure 2.21. With these new point the lines may no longer intersect at x or they may not intersect at all, i.e., they do not satisfy the epipolar constraints. The triangulation problems becomes finding the best estimation x_{est} of x. There are several methods to solve this problem, such as Triangulation using the midpoint, Linear triangulation by minimizing the algebraic error and Triangulation using the reprojection as described by [20], [40], [41].

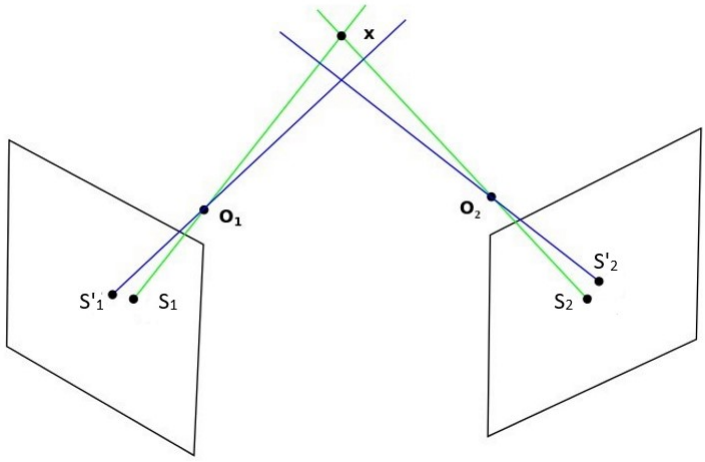


Figure 2.21.: Actual triangulation case [41]

2.5.3. Structured light

Structured light imaging is a technique of using an active stereo vision system for mapping the topography of a surface. As with a passive stereovision, the goal is to find the 3D point of intersection for the optical lines. The basic principal for this can be explained by using the projector to light up a single point in the scene and capturing this point with camera. Then we know the image coordinates for the point in the camera and projector and can use triangulation to calculate the depth of the point. In practice, doing it point wise for the entire scene would be very time consuming. The technique of using structured light is based on this principal but instead of using a point, light patterns is projected onto the scene.

There are many variants of these patterns, from simple vertical lines to more complex grid structures. To explain the principle, a binary code pattern is used, i.e., black and white horizontal bars. The reason for using vertical lines and not horizontal is because the pixel position of horizontal lines does not change if the target is moved in the scene. In the camera the horizontal lines from the projector correspond to the epipolar line, which, as show in Figure 2.19, when the point X or the target in this case is moved it remains on the line. From this we can assume the horizontal line do not give any depth information any vertical lines are used, i.e., we are only interested in the x-coordinate of the projector pixel, the projector pixel y-coordinate information is contained in the y-coordinate of the camera pixel.

We can now project a vertical plane from the projector for a given x-coordinate instead of the beforementioned point and find the intersection between the pro-

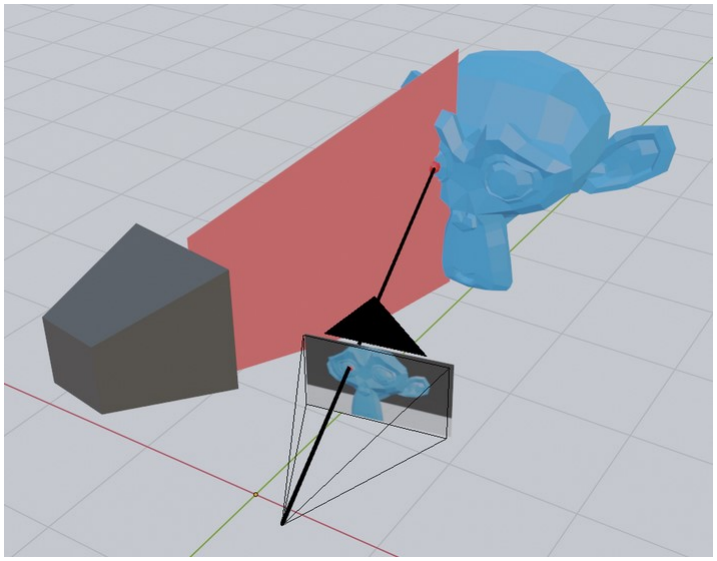


Figure 2.22.: Intersection line and plane [43]

jection line from this camera and the plane as illustrated in Figure 2.22. To determine the projector x-coordinate of the pixel in the camera image we use the color of the pixel. By projecting a binary pattern with different width, as shown in Figure 2.23, onto the scene and taking multiple images we can see which part of the scene that is hit with light from witch part of the projector, e.g. if we have a 50/50 split like the first pattern in Figure 2.23 we would see that the part of the scene with light comes from the right half of the projector. We can then calculate the binary code for a pixel by taking the average of max and min values of the pixels across all images and comparing it with the pixel value of a give image, if the value is greater than the average found we set this pixel the value one. The resulting pixel value is given as grey value. The result of this can be shown in Figure 2.24. Then the 3D coordinate can be calculated as the intersection of the plane with the given projector x-coordinate value and the projection line from the camera [42]–[45].

2.5.4. Point cloud

A point cloud is a data structure used to represent a set of n points in D dimensions, commonly as xyz-coordinates in 3D. The dimension increase with additional information e.g. with color added it becomes 4D. Point clouds are produced by lidars, stereo camera, tof (time-of-flight) camera, 3D cameras or can be generated synthetically from software such as CAD models [46].

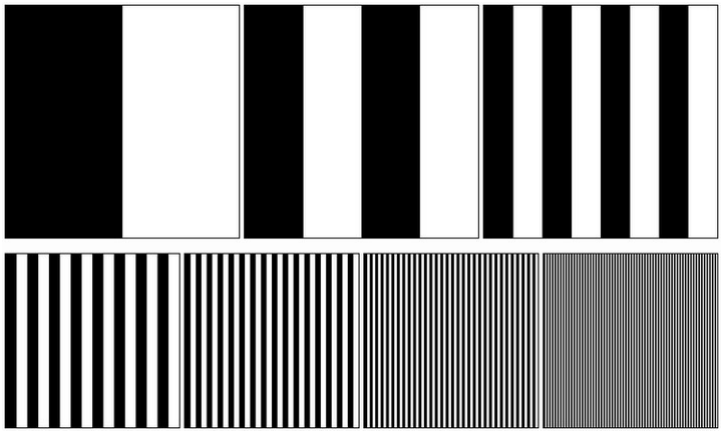


Figure 2.23.: Binary patterns, structured light [43]

Figure 2.24.: Projector x-coordinate [43]

Point clouds can be represented in hundreds of different file types, which can cause issues when working with incompatible software and hardware. The biggest difference is the use of binary and ASCII (American Standard Code for Information Interchange). ASCII is rooted in binary but conveys information using text, common ASCII formats are XYZ, OBJ, PTX and ASC. The advantage of ASCII is the universal accessibility provided by the standardized text abstraction used to convey data, this makes them good for storing data long term. The disadvantage is that the files are larger then binary, they must be read line by line and the information it can store are limited, usually just xyz-coordinates but there are some formats that can store additional information.. Binary files stores the data directly in binary code, common point cloud formats are FLS, PCD and LAS. These files are more compact and can contain more information but there are greater restrictions on how they can be accessed. There are files that are capable binary and ASCII formats, some of the most common are PLY, FBX and E57 [47]. Common point cloud files are presented in the list bellow.

- OBJ: Only represents 3D geometry, normals, color and texture (ASCII)
- PLY: aka. polygon file format or Stanford triangle format, represents objects as nominally flat polygons. Capable of representing color, transparency, surface normals, texture, coordinates and confidence values (ASCII and binary)
- XYZ: non-standardized set of files based on Cartesian xyz-coordinates, difficult to work with because they are not unit standardized (ASCII)
- E57: Vendor neutral file format, can represent normals, colors and scalar field intensity (ASCII and binary)

2.6. Pose estimation

The section describes the pose estimation method point pairs features used in MvTec HALCON.

2.6.1. Surface matching using Point pair features

The method of surface matching using point pair features(PPF) was proposed by Drost, Ulrich, Navab, *et al.* [48] in 2010 and has become recognized as the benchmark for new pose estimation approaches. The assumption for the method is that we have a point cloud of the scene and model, with the coordinates of each point and their associated normal known. Points in the scene and the model are represented by $s_i \in S$ and $m_i \in M$ respectively.

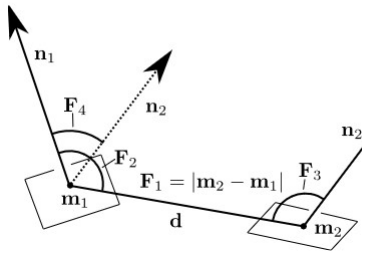


Figure 2.25.: Illustration of PPF from [48]

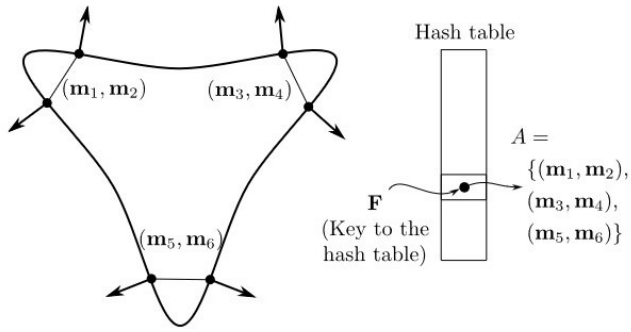


Figure 2.26.: Similar F stored in the same slot in the hash table [48]

Point pair features

PPF describe the relative position and orientation of two oriented points. The feature F between two points m_1, m_2, with normal n_1, n_2 and distance between them $d = m_2 - m_1$ is defined as:

$$F(m_1, m_2) = (||d||_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2)), \quad \angle(a, b) \in [0, \pi] \tag{2.24}$$

This feature is illustrated in Figure 2.25

Global model descriptor

In the offline phase a global descriptor is made by calculating F for all point pairs $m_i, m_j \in M$, this descriptor is a mapping from the sampled PPF to the model, written as $L : \mathbb{Z} \to A \subset M^2$. Point pair vectors with similar discrete features, as illustrated in Figure 2.26 are stored together in the same set (A) in a hash table. Then all model features F_m that are similar to the scene features F_s can be searched for in the hash table using F_s as a key.

Local coordinates

If we have a point that lies on the object in the scene $s_r \in S$, then there exists a corresponding point $m_r \in M$. By aligning these two points and their normal, the model can be aligned with scene by rotating it around the normal of s_r. Thus, we can describe the rigid body motion from the model space to the scene space by a point m_r and a rotation angle α, these are called the local coordinates of the model.

More generally, a point pair in the model $(m_r, m_i) \in M^2$ and a scene pair $(s_r, s_i) \in$

S^2 with similar feature vector can be aligned with following transformation.

$$s_i = {}^{s}T_g^{-1} R_x(\alpha)\, {}^{m}T_g m_i \tag{2.25}$$

This is illustrated in Figure 2.27.

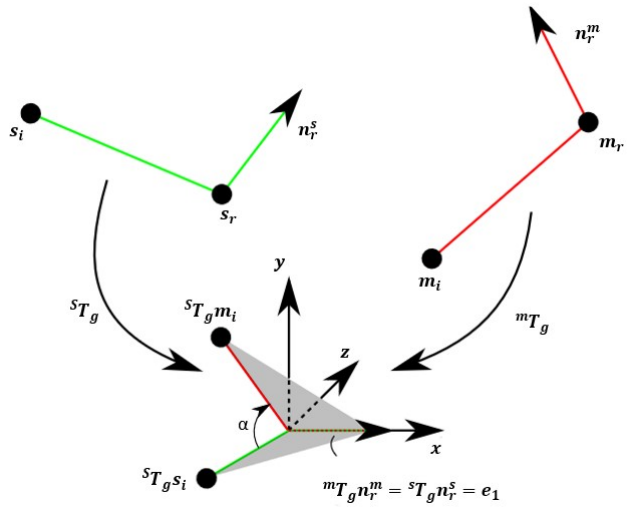


Figure 2.27.: Transformation from model to scene [48]

Voting scheme

A voting scheme is used to find the optimal local coordinate given a fixed reference point s_r in the scene, this is done to maximize the number of in the scene that lies on the model. The voting scheme used is similar to the Generalized Hough Transform, it is a two-dimensional accumulator array. With number of rows equal number of sampling points m and number columns equal to number of sampling steps for α. The process pairs the reference point s_r with every other point $s_i \in S$, calculates the feature $F_s(s_r, s_i)$ which it uses as a key in hash table to find matching model points m_r, m_i. Then (2.25) is used to find α and a vote is cast for the local coordinates, this is illustrated in Figure 2.28. The peak in the accumulator array is the optimal local coordinates, that we can use the calculate the rigid movement.

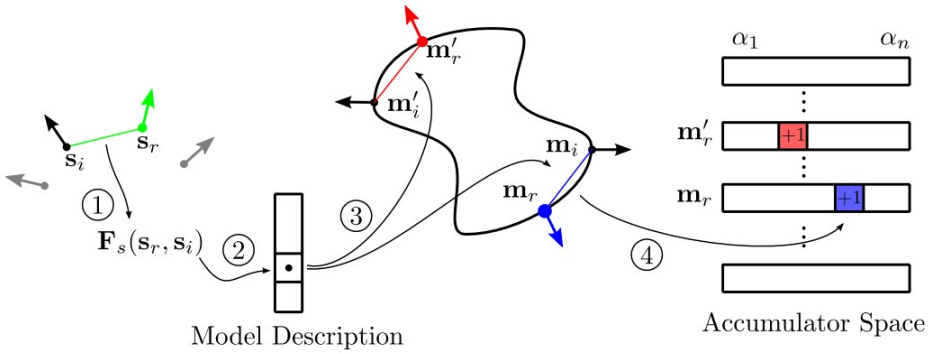


Figure 2.28.: Model descriptor and accumulator [48]

Pose clustering

Multiple fixed reference points s_r are necessary to ensure that one of the points lies on the surface of the model so that voting scheme identifies the correct pose. Because of difference in sampling rate of the model and scene the pose identified is only an approximation of the ground truth. To optimize the result, the identified poses are clustered together, i.e., if the found pose difference is bellow a certain threshold they clustered. Then the score of each cluster is calculated using the combined score of each pose in the voting scheme. The pose of the cluster with the maximum score is averaged and returned as the final pose.

2.7. Bin picking

Bin picking is a CV and robotics problem where the goal is to determine the pose of based on what the vision system sees and use this pose to get the robot to pick up the object. In general the bin picking problem can be divided into three categories: Structured bin picking, the objects are organized into structured patterns making them easy to identify and pick, semi-structured bin picking, the parts are structured with a mild amount of organization to aid the picking. Both tasks can be automated using 2D vision. The last category is random or unstructured bin picking, each object have a random pose and they can be overlapping, this problem requires 3D vision cameras or advanced machine learning algorithms to automate [49]. The problem faced in this project is that random bin picking [50].

2.7.1. Robot and camera coordinate system

A bin picking system typically consist of a robot, camera (sensor), and the object to be picked. They are represented mathematically with coordinate systems. The

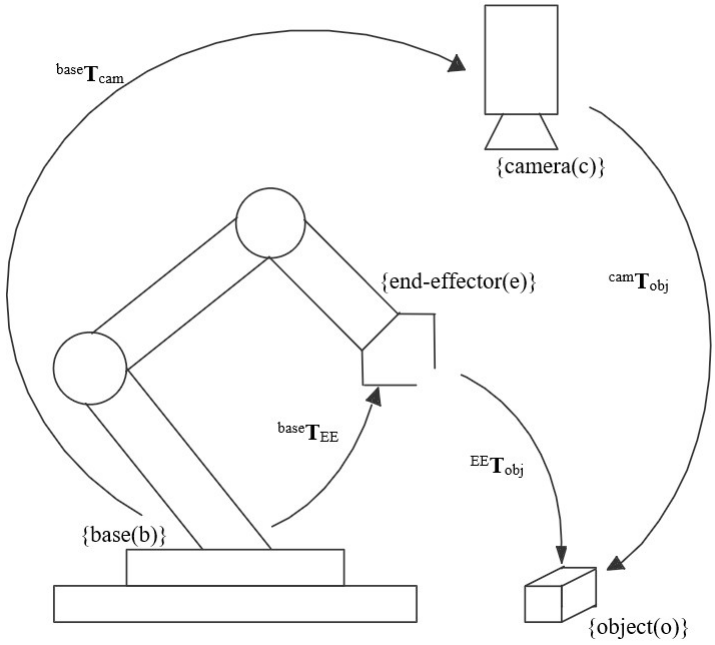


Figure 2.29.: Overview of a bin picking system

robot arm has two main coordinate systems, the robot base coordinate system {b} and the end-effector coordinate system {e}. The camera and each individual object have a single coordinate system, given as {c} and {obj} respectively. This is shown in Figure 2.29.

In the bin-picking system the object in the camera frame represented by ${}^{cam}T_{obj}$, the camera in the robot base frame is represented by ${}^{base}T_{cam}$ the object in the robot base frame is represented by ${}^{base}T_{obj}$ and the end-effector in the robot base is represented by ${}^{base}T_{EE}$.

2.7.2. Object coordinate system and robot target

When the system have found the object in the scene and calculated ${}^{base}T_{obj}$ the next step is to transform ${}^{base}T_{obj}$ into a robot target, i.e., into a pose the robot end-effector can be moved to. The pose estimation does not consider if the if the object pose is reachable for the robot, it only returns the pose of the object coordinate system. There are therefore two key areas that need to be resolved when setting up a bin-picking system. The first is, how is the coordinate system is defined for the object and how does it need to be transformed so that the robot can reach it. The most obvious case to show the importance if this is when the

returned pose of the object has it z-axis pointing directly upwards, that is in the positive z direction of the base. Noting, that we define the end-effector z axis as pointing directly outwards from the hand, as can be seen in Figure 2.30, this would result in the robot having to pick from bellow the bin, called “bottom-up” picking, what we want is “top-down” picking, this is illustrated in Figure 2.31. The required transformations can be calculated using mathematical rules that first check the pose and transformation based on set operations or a AI can be used.

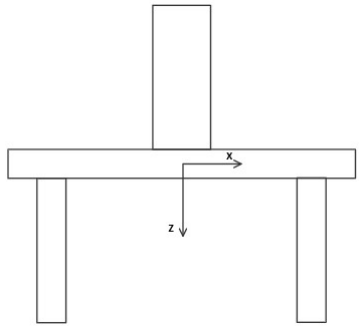


Figure 2.30.: Gripper coordinate system

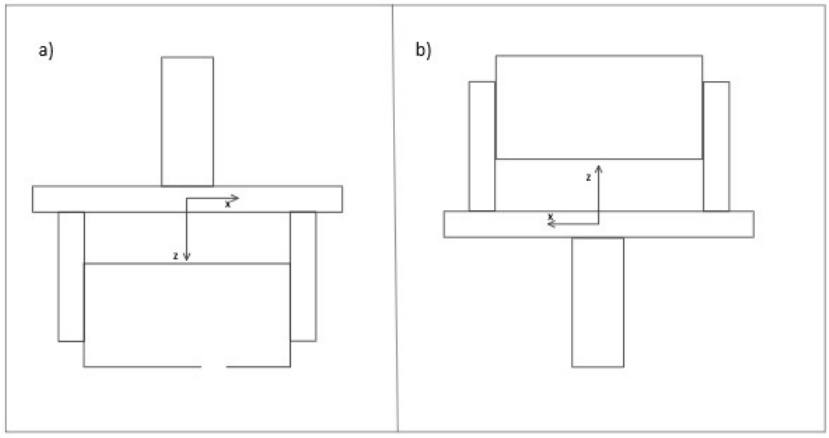


Figure 2.31.: Picking a) "top-down", b) "bottom-up"

2.7.3. Hand eye calibration

Hand-eye calibration is the process of relating what the camera(eye) sees to the movements of the robot arm(hand), i.e., they collaborate in the same coordinate system. There are two types of calibration depending on how the camera is

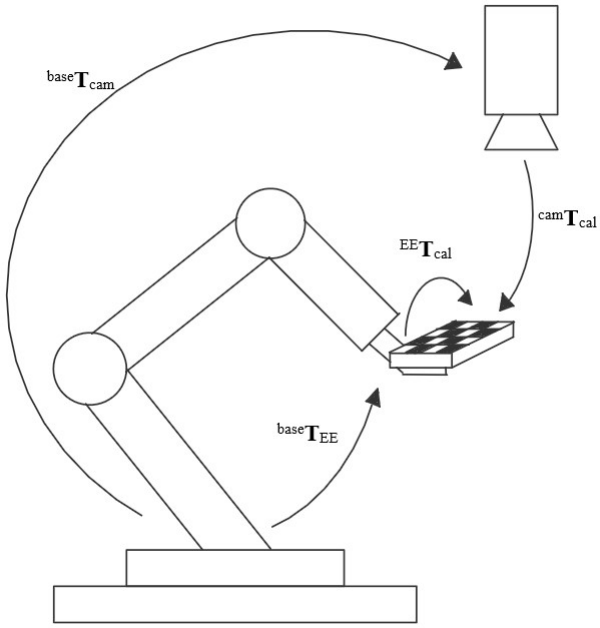


Figure 2.32.: Eye-to-hand calibration

mounted. If the camera is mounted in a fixed position next to the robot the transformation from the camera coordinate system to the robot base is found, this is known as eye-to-hand calibration. If the camera is mounted on the robot arm, the transformation from the robot end-effector to the camera is found, this is called eye-in-hand calibration. For this project, the camera will be mounted above the robot looking down on the task space, therefore the eye-to-hand calibration is explained further [51].

To do the calibration we first need an object of know geometry that can be detected and localized in the camera, known as a calibration object. Common calibration objects for 3D cameras are 2D patterns such as checkerboards and dots, and 3D objects such as spheres. ${}^{EE}T_{cal}$ and ${}^{cam}T_{EE}$ is the calibration object expressed in the end-effector frame and camera frame respectively.

Eye to hand calibration

A common formulation for the eye-to-hand calibration problem is as follows:

$$ {}^{EE}T_{cal} = {}^{EE}T_{base}\,{}^{base}T_{cam}\,{}^{cam}T_{cal} \tag{2.26} $$

It is illustrated with a checkerboard in Figure 2.32

Where the goal is to find ${}^{base}T_{cam}$, to solve this the calibration object is fixed onto the robot end-effector, in such a way that the transformation ${}^{EE}T_{cal}$ stays constant. Then n images are taken while changing the pose for each one, for each pair of these calibration poses we use the following representation:

$$Y = A_i X B_i \tag{2.27}$$

where

$$A_i = {}^{EE}T_{base}, B_i = {}^{cam}T_{cal} \quad i = 1, 2, ..., n \quad , X = {}^{base}T_{cam}, Y = {}^{EE}T_{cal}$$

Since Y is constant we can eliminate it from the equation by setting Y of pose state i equal to the Y of pose state j, we then get $A_i X B_i = A_j X B_j$. We can rearrange this into:

$$AX = BX \quad , \quad A = A_j^{-1} A_i, \quad B = B_j B_i^{-1} \tag{2.28}$$

Where A and B represents the motion between pose i and pose j of the end-effector and calibration object respectively. Then the calibration problems boils down to solving this equation for X [52].

Chapter 3.

Tools

This chapter is divided into three section describing Zivids camera and software, MvTec HALCON machine vision software and ABBs YuMi robot and robot software RobotStudio. The majority of the content is from the specialization project delivery in the course TPK4560 [4].

3.1. Zivid

Zivid is a provider of industrial 3D vision cameras and machine vision software. This project uses their zivid one 3D camera [53] as well as their user interface (UI) zivid studio [54]. They provide extensive resources through their software development kit (SDK) [55], zivid knowledge base (ZKB) [56] and their blog [57].

3.1.1. Zivid One 3D camera

The Zivid One camera is a rugged and high-quality 3D camera with rgb colors, it is an active stereo vision system that uses the structured light technique to capture images. It is developed for industrial applications such as robot bin picking, inspection of manufactured parts and machine vision. With its relative light weight and compact size it can be mounted on the robot arm or be fixed with a top down view. It can acquire images at a rate of 10hz with a quality of 2.3 Mpixels and a depth resolution of 0.1mm@0.6m. The field-of-view (FOV) of the camera is shown in Figure 3.1 and the values are given in Table 3.1. The highlighted specifications are given in Table 3.2 [53].

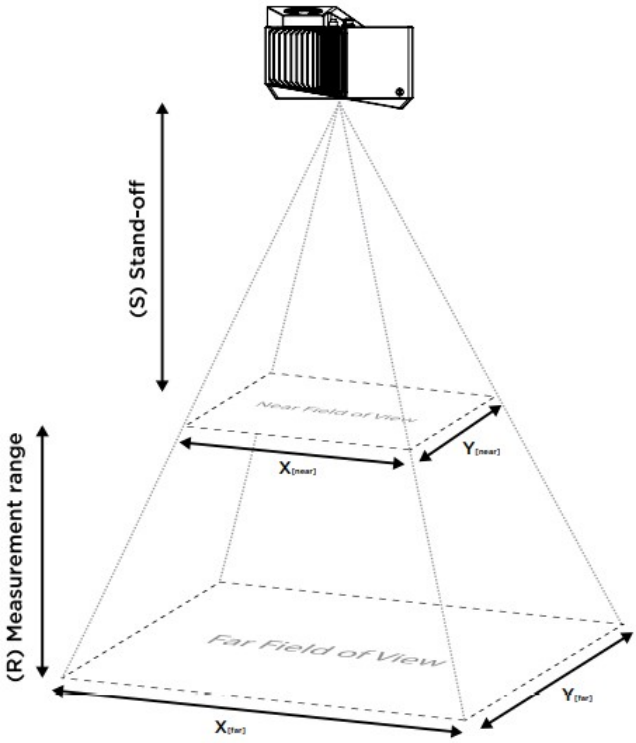


Figure 3.1.: Field-of-view Zivid One [53]

Zivid	**Distances (mm)**
(S) Stand-off distance	600
(R) Measurment range	500
(S+R) Maximum distance	1100
Near FOV image area($X_{[near]} \times Y_{[near]}$)	430 × 270
Far FOV image area($X_{[far]} \times Y_{[far]}$)	780 × 490

Table 3.1.: Field-of-view Zivid One [53]

3.1.2. Zivid Studio

Zivid Studio is a cross platform (linux and windows) UI where we can easily adjust acquisition setting for the camera and evaluate the quality of the captured point cloud. The main features are focused on acquisition, we can choose between assisted mode and manual mode. With assisted mode all we need to specify is the capture time, then the software adjust the other settings. The main settings and

Zivid	**Specifications**
Output	2.3 Mpixels 3D RGB image
Acquisition rate	≥10 Hz, 100 ms snapshot
Optimal Working Distance	0.6 m - 1.1 m
Depth Resolution	0.1 mm @ 0.6 m
Software APIs	C++, C#, .NET, Python, MATLAB
OS	Windows 7/8/10
Housing	Rugged aluminum Water & dust resistant
Dimensions	226 x 165 x 86 mm
Weight	2kg

Table 3.2.: Technical data Zivid One [53]

Settings	**Filters**
Exposure time	Noise removal
Aperture Step/ F-number	Outlier removal
Brightness	Reflection Removal
Gain	Smoothing Gaussian
Color balance	Contrast Distortion
Ambient light adaptation	

Table 3.3.: Setting and filters, Zivid Studio

filters we can adjust are listed in Table 3.3, they are explained in detail in **??**. To evaluate the image, we can view the point cloud, a dept map and the 2D color image, it also features a histogram to analyze the pixel intensity distribution in the image. From Zivid studio we can export the captured point cloud to common file formats such as ordered and unordered PLY, ASCII (XYZ) and point cloud data (PCD) file format [54], [58].

Evaluating the point cloud

The histogram is a tool to evaluate the point clouds, it records the intensity value of the pixels and groups them in bins from 0 to 255. The values can be displayed with a logarithmic x-axis, then each bin represents a stop. This is useful because it shows the exposure values needed to expose certain regions of the image. How an histogram is read is shown in Figure 3.2 and the use of the logarithmic stops are described in **??** [59].

In the point cloud viewer, the evaluation is done by a visual inspection. In the 2D image the RGB values and the SNR can be read from each pixel. The dept shows if there are missing points or depth related errors in the point cloud.

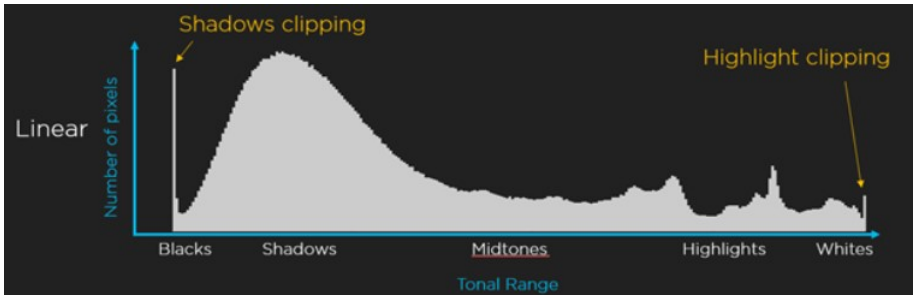


Figure 3.2.: Explanation of the histogram in zivid studio [59].

3.1.3. Vision engine

Zivid vision engine control the pattern projecting, imaging and processing to generate the 3D point cloud. Zivid has two different engines, the phase engine is the default and is recommended for general scene with object that are diffuse and moderately specular. This makes it suitable for piece picking and logistics, they call it good compromise between quality and speed. The stripe engine is built for exceptional point cloud quality in scene with very specular reflective objects like metallic cylinders. This engine is less sensitive to highlights, blooming, and corrupted data due to direct reflections from the projector and ambient light sources. It is recommended to use in bin picking or with scenes containing specular objects. This improvement in point cloud quality comes with an increased capture speed because of an higher dynamic range compared to the phase engine [60].

3.2. MvTec HALCON

HALCON is comprehensive machine vison software that comes equipped with more than 2100 operators that, among others, can solve object detection and pose estimation problems using various machine learning techniques. It comes equipped with an integrated development environment (IDE) called HDevelop.

HALCON provides a lot of relevant features for this project, it has a pretrained convolutional neural network (CNN) that can perform object detection and segmentation with pixel precision. It can perform various 3D recognition tasks such as shape or surface matching for pose estimation, this can be trained with CAD models of the object or by using point cloud images from 3D cameras. It can find objects with an accuracy of 1/20 pixels in 2D and 3D scenes and use this to evaluate placement or distinguish between touching objects. It can perform bot intrinsic and extrinsic 3D calibration. It can read bar codes and other data codes. MvTec offers free support and access to training, there is extensive documentation

on the operators and the product can run on Windows(64-bit), Linux(64-bit) and macOS [61].

3.2.1. HDevelop IDE

HDevelop is HALCONs programming environment, the graphical user interface (GUI) is designed for usability and comes with useful features for image processing. The program supports C, C++, Python and .Net languages like C# and VB.NET. Included in the download is a example programs that cover many of the applications HALCON is developed for. Theses examples and the text editor that checks syntax, debugs and give recommendation makes it possible to rapidly develop prototypes and test different functionality. In the environment, images and 3D scenes can be inspected in real-time. It comes with software that support GigE Vision, GenICamTL and USB3 Vision to interface with image acquisition devices and a software interface for digital I/O devices. The developed code can be password protected, so that the functionality can be shared without revealing the code [61], [62].

3.3. ABB

3.3.1. ABB YuMi - IRB 14000

ABBs YuMi®- IRB 14000 is a collaboration robot designed to meet the requirements of flexibility and agility in small parts assembly. Because of its small size, protective padding, impact sensors and low torque it does not require a closed of cell. In fact, YuMi is designed to work side-by-side or face-to face with humans and has safety features collision detection. Its light weight, partly due to its magnesium skeleton, makes it easy to relocate YuMi when the production line changes, or other task requires more resources. YuMi has two arms that are mirrored about the center, each arm has 7 revolute joints giving it 7 degrees of freedom.

Precision and speed are key specs in assembly task, the YuMi has a max tool center point (TCP) velocity of 1500mm/sec and it can return to the same point with an accuracy of 0.02mm. At full stretch it can reach object 559 mm away and can carry a payload with weight up to 0.5kg. At the end of the arms it is equipped with smart servo claw grippers with a travel length of 50mm and a max gripping force of 20N. These grippers can additionally have a camera in the palm and up to two vacuum suction cups on the sides of the wrists, the different configuration are shown in Figure 3.3. On the YuMi robot used for this project it has one standard claw gripper and one with the vacuum suction cup.A full data sheet for YuMi by ABB can be found in appendix A [63]–[66].

Figure 3.3.: YuMi Smart gripper configuration

The YuMi has an embedded controller based on the 5th Generation ABB Industrial Robot Controller IRC5 that features ABB's state-of-the-art (SOTA) motion control. The IRC5 optimizes the robot performance by reducing cycle times and providing precise path accuracy, coined QuickMove ® and TrueMove® respectively by ABB, based on advanced modeling. This makes the robot predictable and it does not require any tuning. Another feature of the IRC5 is the FlexPendant, shown in Figure 3.4, a handheld operator unit with a touchscreen, joystick some basic function buttons and an emergency stop. It is used for many of the operating tasks such as jogging manipulators, running and modifying programs, and calibrating the robot [67]–[69].

Figure 3.4.: ABB FlexPendant

Coordinate system of the robot

The different coordinate systems for the robot is shown in Figure 3.5. They are color coded with the x-axis in red, the y-axis in green and z-axis in blue. For the robot base the x-axis is defined as pointing outwards from the robot, i.e., into the workspace, the y-axis points transverse to the robot and the z-axis points vertically upwards. For the gripper the x-axis is the direction that the claws open

and close, the y-axis are perpendicular to the claws movement and the z-axis points away from the end-effector.

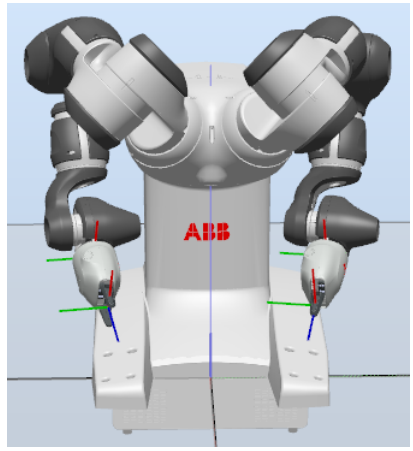


Figure 3.5.: ABB YuMi® - IRB 14000 Coordinate system

3.3.2. ABB RobotStudio

Robot Studio is ABB simulation and offline programming software, it makes it possible to setup the entire production line in a virtual environment before building it. This digital twin can be shared with all relevant partners to give updates, ensure that the customers gets what they expect and modify it if needed. When the production line is set up any new modifications or alterations can first be incorporated in the digital twin so that it does not disturb the production. All programming done is built on the ABBs VirtualController which is an exact copy of the software that runs on the robots. Highlighted features are: TrueMove™, a path visualization and optimization tool, SmartComponents, a tool that can add real world behavior to CAD objects, program editing and debugging, SafeMove a tool that autogenerates the safety sones around the robot, the virtual program can easily be transferred to the real robot, RobotStudio forum, an online community for developers and the production line can be visualized on the shop floor with augmented reality (AR). An example of a production line in RobotStudio is shown in Figure 3.6. To programming of ABB robots uses RAPID, a high-level programming language developed by ABB for their robots [70], [71].

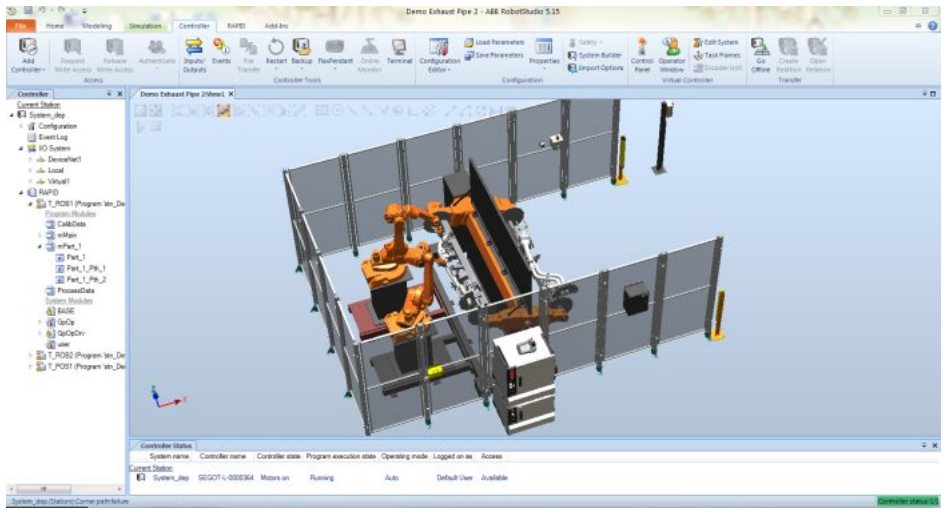


Figure 3.6.: ABB RobotStudio

Chapter 4.

Method

This chapter describes the settings and filters used when capturing point clouds with Zivids cameras. How to use MvTec HALCON for pose estimation and how to program the ABB YuMi robot. The majority of the contents is from the specialization project delivery in the course TPK4560 [4].

4.1. Image acquisition

This section describes the image acquisition settings and filters, condition for good quality and image evaluation.

4.1.1. Acquisition Settings

Correlated sampling

Correlated sampling is the concept of synchronizing the FPS of the camera with the frequency of a light source to make it appear constant. It can be used to filter out time variant noise from ambient lighting when using structured light cameras. To apply this filtering the exposure time t_{exp} is chosen to satisfy the following equation provided by the Zivid Knowledge Base (ZKB) [27].

$$t_{exp} = \frac{n}{2f_s}, n = |Z| \tag{4.1}$$

n is a positive integer and f_s is the frequency of the light source. It is noted that DC powered light sources and LED with a a frequency $\geq$ 1kHz will not cause noise that varies with time.

Aperture effects on image sharpness

The f-number affects the depth of field, a high f-number increases the dept field, i.e., more objects in the image are in focus, as shown in Figure 2.12. Therefore aperture needs be considered when the working distance of the camera is specified. Zivid provides tables with values for f-number, number of stops, (d_{near}),(d_{far}) and Depth of Field top [26]. They also provide a depth of focus calculator to find the recommended aperture based on acceptable blur radius and working distance [72].

Projector brightness

Increasing the brightness results in an increased signal-to-noise ratio (SNR). The signal amplitude of the camera will increase which in turn decreases the impact from noise, given that the pixels are not over saturated. The mean intensity of the image will increases as well, it can therefore be used as stops, see Figure 4.1 [73].

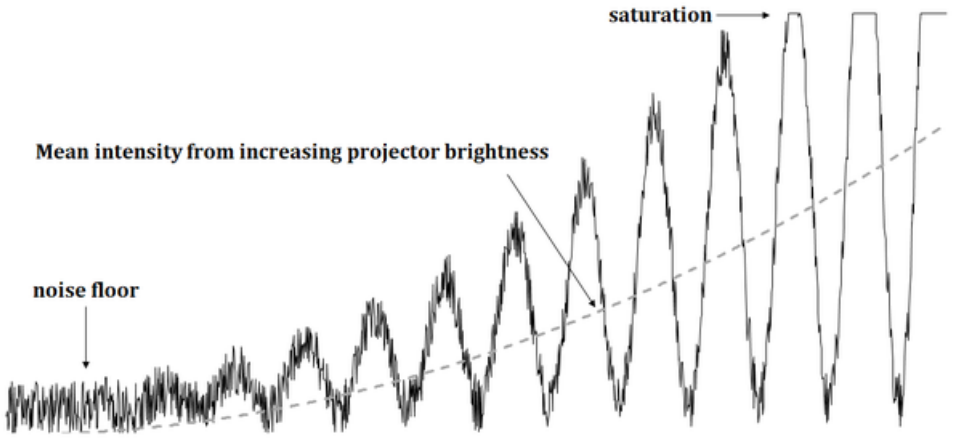


Figure 4.1.: Increase brightness [73]

Using gain

Gain is beneficial when the objective is to capture point cloud in very dark scenes and for increasing the dynamic range in HDR image with minimal time penalty. The drawbacks of gain is that everything is amplified, including noise, which reduces the SNR and the dynamic range as can be seen in Figure 4.2 [34],

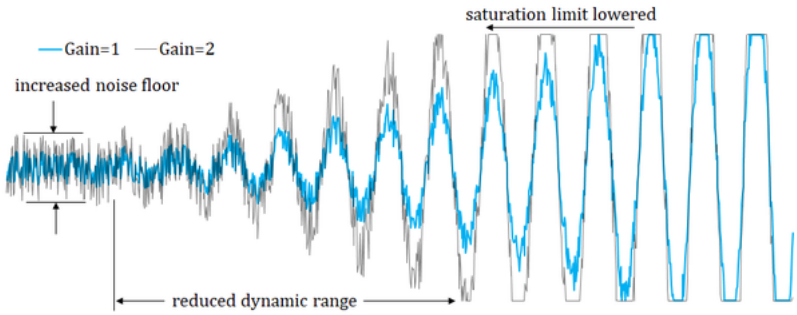


Figure 4.2.: The effect on signals from gain [34]

4.1.2. Filters

The following subsections describe different filters that can be applied with zivid camera, a table of recommended values can be found in ZKB [74].

SNR filter

The SNR filter can be applied to remove points that are bellow the quantifiable limit, shown as SNR<1 in Figure 2.9 and noisier point with a low SNR value [75].

Reflection filter

The filter removes points caused by reflections, it compares the date captured by the camera with what how it expects the point to look like, if they points collected do not make sense the corresponding pixels are discarded [36].

Outlier filter

Outliers are pixels that, within a local region of neighboring pixels, are deemed to far away from their neighbor. The filter removes these pixels from the point cloud based on a set threshold distance [76].

Gaussian smoothing filter

Gaussian smoothing or Gaussian blur is the process of blurring an image with a Gaussian function, i.e., performs averaging on pixels within a small local region. The filter is used to suppress sparse noise, reduce details and align pixels to a grid in image processing. The visual effect is similar to seeing an image trough a translucent screen [77], [78].

Contrast Distortion filter

The filter can correct and/or remove contrast distorted points that are a result of blurring in the camera lens. How much a point is corrected and/or is removed depends a strength parameter threshold defined. Correction is commonly used in bin picking to make the resulting point cloud better resemble the CAD model of the actual object [79].

4.1.3. Conditions for good 3D data on pixels

Zivid has defined two conditions that must be satisfied for the camera to camera to calculate the distance [80].

- Pixel brightness must be within the measurable range of the sensor [0-255], if not it becomes impossible to distinguish whether the value is 275 or 1000.
- The camera must be able to distinguish the difference between the projector being on and off. I. e the SNR values must be large enough for the camera to decode the signal and small enough so that the peak-to-peak intensity of that signal can be captured within the dynamic range of the camera.

4.1.4. Evaluating the image by pixel

By evaluating the RGB and SNR values in Zivid Studio we can adjust the acquisition settings and filters to satisfy these conditions. The pixels should be exposed such that the color value is between 32-255 and the SNR values should be higher then [80].

4.2. Surface matching

4.2.1. HALCON surface matching procedure

HALCONS surface matching procedure can be summarized in the following steps:

1. Access the 3D object model needed to create the surface model
2. Create the surface model
3. Access the 3D object model that represents the search data
4. Find the surface model in the search data

All functionality is described in detail in HALCONs manual [81], this section gives an overview of it works, important considerations and key parameters.

4.2.2. HALCON surface matching operators

HALCON operators are used to access the functionality from the HALCON library. The most important ones for surface matching are the following.

Read_object_model_3d(operator)

Reads a 3D object model from file .e.g. the scene or the CAD model of the object. Can invert the normals of the model when read if the do not match the scene [81].

Create_surface_model(operator)

Creates a surface model from the read object model that has the data structured needed for surface matching. A requirement for this operator is that the 3D object contains point and point normals. RelSamplingDistance from Table 4.1 is specified here. How this model is displayed in the GUI is shown in Figure 4.4.

Find_surface_match(operator)

The three steps of matching with this operator are as described in this section, these are on as default when using the operator

Step 1: Approximate matching: Points are sampled for the scene and then key points are randomly selected. For each key point, with the assumption that it lies on the surface of the object, the optimum pose of the surface model is computed as described in subsection 2.6.1. The pose that has the largest number of sampled scene points on the surface of the object is determined the best, the number of points is given as the score. There are several parameters that can be set, e.g. how many matches to look for and the min score threshold for the pose. The parameters are explained in detail in the manual [81].

Edge supported surface matching: With edge supported matching on, edges are extracted from the 3D scene and sampled as well. The points of these edges, i.e., edge points, are paired with the key points and used to find similar point-edge combinations on the surface model.

Step 2: Sparse pose refinement In this step the pose found in 1 is refined to increase the accuracy of the pose and significance of the score. This is done by minimizing the distance from the sampled scene point to the plane of the closes model point. The plane of the point is defined perpendicular to the normal, this is illustrated in Figure 4.3. Additionally, if the edge support is used, the pose is

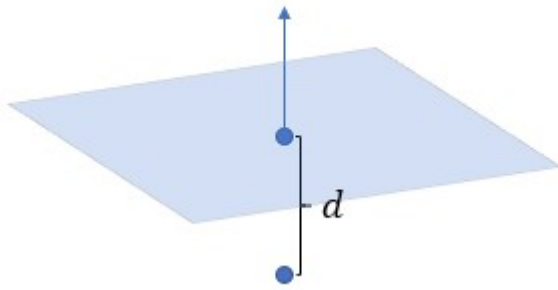


Figure 4.3.: Distance (d) from the scene point to the plane of the closest object point

further optimized by aligning the sampled edge points in the scene with the edges of the surface model.

Step 3: Dense pose refinement The dense pose refinement works in a similar way as sparse, the difference being that it only refines the returned poses specified with the "Num_matces" parameter described in Table 4.1, it uses all the points from the 3D scene and if edge support is on it uses all edge points, not just sampled.

Parameters	Description:
RelSamplingDisctanc	Controls the sampling distance based on the diameter of the axis-parallel bounding box around the object, e.g. object diameter = 10cm, sampling = 0.05, then the points are 5mm parts. Increased = less sampled scene points, decreased = more sampled scene points. Robust matching: 50-100 scene points sampled for each object instance
KeyPointFraction	Controls the number of selected key points from the sampled points, e.g. value = 0.2 then 20% of the sampled points are selected as key points. Increasing key points leads to better match but slower, decreasing has the inverse effect. It is important that each instance of the object is covered by several key points for a stable matching
Num_matches	Sets the maximum number of matches returned, chosen based on highest score
Pose	The output pose is given in the coordinates of the camera frame {c}

Table 4.1.: HALCON parameters

4.2.3. Evaluating the match

The score of the match is an important evaluation criterion of the accuracy. There are different ways to calculate the score of the pose estimation depending on programmer preference and methods used. Bellow is a summary of the different options

With pose refinement:

- Model_point_fraction without edge support
  - The score is the fraction of the model surface visible in the scene, done by counting number of model points that have a corresponding scene point and dividing it by total number of points.
- Model_point_fraction: With edge support
  - The score is the geometric mean of the surface fraction and edge fraction.

  - Num_model_points: The score is the number of sampled model points that where detected in the scene
  - Num_scene_points: The score is the weighted number of sampled scene points that lie on the surface of the object, the points are weighted based on the distance to the found object instance.

Without pose refinement:

- And without edge support
  - Number of point from the subsampled scene that lie on the found object
- With edge support
  - umber of points from the subsampled scene that lie on the found object multiplied with the number of points from the sampled scene edges that are aligned with the edges of the model.

Another evaluation can be done by importing the surface model into the scene with the estimated pose to see how it compares to actual pose. This is demonstrated in Figure 4.4, the surface model is the grey pipe joint.

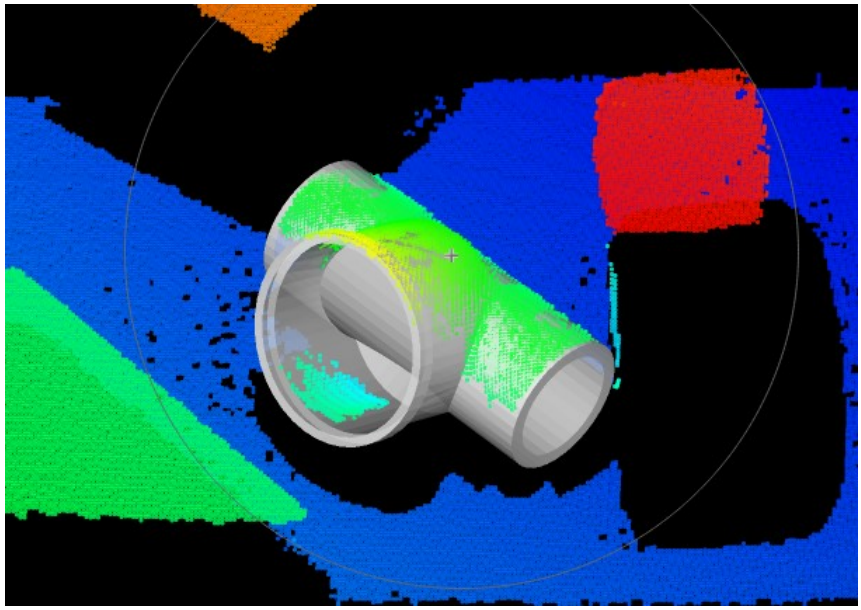


Figure 4.4.: Visualizing the estimated pose in the point cloud, grey t-pipe is the CAD model inserted into the scene

4.3. Robot programming

This section describes the operator and data types used to program the ABB YuMi robot.

4.3.1. Operator and data types

This section describes the main operators and data types used to control the robot in RobotStudio, this will serve as the foundation to understand the code. A detailed description of RAPID can be found in the manual [82].

`robtarget` is the data type used to define the position of the robot and additional axes. It consists the following parameters

- `pos` – Defines the position of TCP in x,y,z coordinates, whit mm as unit.
- `orient` – Defines the orientation of the TCP, given in quaternions q_1, q_2, q_3, q_4
- `confdata` – Defines the axis configuration of the robot. As mentioned in subsection 2.1.5 and illustrated in Figure 2.5 there are several possible configurations that achieve the same end-effector pose. ABB uses `confdata` to unambiguously denote one possible joint configuration. `confdata` has four parameters `cf1, cf4, cf6`, which represents axis 1,4 and 6, and, `cfx` which represent one of 8 possible configurations. The values for `cf1, cf4, cf6` specify the axis value by assigning them to a quadrant which gives the rotation range for the axes. The quadrants are illustrated for a 7-axis robot in Figure 4.5, the corresponding values are in Table 4.2. `cfx` is set as integer values from 0-7, different configuration can be seen in the RAPID manual [82].
- `extjoint` – Defines the position of addition axes, positioners, or workpiece manipulators. In addition to the six internal axes the robot can control six external axes, called logical axes and donated a-f. The values for these axis is given as the rotation in degrees from the calibration position if they are connected to a physical axis or 9E9 if it is not connected to a physical axis.

`jointtarget` is the data type used to define each individual axis positions (internal and externals). It consists of the following parameters

- `robjoint` – Defines the rotational position of the axis from the axis calibration point
- `extjoint` – Defines position of external axes as described above for `robtarget`

`CalcJointT` - Calculates `jointtarget` from a given `robtarget`

`CalcRobT` – Calculates `robtarget` from a given `jointtarget`

`MoveL` – Used to move the TCP to a given pose linearly, i.e., `jointtarget`. Can be given several paramters, see [82] for detailed information. Most relevant are velocity of tool center `speeddata`, size of corner in path `zonedata` and work object `wobjdata`.

`MoveJ` - Used to move the TCP to a given pose when the motion does not have to be in a straight line, usually quicker than `MoveL` and there are less problems with configuration limitations and singularities. Therefore this move instruction is preferred.

`ConfJ` – Used to control the configuration during movement. With this turned off the robot moves to the closest axis configuration. This is beneficial to use because sometimes it is not possible for the robot to use the specified configuration and program stops if `ConfJ` is turned on, which is the default.

Singularities

Singularities, as mentioned in subsection 2.1.7, are ha handled automatically in YuMi's motion planner, by deviating from paths leading to singularities

IRB 14000 Gripper

The gripper commands can be viewed in detail in ABB manual [83], below is a presentations of the most relevant.

`g_init` – Used to initialize the gripper with default or specified finger speed and hold force values.

`g_calibrate` – Used to calibrate the gripper, must be done before any movement

`g_GripIn` – Used to jog the griper inwards, can specify holding force, target position and position tolerance. Can raise an error if position is wrong.

`g_GripOut` – Used to jog the gripper outwards, can specify holding force, target position and position tolerance. Can raise an error if position is wrong.

`g_MoveTo` – Used to jog the gripper to a specified position.

-4	$-315° \rightarrow 45°$
-3	$-225° \rightarrow -315°$
-2	$-135° \rightarrow -225°$
-1	$-45° \rightarrow -135°$
0	$-45° \rightarrow +45°$
1	$+45° \rightarrow +135°$
2	$+135° \rightarrow +225°$
-4	$+225° \rightarrow +315°$

Table 4.2.: `confdata` quadrant values for 7-axis robot

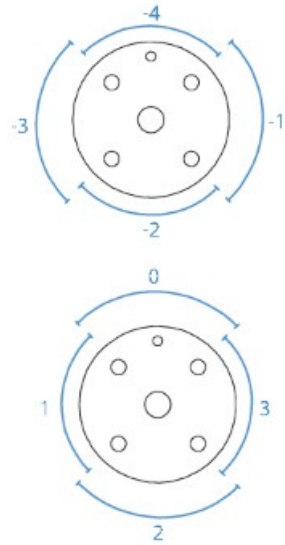


Figure 4.5.: `confdata` illustration

Chapter 5.

Result: System

This chapter describes how the software for the bin-picking system works. A simplified version of the complete pipeline is illustrated in Figure 5.1.

5.1. Setup

This section describes camera positioning, hardware connections, hand-eye calibration procedures, how the CS is defined for each part and the approach used to determine the acquisition settings.

5.1.1. Working Distance and Camera Positioning

Working distance and camera positions are critical for point cloud quality, first the region to be captured must be established, then zivid provides a data sheet and a FOV calculator [84] to define the position. The camera should not be mounted perpendicular to the scene but at a slight tilt to avoid blooming effects. Additionally in bin picking applications the camera should be mounted in such a way that the 2d camera focuses on the center of the bin and that the projector rays do not fall on the inner surface of the two walls closest to the camera as shown in Figure 5.2 [85]. In the robotics lab at NTNU the Zivid camera is mounted on top of the YuMi as shown in Figure 5.3, where $\alpha = 15°$ and $l = 71cm$. Following the datasheet [53] for the camera this setup is within a good working range for the camera. To avoid noise from the surface around the objects a black textile is used, this is based on zivid recommendation when working with shiny objects [38].

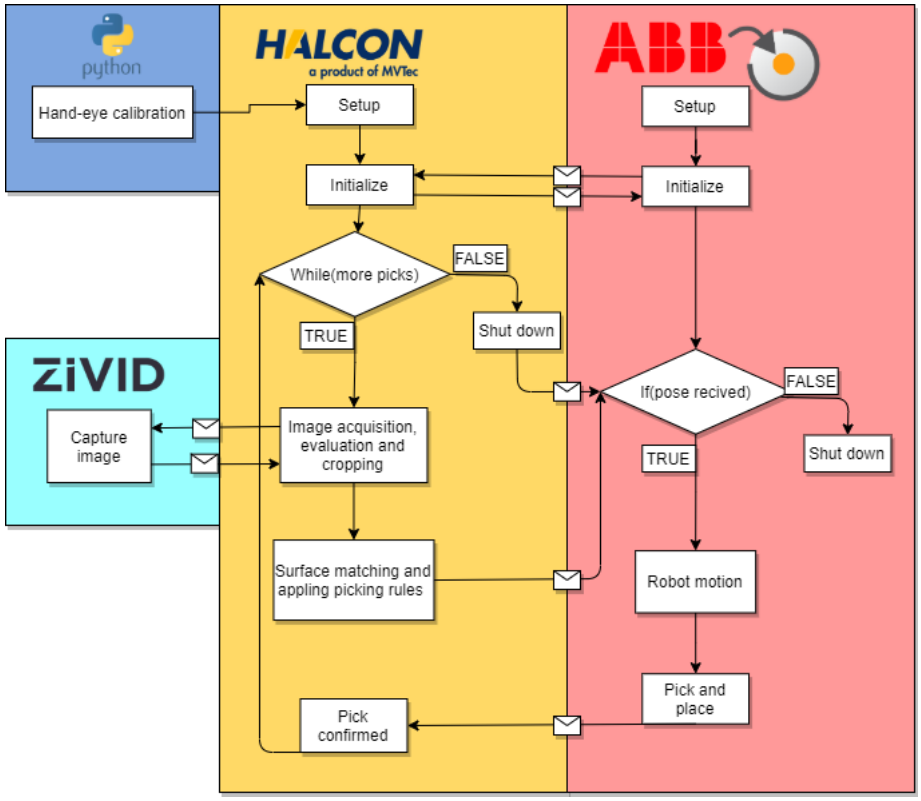


Figure 5.1.: Simplified system overview

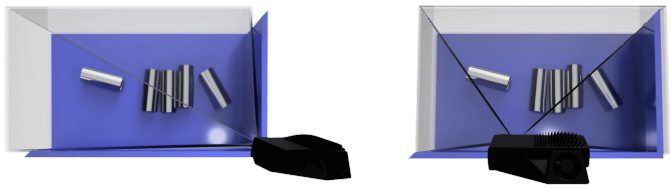


Figure 5.2.: Camera mounting bin picking [85]

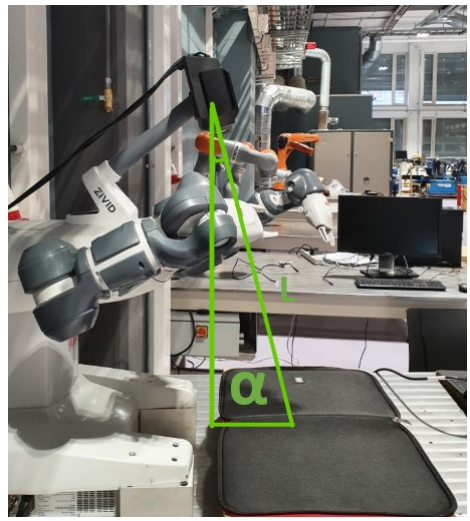


Figure 5.3.: Setup lab, showing the angle, $\alpha = 15°$, and the distance, $l = 71$cm from the camera to the scene

5.1.2. Connections

The robot is physically connected to computer via an Ethernet cable and the communications is a TCP/IP socket communication. The camera is connected to the computer via USB.

5.1.3. Hand-Eye Calibration Zivid

The hand-eye calibration functionality in the program is a resource from zivids github. It has been modified to work with the ABB robotic system. The number of poses used are user determined and they are defined before the first while loop. The poses are chosen based on Zivids recommendations, illustrated in Figure 5.4 and the position is focused around the same area as the bins will be placed. This loop runs as long as the calibration has not been done, it then leads to second while loop which runs until all poses has been used. In this loop the robot is moved

to calibration position by the user and a image is captured. A feature recognition program is used to detect the checkerboard corners and store the corresponding feature points. These features are paired with the pose in a dataset and used for the hand-eye calibration. The accuracy of the resulting transformation matrix can be evaluated with translation and rotation residuals.

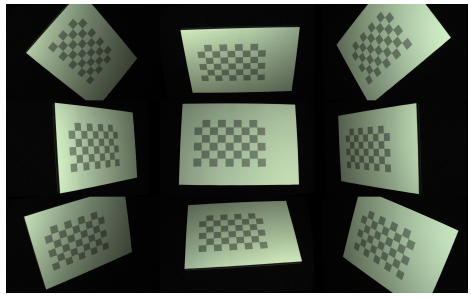


Figure 5.4.: Recommended checkerboard angles Zivid [86]

The Hand-eye calibration pipeline is illustrated in Figure 5.5. Before starting the code, the scene is analyzed in zivid studio and capture settings for the camera are stored in a .yaml file.

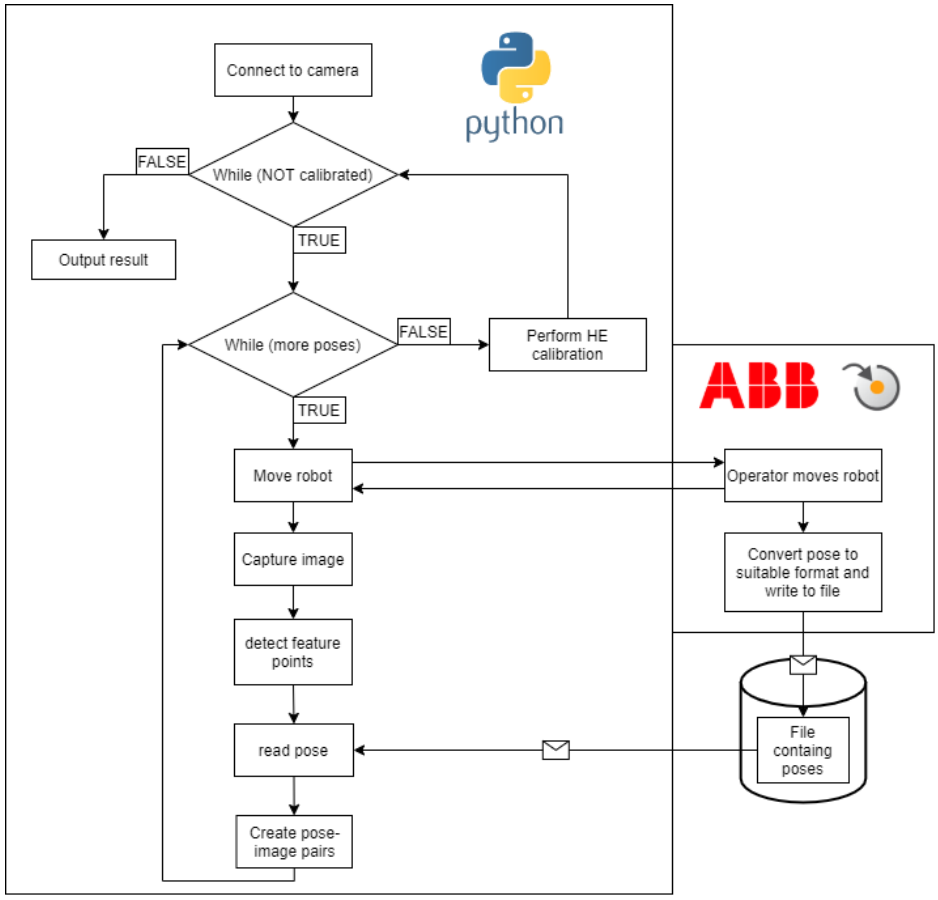


Figure 5.5.: Diagram of the hand-eye calibration pipeline using Zivids procedure

The calibration object used is the recommended gray and white checkerboard from Zivid shown in Figure 5.6.

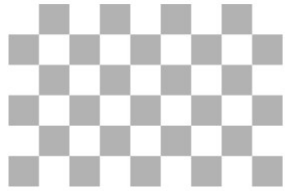


Figure 5.6.: Zivid recommended calibration object; gray/white checkerboard

An advantage to using Zivids hand-eye calibration compared to opencv is that is uses 3D data which results in a more accurate calibration as can be seen in

Figure 5.7 and Figure 5.8.

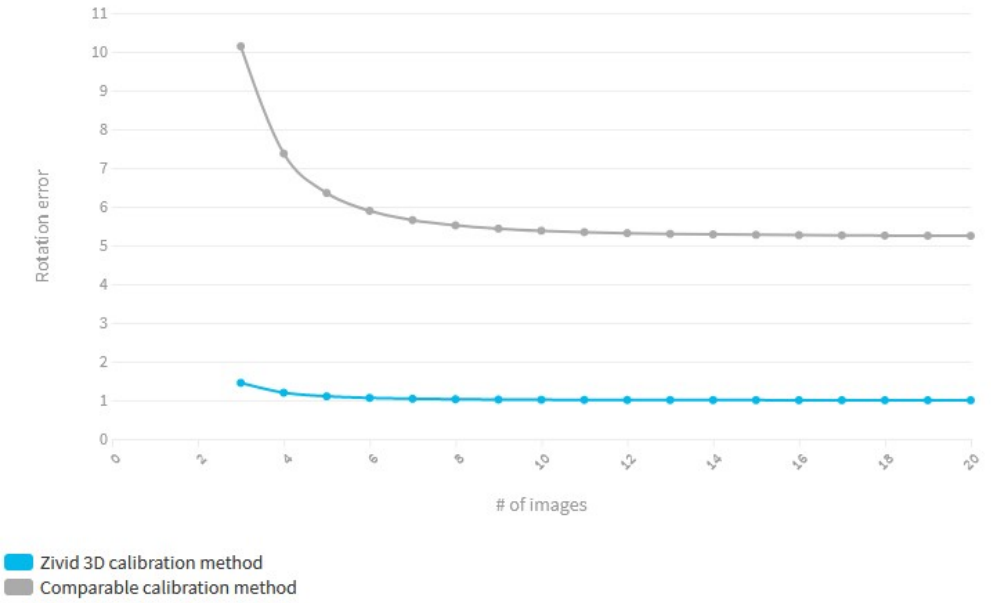


Figure 5.7.: Zivid Hand-eye calibration compared to with comparable methods. Analysing number of images vs. rotation error [87]

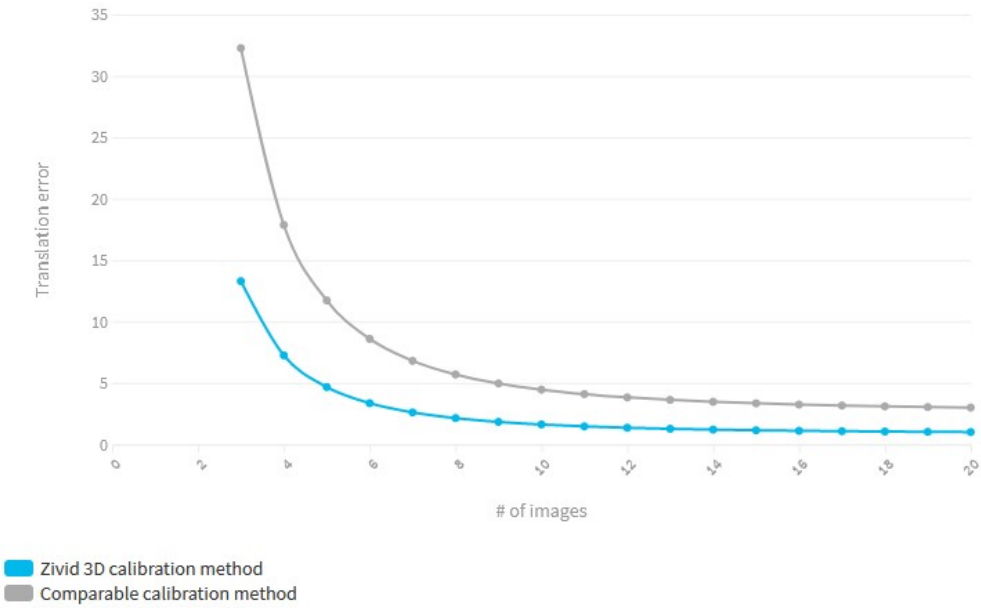


Figure 5.8.: Zivid Hand-eye calibration compared to with comparable methods. Analysing number of images vs. translation error [87]

5.1.4. Hand-eye calibration HALCON

HALCON has two methods for calibration that can be used in this project, the first one utilizes a calibrated 2D camera, a feature recognition procedure called `find_calib_object` and special calibration plate developed by HALCON. The intrinsic calibration data for the camera was given by a representative of Zivid. The second use the 3D camera to generate a point cloud in which a surface matching procedure searches for a predefined calibration object in the scene as described in section 4.2. A detailed description of the procedures can be found in the manual [81]

Calibration object

The HALCON calibration consists of diagonally arranged marks as shown in Figure 5.9. Depending on the size the plate contains 1-5 finder patterns that are a special mark hexagon where 4 or 6 of the marks contains a hole. These finder patterns are unique and used to determine the orientation of the plate and the position of the finder patterns on the plate. The calibration object used in 3D calibration is user determined, in this case the hand of the robot is used.

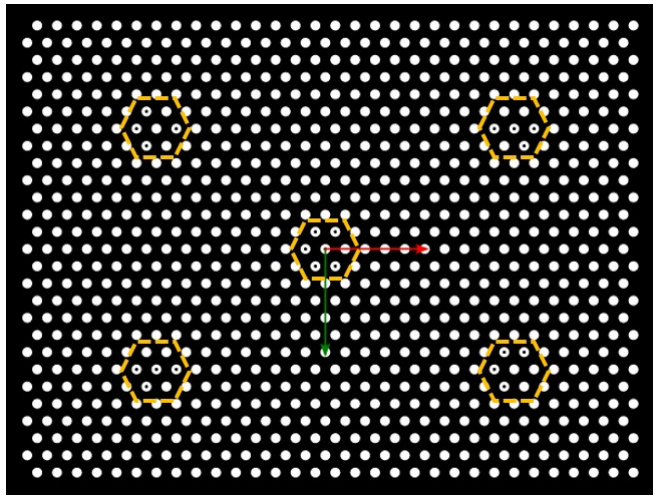


Figure 5.9.: A standard HALCON calibration plate with hexagonally arranged marks and its coordinate system. The yellow hexagonals highlights the finder patterns [88]

5.1.5. Hand-eye calibration residuals

Evaluation of hand-eye calibration is done with a translation and rotation residual. Using a checkerboard as an example where the detection algorithm extracts a certain set of feature points corresponding to the checkerboard corners for each captured image. Theses feature points are given in camera coordinates and transformed to base coordinate with the result from the hand-eye calibration. If the hand-eye calibration, camera and robot where perfect all feature points from the data set would have exactly the same coordinates. This is not the case in the real world and residuals are used as an way to estimate this difference.

Zivid residuals

Zivids hand-eye calibrations calculates reference feature points as the arithmetic mean of all other feature points set. The residual is given as the the relative position between all the feature points and the reference points [89].

5.1.6. Define CS for each part

The CAD model of the parts need some initial work before they are ready to be used in the surface matching procedure. Firstly, if there are hollow cores or other features that are not visible for the camera they are filled or simplified. This is done to reduce the complexity and the correlated matching speed of surface

matching calculations. Secondly, the best suited object coordinate system for picking each part is defined, this is directly related to how the object is picked as this is what the robot will align its end effector coordinate system with. How the coordinate system is defined for the parts is shown in Figure 5.10 - Figure 5.12

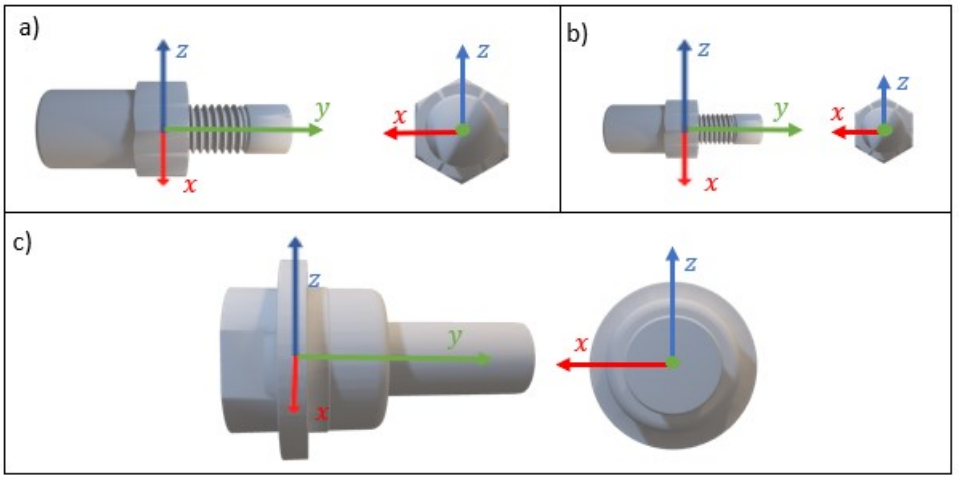


Figure 5.10.: CS definitions for symmetric cross sections, a) LFC, b) SFC and c) brass connector

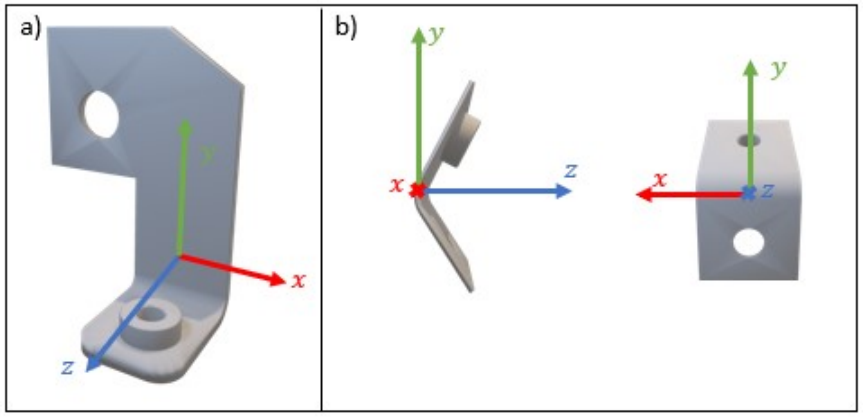


Figure 5.11.: CS definitions for a) large and b) small lask

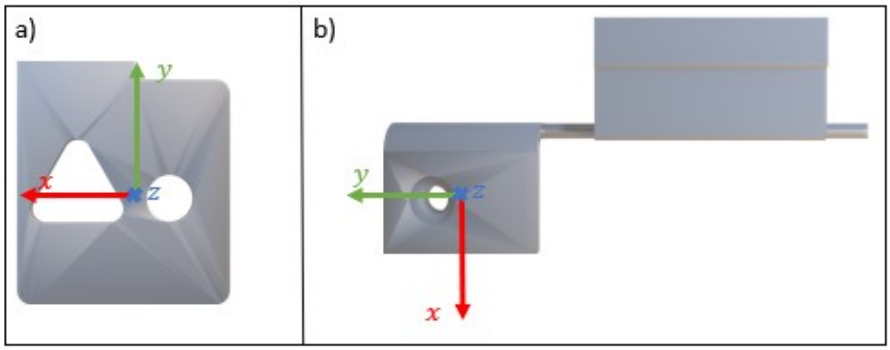


Figure 5.12.: CS definitions for lask in foam, a) triangle lask, b) elongated lask

5.1.7. Image acquisition settings

The image acquisitions settings are determined in Zivid Studio by first using the assisted mode described in subsection 3.1.2, which sets number of acquisitions, settings and filters. Then the capture is evaluated with respect to highlights, RGB values and SNR. To evaluate the source of the highlight in the scene the brightness is set to 0 and the projector is covered up. Then it is compared with the initial image to check if the projector causes the highlights. The next step is to find the lowest imaging sensor exposure by only using the first acquisition to check that it reveals a dark scene without containing many overexposed pixels, this is the highlight acquisition. The next step is to expose for mid-tones, this is done by evaluating the histogram and adjusting the settings, mainly by adding stops in the second acquisition, to move the region with an intensity bellow 32 into the upper half of the histogram. Finally additional acquisition can be used to increase the exposure towards lowlights by adjusting stops, brightness and gain. Then the filter is adjusted to correct for the errors described in section 2.4.

5.2. Initialize

The initialization procedure runs once at startup and it is here the system is configured and connections are established. ${}^{base}T_{cam}$ from the HE calibration is inputted, connection to the camera is established, HALCON surface models are created from .ply, socket communication between the computer and robot is setup and connected, parts to pick are determined and the gripper is calibrated. These steps are illustrated in Figure 5.13 and described in the following section.

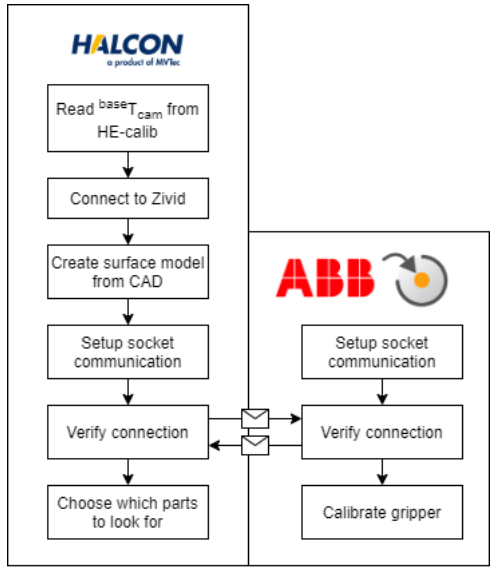


Figure 5.13.: Diagram of HALCON and ABB initialize

5.2.1. Connecting to the Zivid camera

A connection to Zivid camera is established by first using the `open_framegrabber` procedure, which is a query that returns a list of image acquisition devices. This list in used as input in a Zivid developed procedure called `get_first_available_zivid_device`, which returns the first zivid device from this list. This is then used in `open_framegrabber` procedure that open and configure the image device and returns a handle for it.

5.2.2. Create HALCON surface models from CAD

A local procedure `CreateModelFromCAD` creates HALCON surface models by using HALCON procedure `read_Object_model_3d` and `create_surface_model` as described in subsection 4.2.1 with a `RelSamplingDistance` = 0.02. These surface models are appended to tuple `SurfaceModelID` which is returned to `main(:::)`. The following tuple shows how the parts are sorted.

```
SurfaceModelID := [0.fc, 1.fcSmall, 2.  brass_connector, 3.  Small_lask,
4.  large_lask, 5.  triangle_lask, 6.  large_lask_foam, 13.  Nut M12]
```

5.2.3. Open socket communication

Socket communication in HALCON is initialized by declaring protocol, in this case TCP4, IP and Port number. HALCON procedure `open_socket_connect` opens a socket, connects it to the IP and port declared in RAPID and returns a handle. A similar procedure is run in the robot OS and a message is sent between them to confirm that the connections is established.

5.2.4. Choose which part to look for

Which parts and how many of them to look for is user controlled by inputting the number of parts into `Num_parts` tuple which follows the same index as in `SurfaceModelID`, e.g. `Num_parts := [3,2,...,0]` means pick 3 of the large female connector (LFC) and 2 of the small female connect(SFC). Nested `for` loops are then used to create `pickTuple` which is a tuple consisting of part index and number of parts, `pickTuple := [0,0,0,1,1]` is how it would look for the example above.

5.2.5. Gripper calibration

The gripper is calibrated after the connection is established by running using the `g_calibrate` command.

5.3. Point cloud acquisition, evaluation and cropping

This section the method of capturing, evaluting and cropping the pointcloud in HALCON.

5.3.1. Acquisitions

Images are captured using the local procedure `captureImage` which is based on an example on how to capture images with a Zivid 3D camera in HALCON. The resulting settings from the method described in section 4.1 are implemented in the procedure and the image is converted to a .ply format.

5.3.2. Evaluation

After the image is captured an evaluation filter is run to ensure there are no major errors in the point cloud caused by reflections. The filter evaluates the point cloud by creating a bounding box around using the HALCON procedure

`smallest_bounding_box_object_model_3d()` which returns the three lengths of the bounding box in descending order. Based on the camera FOV, the width and height of the point cloud is known, and the depth can be estimated. The depth is the largest value, and it is this that is compared to the largest value of the bounding box, if the returned length exceeds a predefined threshold, then a new image is captured. This is done until the captured point cloud size is within the expected values.

5.3.3. Cropping

There are several implementations that can improve the accuracy and reduce the time of the surface matching method. The following section describes two developed methods to narrow the search area based on different techniques.

Using ArUco markers to define the ROI

This approach uses ArUco markers attached to each part box to define the ROI. A python script using functionality from OpenCV and Open3D is developed to achieve this.

First OpenCV is used to identify the ArUco marker in the 2D color image from the camera. After identification the transformation from the camera to the ArUco marker is estimated using the four corners detected in the marker. Then the subsequent transformations from the ArUco marker to the the four vertices of the bin are calculated. These transformations are based on the dimensions of the box, the distance from the marker to the box and the assumption that there are no rotation from the AruCo marker to these vertices, this is shown in equation (5.1). The points are illustrated in Figure 5.14. From the points Open3D is used to define the bounds for the crop, i.e., a polygon that encloses the region of interest. The result is a cropped point cloud only showing the selected ROI. This procedure is demonstrated using a box with pens and 12 ArUco markers. The program is told to look at ArUco marker number 10, the detection and pose estimation result is shown in Figure 5.15. Based on the pose the point cloud is cropped as shown in Figure 5.16

$$^{cam}T_{point} = {}^{cam}T_{ArUco}\,{}^{ArUco}T_{point} \tag{5.1}$$

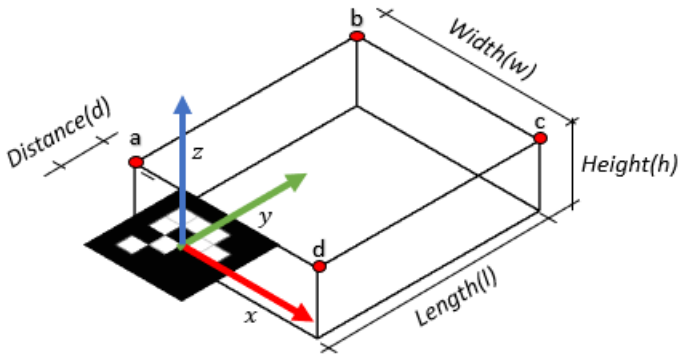


Figure 5.14.: Define the size of the bounding box given in the coordinates system of the detected ArUco marker

Figure 5.15.: Step 1: Detect the ArUco marker and the pose

Figure 5.16.: Step2: Crop the pointcloud based on ArUco

Using fixed locations on a tray

This method uses a tray to fix the bin locations, the x and y axis of the tray are collinear with the x and y axis of the camera. With this approach we can use the HALCON operator `select_points_object_model_3d()` that crops the pointcloud based on a max/min x, y value. This is illustrated in Figure 5.17 and the value calculations are shown in (5.2) and (5.3).

$$x_{crop,min/max} = x_{cal} + x_{min/max} \tag{5.2}$$

$$y_{crop,min/max} = y_{cal} + y_{min/max} \tag{5.3}$$

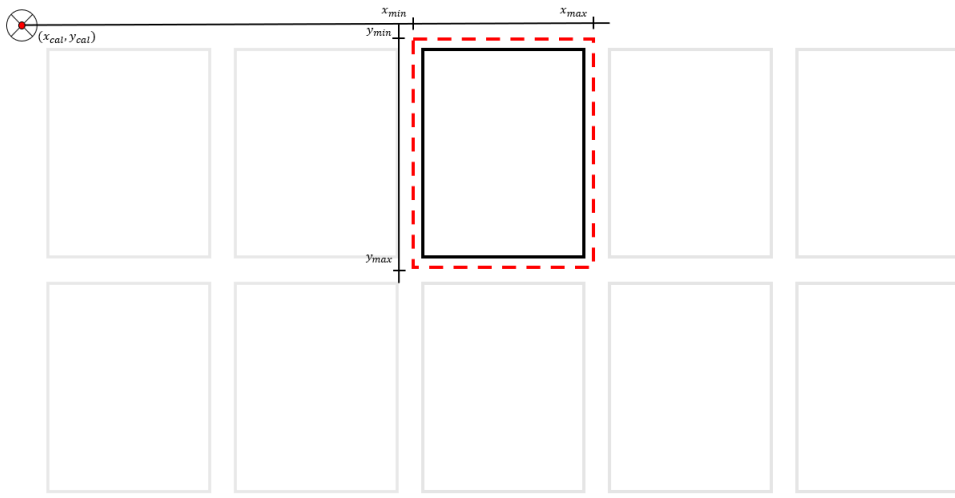


Figure 5.17.: Define ROI with axis aligned sorting tray based

5.4. Surface matching & create target

This section describes surface matching, match evaluation, picking rules and the conversion to robot target.

5.4.1. Surface matching

The surface matching is as described in subsection 4.2.1 with the following settings. `RelSamplingDistance` = 0.02, `KeytPointFraction` = 0.05, `num_matches` = 5 and `scene_normal_computations` = mls. The procedure returns 5 matches, this is done because early testing in the specialization project showed that false matches can have the highest score [4]. By returning more matches they can be further evaluated to identify a good match.

5.4.2. Match evaluation

The primary evaluation of the matches is done by using the `model_point_fraction` from the surface matching procedure as described in subsection 4.2.3. The returned matches may be a good or they may be false positives, to further evaluate a threshold is defined to remove false positives. This threshold combined with the highest(max) score is used to filter the matches by eliminating all matches that has a score bellow $score_{max} - threshold$. This is implementing uses loops which iterates over the returned scores, the valid scores are appended to a new list.

The next matching criteria is the position of the object in the bin, i.e., the z-value of the object given in base coordinates. The assumption is that the object highest in the bin is also the one that is easiest to pick because there are no objects above it hindering the gripper from grasping. To evaluate this the z-value for each object is compared and the match with the highest z-value is used for picking.

Additional evaluation LFC, SFC and brass connector

For the LFC, SFC and brass connector and additional evaluation criteria based on orientation is used in combination with the z-value. It is desired that object is situated with the longitudinal axis (y-axis) close to parallel with the xy-plane of the robot base, as shown in Figure 5.18. This is because this orientation has the lowest risk of having adjacent object covering some parts of it and it is also easier for the robot to grasp. To evaluate this the angle between the object y-axis (longitudinal axis in green) and the robot base plane is calculated. If this angle is bellow a predefined threshold this object is passed forward as the best match.

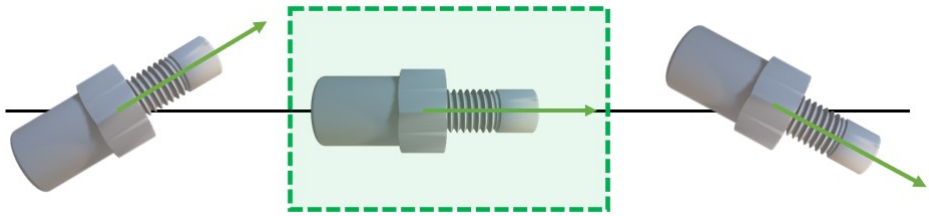


Figure 5.18.: Evaluate match based on orientation

.

5.4.3. Grippers

To grasp the parts different solutions have been proposed, use the stock gripper with and without modifications, create a custom gripper and use the vacuum gripper.The stock gripper are explained in subsection 3.3.1. The modification done was to add vulcanized rubber on the inside of the fingers to increase the friction between the finger and the part, as the stock material is a hard plastic with low friction. The modified gripper design is based on the stock grippers, they use the same mounting mechanism and have the same width and length. This means that the gripper tooldata, i.e., pose in relation to the wrist does not need to be altered and they can easily be switched out. The thickness is reduced, and fingertips are narrowed, the intention is to make it easier for the fingers to slide into the gaps between the parts. The following thicknesses were printed in PETG

plastic: 2,3,4 mm. CAD drawings and material properties can be found in the appendix.

Possible grasps

To get an indication of how the robot would cope with gripping the parts in different pose, a selection of parts where used as a test. The brass male connector is used to test strength because this is the heaviest part and has a difficult geometry to get a good grip on. Bracket nr 3 in Figure 1.6 was used to test strength at the end of the grips working range because of its thin cross-section. This was followed up with using bracket nr 2 to test how it would manage a heavier part. The female connector was used to test the range of object poses it could get a grip of. The result of these test can be seen in Figure 5.19, Figure 5.20 and Figure 5.21 respectively.

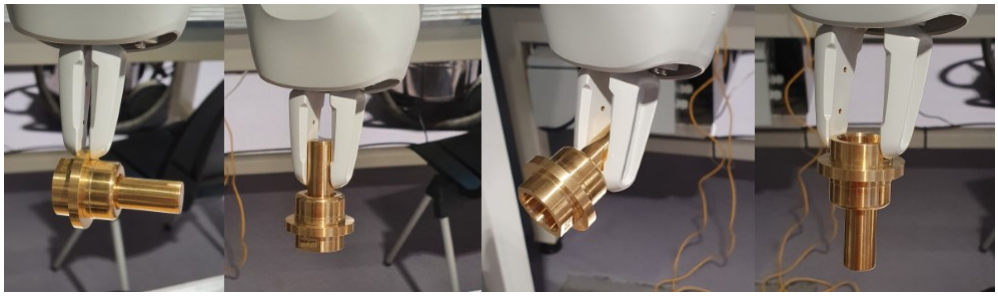


Figure 5.19.: Test of grip strength on various poses of the brass connector

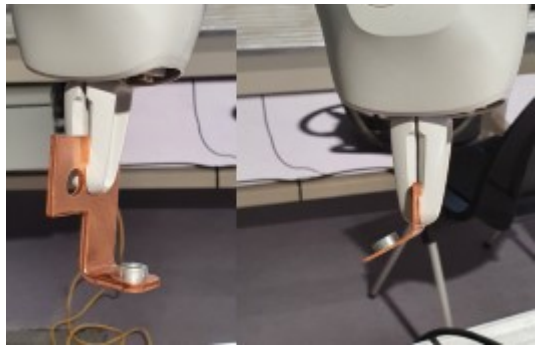


Figure 5.20.: Test of grip strength at end of gripper working range on bracket 2 and 3

Figure 5.21.: Test of grip on various object poses of the female connector

5.4.4. Picking rules

When a match is found the returned pose of the match is that of the parts CS and this is then used as a target for the robot to align its end-effector CS. Therefore how the CS is defined for each parts must be determined is such a way that it can be used as target for the end-effector to grip. This target however can be oriented in ways that make it impossible of the robot reach, e.g., if the target pose is so that the robot would have to pick it from the bottom. Therefore, a certain set of rules and transformations are applied to the match pose before it can be sent as a target to the robot. These procedures are defined for subgroups of parts with similar geometries and individual part depending on how they are situated, their shape and how the CS is defined for the parts.

Symmetric cross section

The large and small female connector, and the brass connector have symmetric cross-sections. This implies that the angle they are grasped with can be chosen arbitrarily, so the first step is to rotate the target so that the z-axis points in the negative z-direction of the robot base, this ensures a top down pick as described in subsection 2.7.2. To do this the target is rotated around its y-axis until the x -axis is close to parallel with the robot xy-base plane, then the z-value of the z-axis is checked, if it is positive the target is rotated 180 degrees about the y-axis,

if it is negative nothing is done. This functionality is executed in the procedure `DefineTarget_Symmetric CrossSection` and is shown in Figure 5.22

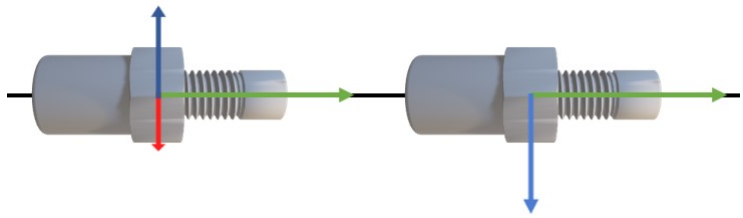


Figure 5.22.: Symmetric cross section rotate target

Additional rules for the brass connector

Depending on the angel between the y-axis of the brass connector CS and the xy-plane of the robot there are two possible picks defined. If the angle is greater than 10° then part is picked perpendicular to the y-axis, i.e., grasp the ending rod of the brass connector, this is illustrated in Figure 5.23. If the angle is less then 10 ° then the object is picked by its outer ring as illustrated in Figure 5.24. This functionality is implemented in the local HALCON procedure `DefineTarget_brassConnector`.

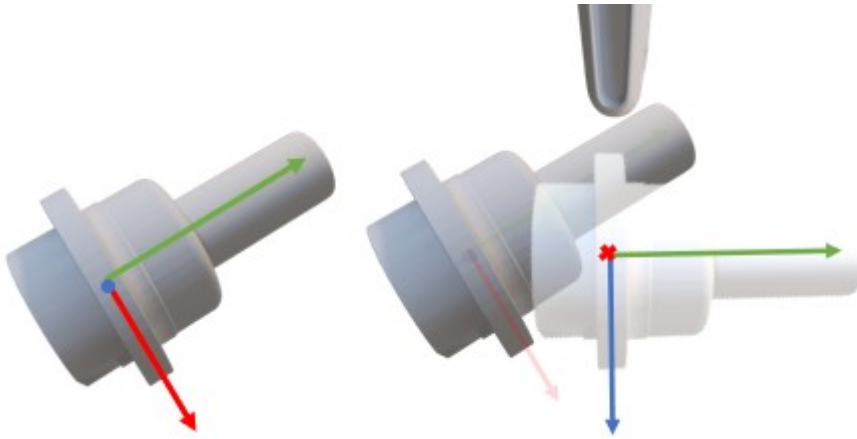


Figure 5.23.: Pick brass connector perpendicular to the y axis

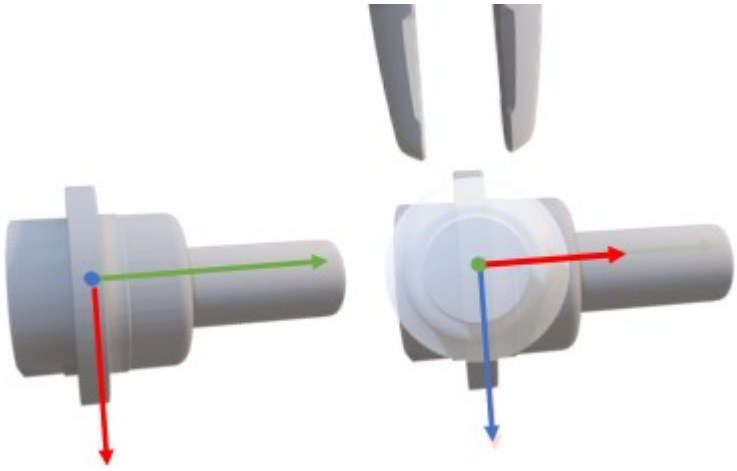


Figure 5.24.: Pick brass connector outer ring

Large lask

For the large lask to possible picks are defined based on how it is oriented in the bin. Ideally it should be picked as illustrated in Figure 5.25 a), if this is not possible because the blocky upper parts of lask hinders the gripper from reaching the pose it should be picked as illustrated in Figure 5.25 b). To implement this functionality a local HALCON procedure called `DefineTarget_largeLask()` has been made. First the procedure checks the orientation of the object y-axis compared to base plane of the robot, if this angle exceeds 60°s then we are in scenario b). Then the target is rotated so that the z-axis now points in the negative y-axis of the object CS and the x-axis points in the z-axis of the object CS and translated in the object y-axis to move it closer to the edge of the part. This is all illustrated in Figure 5.25 b). In the other scenario the target must be rotated so that the z-axis points downwards ensuring a top down pick, the procedure steps is illustrated in Figure 5.26. Even though the object is at an angle the target is always planar.

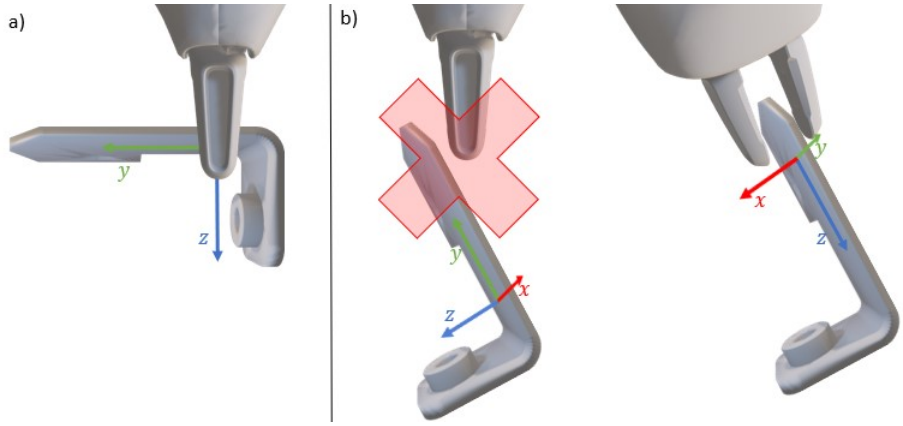


Figure 5.25.: Grasps large lask, a) scenario 2, b) scenario 1

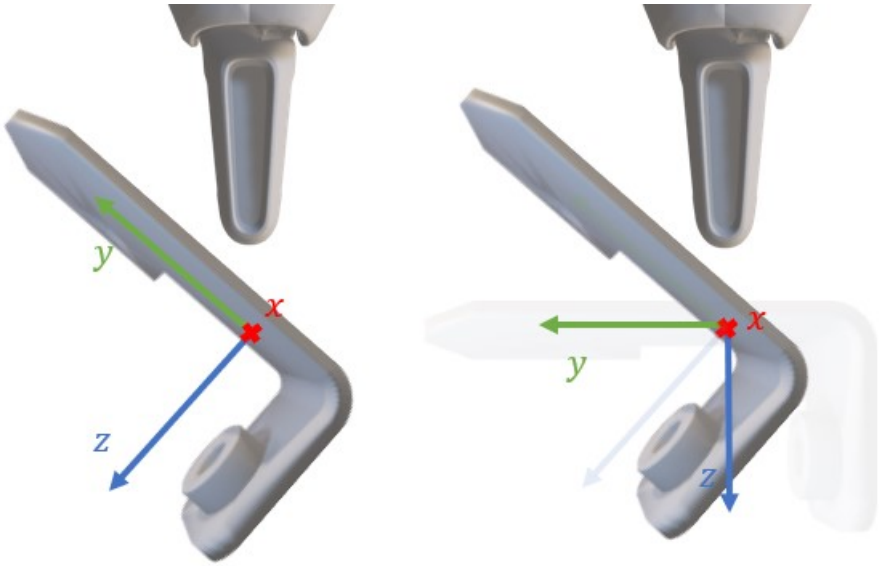


Figure 5.26.: Pick large lask, target transformation scenario 2

Small lask

For the small lask to possible picks are defined based on how it is oriented in the bin. Ideally we want to pick the lask at the center where the CS is defined, but in some cases this is not possible. These cases are when the lask lies "face up" as illustrated in Figure 5.27. As it can be seen it is not possible to get a good grasp at the center, therefore the target is moved in either the positive or

negative object y-axis direction depending on which side the parts tilts towards as illustrated in Figure 5.27. Finally the target is rotad so that z-axis points in the negative z-axis of the robot base. The other case is when the lask can be picked in the center, then the target must be rotated so that the z-axis points in the negative z-direction of the robot base. This is done by rotating the target so that the target x and y axis are planar with the robot xy-base plane. This is illustrated in Figure 5.28 . The functionality described is implemented in the local procedure `DefineTarget_SmallLask()`.

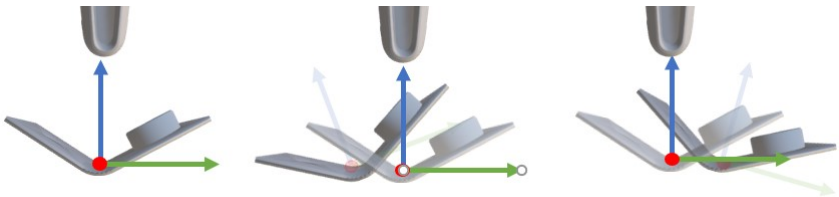


Figure 5.27.: Small lask face up

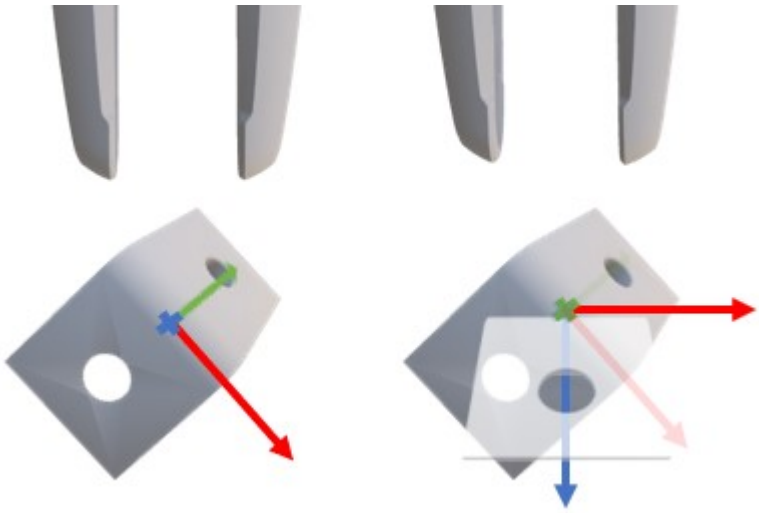


Figure 5.28.: Rotate target to pick small lask

Lasks in foam

The triangle lask and elongated lask are confined to the foam cut-out and therefore always have the same orientation. So the defined CS is the target for these parts and it is defined at the best suitable place to pick them given the restraints of the

part and the cutout. The CAD model of the parts were modified so that only the parts of them that are visible in the camera is used, i.e., whats not in the foam..

5.4.5. Convert to robot target

The surface matching procedures and subsequent application of picking rules use the homogeneous transformation representation shown in (2.4). This is transformed to the representation shown in (5.4) because ABB requires this representation.

$$^{robot}p_{object} = [[x, y, z], [q0, q1, q2, q3]] \tag{5.4}$$

5.5. Robot motion

This section describes the generation of `robtarget` and robot motion.

5.5.1. Target generation

The position and orientation are used in combination with an initial guess for `confdata` = [0,0,0,0] and `extjoint` = [101.964427336,9E+09, 9E+09,9E+09,9E+09,9E+09] to create a `robtarget`. This `robtarget` is fed into a target generation pipeline that first creates `jointtarget` with the rapid procedure `CalcJointT` to calulate the joint angels of the robot axes and external axes, then uses this to create a `robtarget` with the RAPID procedure `CalcRobT`. This target is checked for error with an error handler.

5.5.2. Robot positions/targets

This section describes the various poses the end-effector is in during a bin picking operation. All poses are defined using `robtarget`.

Home:
The Home configuration is defined by ABB and it is used in this program when the robots work is done and it is going to be turned off, and to calibrate the robot robot. There are extrude edges in the casing on both sides of the joints on the robot so that joints can be jogged to obtain this position when the robot is calibrated.

Ready:
The Ready pose is a predefined pose used as a waiting point, it is defined out of FOV of the camera and in stretched out joint configuration. This speed up the

process because the camera can take new pictures when the robot has reached this point and it gives the inverse kinematics algorithm of RobotStudio a better starting point. Testing have shown that moving to the Pick directly from the home configuration causes joint limitation which, with the Ready pose, can be avoided.

AbovePick:
The AbovePick pose is used to get a good starting point before the pick and to ensure that the arms does not collide with the bins on the way to target. Without having this pose defined and used as a waypoint the system would create a path directly to the object which would lead to a collision with the bimn walls. This pose is defined as the pick target translated 100mm in the negative z-direction of the target, this is shown in Figure 5.29. This pose is also used after the object is picked for the same reason as mentioned.

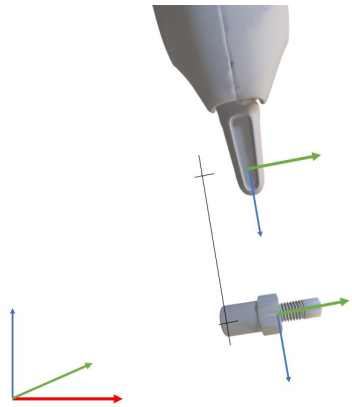


Figure 5.29.: Above target pose

Pick:
The pick pose is the pose of the end-effector target, i.e., the pose after the pose estimation algorithm and gripping rules have been applied. The rules for different objects are described in subsection 5.4.4.

AbovePlace:
The AbovePlace has the same purpose as the AbovePick pose. It is there to ensure a god starting point for placement and avoid collision.

Place:
The placepose is defined differently depending on the part. For LFC, SFC and brass connector this is the intermediate step where the part is placed in the funnel that aligns the part in the same pose each time to correct for variations in the

when picking. For the lask this pose is also an intermediate step where the parts is placed on flat tray to correct for any slip when picking.

5.5.3. Robot motion between poses

There are different operators to move the robot as mentioned in section 4.3. To move the robot between poses AbovePick, Pick and Place `MoveL` is used with high accuracy and slow speed. For the other poses `MoveJ` is used with high speed in low accuracy.

5.5.4. Grasping

The `g_GripIn` command is used to verify that an object is gripped by adding the `targetPos` and `posAllowance` variables. E.g if the object is hexagonal nut with dimensions as shown in Figure 5.30 the command would be as follows: `g_GripIn, targetPos := 9, ,targetAllowance := 4`. For the system the `targetPos` is the width of the part which is sent from HALCON.

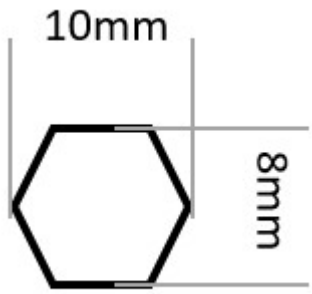


Figure 5.30.: Dimensions hexagonal nut

5.6. Communication

The communication between the robot and HALCON is setup as a continuous, i.e., there are multiple messages being sent during the picking process. When the robot is out of the FOV of the camera a message is sent to HALCON which then starts the target generation and returns a message confirming that the pose of the target is calculated. The robot responds with a ready message and HALCON sends the target along with the specified gripper width. This is packaged into a string that is unpacked and converted to `pos` and `orient` data types presented in subsection 4.3.1 in the robot OS. This is looped as long the system is running. If

the robot experiences an error while picking, an error message is sent to HALCON which starts the error handler procedure. Regardless of when this message is sent in the loop, HALCON always inspect the message first to verify that is not an error.

5.7. Error handler

The error handlers in HALCON is the mentioned point cloud evaluation in subsection 5.3.2 and if the robot returns an error message. If an error message is returned the system returns an error message to the robot, closes the socket connection and stops the program, the same thing happens in the robot program. The error handler for target generation is used when the proposed `confdata` configuration is not usable for the given pose and the `ERR_ROBLIMIT` error is raised. The handler iterates through different configuration suggestions until it reaches its limit. If the limit is reached an local error handler procedure is run that messages HALCON that the target can not be reached, then HALCON and the robot system is shutdown.

If the `targetPos` is not reached `ERR_HAND_FAILEDGRIPPOS` is raised. The error handler tells the system to move back to ready pose and a new acquisition is made and the procedure restarts.

5.8. Picking loop summary

The main part of the procedure is placed within a while loop which keeps looping until `pickTuple` has been iterated trough or an error has been raised in the matching procedure or robot programming. The pipeline is illustrated in Figure 5.31 and this section describes the described methods are implemented in the script.

The picking is initiated by the robot when it sends a message “captureImag” to HALCON signalizing it is ready to pick. Based on this message HALCON first checks that there are more picks, if it is it starts the image capture procedure as described in subsection 5.3.1 and evaluates the point cloud as described in subsection 5.3.2. If there are no more picks an exception is raised and the program termination procedure is executed. The program continues to capture images until the point cloud is approved and when it is approved the surface matching begins.

The `SurfaceMatch` procedure contains all functionality from image processing to final pose conversion. The first step is to crop the point cloud using the procedure based on fixed locations as described in subsection 5.3.3. The cropped image is used in the HALCON `find_surface_match_procedure` which is set to return

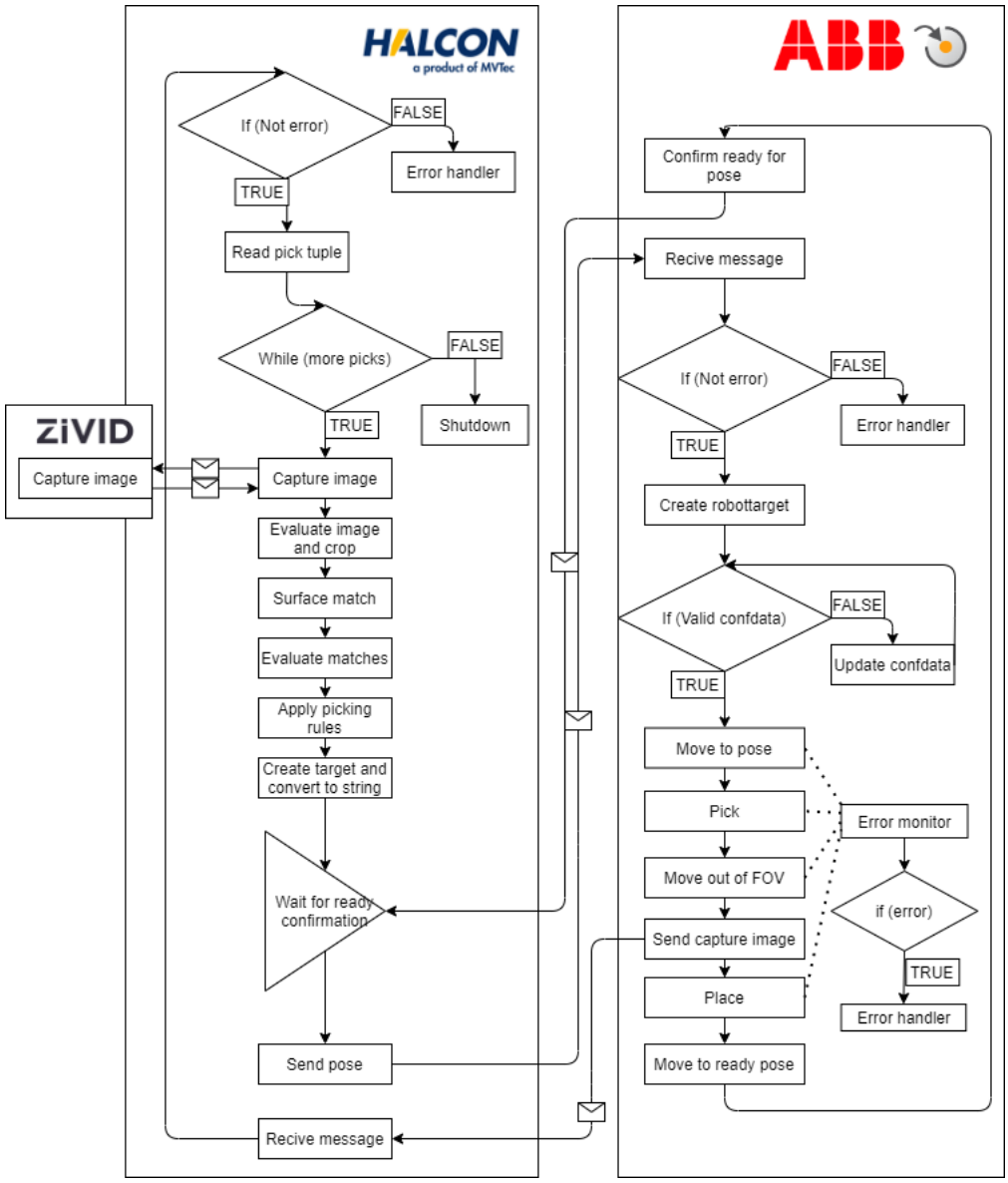


Figure 5.31.: Diagram over the main loop of the system

5 possible matches. These matches are evaluated in the local `EvaluateMatch` procedure described in subsection 5.4.2. To create the target the match that best meets the criteria is used and the picking rules described in subsection 5.4.4 is applied depending on which part it is. From this target the above target pose is defined. Finally the target and above target are converted into a string in the preferred representation as described in subsection 5.4.5. The procedure returns both poses and specified gripper with needed for the given part to `main(:::)`

The target and above target are sent separately to the robot where the pose is unpacked as described in section 5.6 It is then used to calculate the robottarget as described in subsection 5.5.1 If there are any errors while calculation the target the error handler is used to change the configuration. Before the robot starts is motion towards the target it checks that it opens the to the defined gripper width and checks that it is in the readyPose. It then starts moving towards the above target via an predetermined intermediate step to ensure no collisions with the bins. All these moves are done ss described in section 5.5. When the robot has reached the target the gripper activates, holding the object whit max force, picks it up and moves it to the place position. This route goes via the ready pose and when it has reached this pose the arm has cleared the camera FOV and "captureImag" is sent to HALCON telling it to capture a new image and start the loop again while the robot continues to the place location where it places the object and moves back to ready position and waits for a new pose. If there are any error along the way these are handled by the error handler described in section 5.7

Chapter 6.

Result: Evaluation

This chapter presents an evaluation of the system performance. The adjusted image acquisition settings are compared with assisted mode settings. The hand-eye calibration implementation are presented along with resulting residuals. The target generation pipeline is evaluated with respect to matching score, number of good matches and usability of the target match. The performance of the custom gripper fingers and the usability of the vacuum grippers are evaluated. The system time from capturing image to generated match target is evaluated. An evaluation of the systems ability to continuously pick is tested and evaluated. Finally common errors are described.

6.1. Acquisition settings

The assisted mode with the phase engine was used to set the initial settings showed in Figure 6.2 which returned the histogram shown in Figure 6.1. The highlighted area in the histogram shows that 45.3% of the image has RGB values that resides between 32-255. The highlight acquisition shows only a few pixels in a dark scene. Implementing the second acquisition shows that 54.7% of the image has values bellow 32 and 1.8% has values between 216-255. After adjusting the aperture to 2.38, 34.8% of the image has values bellow 32 and 4.4.% has values between 216-255. In the resulting acquisition 64.9% of the image is resides between 32-255. In the final acquisition the gaussian smoothing filter were adjusted to 1.0 and the contrast distortion filter where set to correction and removal with a threshold of 0.4. The resulting histogram after the adjustments and the settings and filter used is shown in Figure 6.1 and Figure 6.2 respectively.

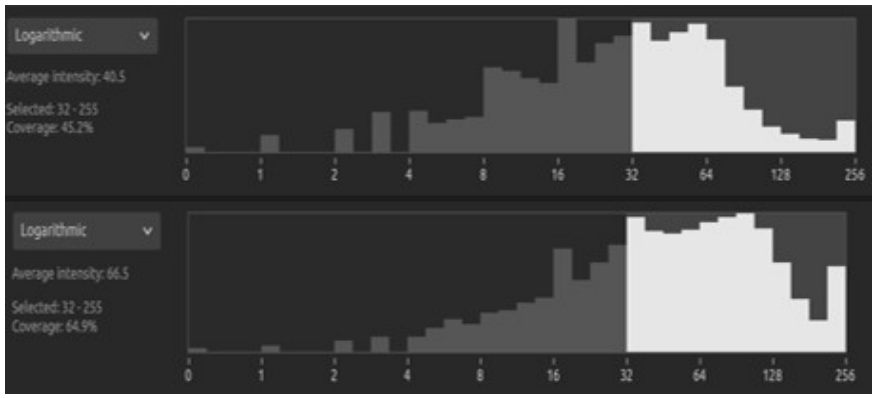


Figure 6.1.: Histogram, highlighted area with RGB values between 32-255, top) before adjustment, bottom) after adjustment

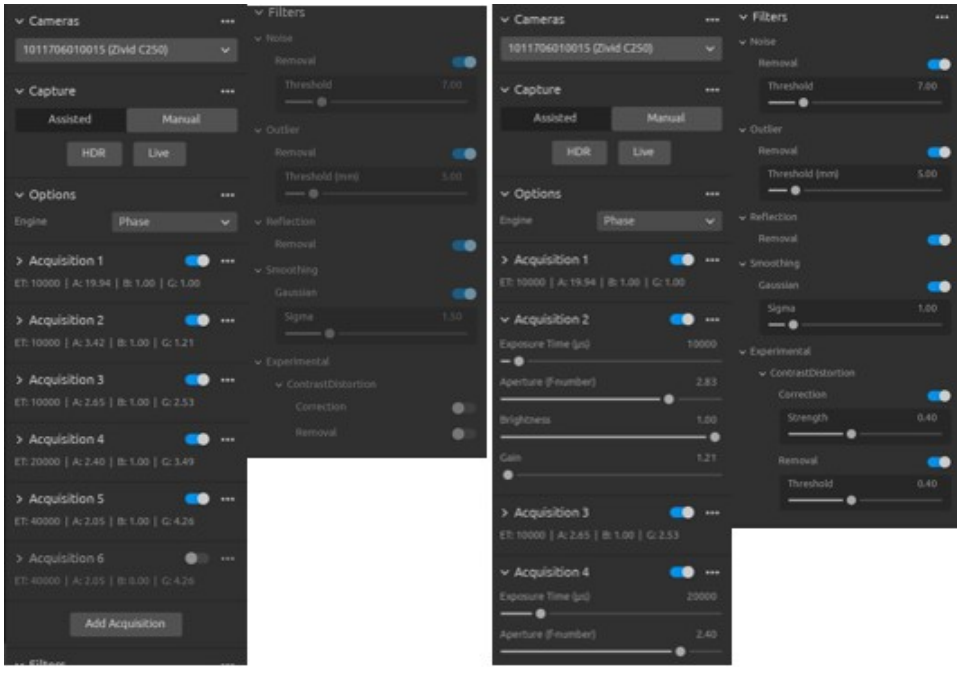


Figure 6.2.: Settings and filter, left) before, right) after

6.1.1. ZIVID/Python hand-eye calibration

The Zivid/Python calibration used 19 pose/acquisition pairs. The residuals are presented in Table 6.1 and ${}^{base}T_{cam}$ is shown in (6.2).

Table 6.1.: Zivid hand-eye calibration residuals

	Rotation [deg]	**Translation[mm]**
1	0.407087	3.77297
2	0.516402	5.22367
3	0.577084	3.31329
4	0.157297	2.75479
5	0.389591	2.63191
6	0.5704	3.07258
7	0.657906	4.68146
8	0.64584	3.38359
9	0.862091	5.00684
10	0.427363	3.42613
11	0.664082	2.32403
12	0.315225	3.34321
13	0.215805	3.11083
14	0.416563	2.00793
15	0.486436	3.76653
16	0.504935	0.929946
17	0.794865	2.44783
18	1.08323	6.34986
19	0.57691	1.93054
Average	0.54047958	3.340944

$$ {}^{base}T_{cam} = \begin{bmatrix} -0.044353 & -0.959480 & 0.278266 & 187.675507 \\ -0.987824 & 0.000544 & -0.155575 & 65.260567 \\ 0.149120 & -0.281778 & -0.947821 & 686.755127 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix} \tag{6.1} $$

6.1.2. HALCON 2D hand-eye calibration

The 2D calibration used 20 image/pose pair. The feature recognition result for the 2D calibration and the projected result after calibration is shown in Figure 6.3. The error is shown in Table 6.2 and ${}^{base}T_{cam}$ is shown in (6.2).

Table 6.2.: Result error 2D calibration HALCON

Error of the camera calibration:	0.1459 pixel	
Errors of the hand eye calibration:		
	RMS	Maximum
Translational part [deg]	6.951	15.961
Rotational part [mm]	0.782	1.473

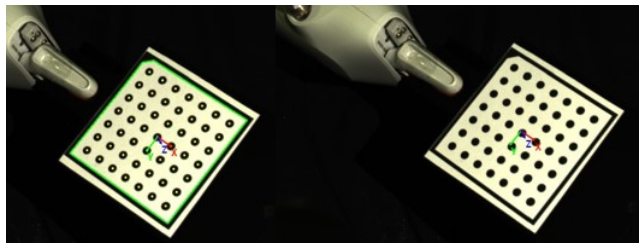


Figure 6.3.: HALCON feature recognition hand-eye

$$ {}^{base}T_{cam} = \begin{bmatrix} -0.0299376 & -0.967327 & 0.251757 & 170.63 \\ -0.993248 & 0.000546863 & -0.11601 & 71.1913 \\ 0.112082 & -0.253531 & -0.960812 & 721.406 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix} \tag{6.2} $$

6.1.3. HALCON 3D hand-eye calibration

The 3D calibration used 15 image/pose pair. The surface matching result for the 3D calibration and the projected result after calibration is shown in Figure 6.4. The error is shown in Table 6.3 and ${}^{base}T_{cam}$ is shown in Equation 6.3.

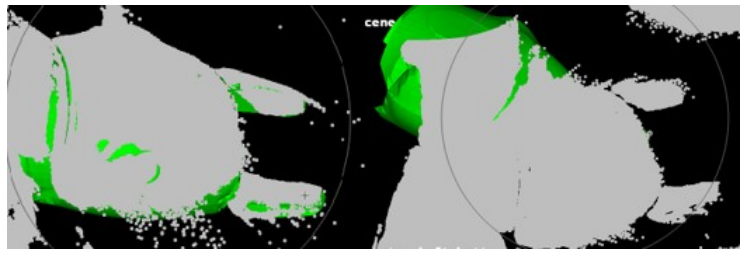


Figure 6.4.: Left) Surface matching result visualization HALCON right) projected result after calibration. Green = calib object

Table 6.3.: Result error 3D calibration HALCON

Errors of the hand-eye calibration:		
	RMS	Maximum
Translation part[mm]	62840.6446	85995.1531
Rotational part[deg]	69.7814	92.3038

$$ {}^{base}T_{cam} = \begin{bmatrix} 0.178182 & -0.820948 & 0.542491 & -276.394 \\ -0.224693 & 0.502802 & 0.834688 & -444.759 \\ -0.958 & -0.27062 & -0.0948703 & -13.1557 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix} \tag{6.3} $$

6.2. Target generation pipeline

To evaluate the target generation pipeline testing has been done for bins with all the parts. For each parts bin an image is captured and the pipeline is run until a target is generated, then the found target is removed if it is a good match. This is done until all the parts are picked, if there are instances where a false match is used to generate a target or if there are only false positive matches the parts are redistributed, i.e., the bin is shaken. As described in section 5.8 5 matches are found initially before they are returned, the number of good and bad matches are evaluated as well as if the procedure returns a good or bad match and if this match is obstructed making it impossible to pick. The results are plotted for all tested parts in the following sections, the blue arrow indicates that the parts have been redistributed.

6.2.1. Large female connector

The target generations pipeline was tested on a bin with 51 parts of the large female connector. The score distribution is presented in Figure 6.5, the curve approaches a bell curve indicating normal distribution, the average match score is approx. 8,74%. The matching results is shown in Figure 6.6. It can be seen that 67 runs and 3 part redistributions were needed to empty the bin. The matching procedure on average return approx. 2.9 good matches (57%) out of the 5 it is set to look for. It returns a good match 76% of the times, out of these 82% were unobstructed.

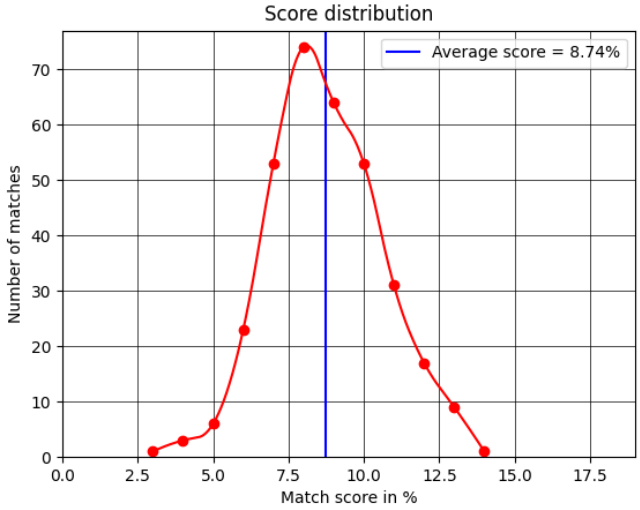


Figure 6.5.: Score distribution LFC, the match score % tells how many of the points from the CAD model of the object is found in the scene

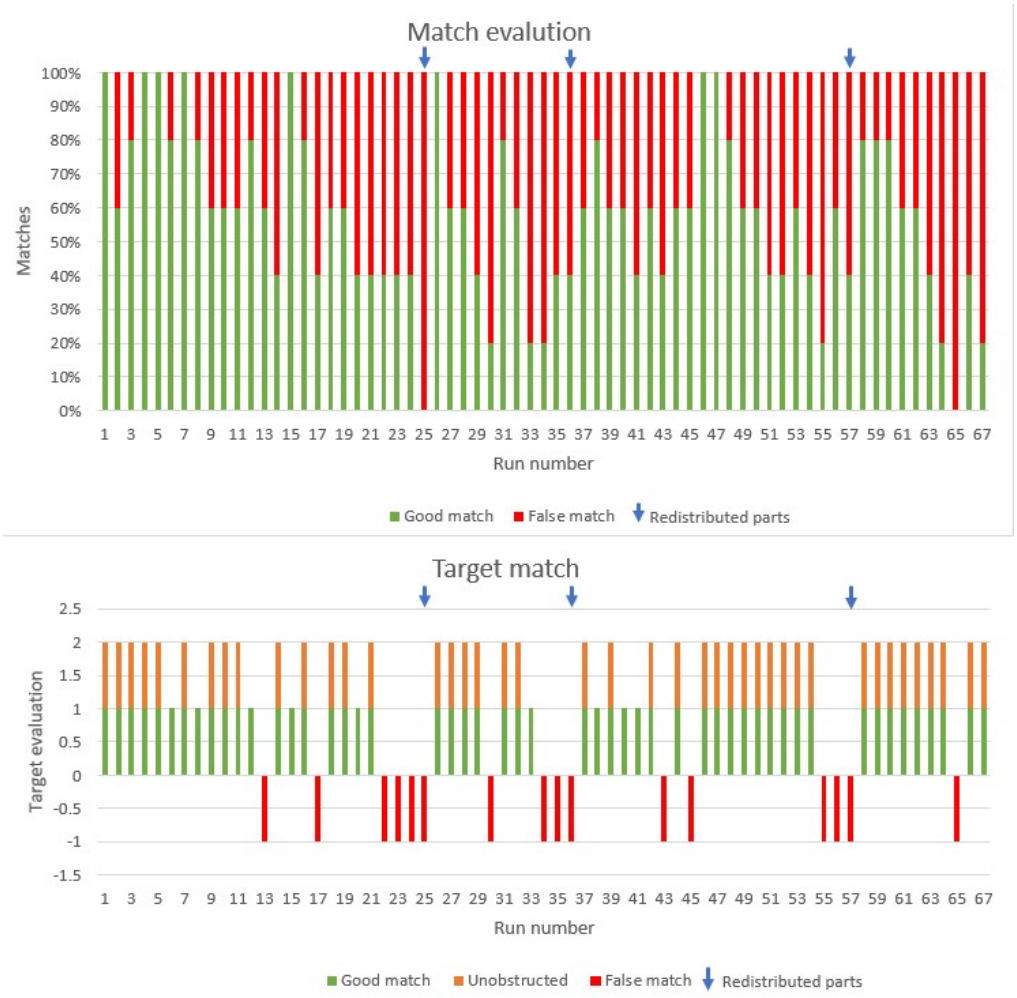


Figure 6.6.: Match results LFC, top) % of good matches from the 5 returned matches of surface matching procedure, bottom) Evaluating if the returned target is reachable, reachable and unobstructed or a false match

6.2.2. Small female connector

The target generations pipeline was tested on a bin with 62 parts of the small female connector. The score distribution is presented in Figure 6.7, the curve approaches a bell curve indicating normal distribution, the average match score is approx. 8,08%. The matching results is shown in Figure 6.8. It can be seen that 73 runs and 11 part redistributions were needed to empty the bin. The matching procedure on average return approx. 3.1 good matches (63%) out of the 5 it is set to look for. It returns a good match 82% of the times, out of these 83% were

unobstructed.

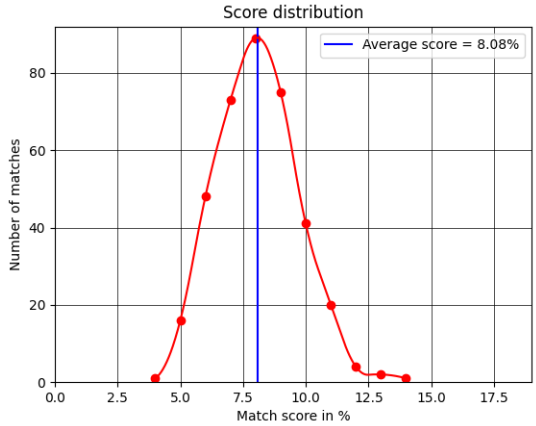


Figure 6.7.: Score distribution SFC, the match score % tells how many of the points from the CAD model of the object is found in the scene

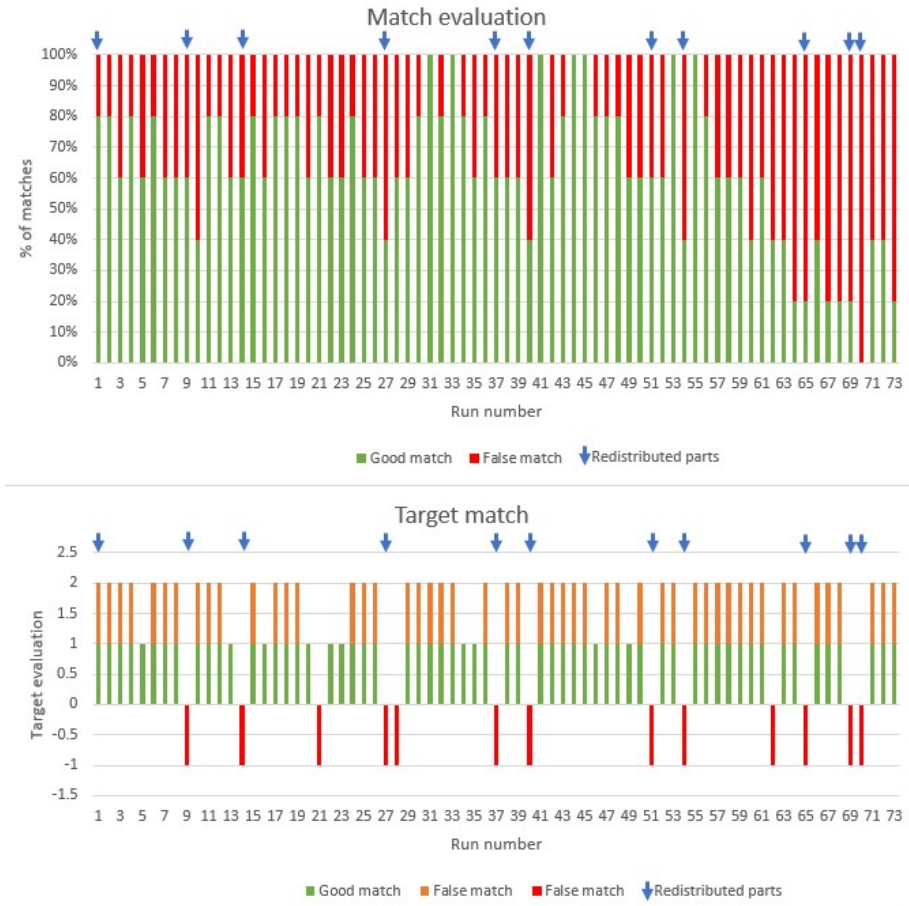


Figure 6.8.: Match results SFC, top) % of good matches from the 5 returned matches of surface matching procedure, bottom) Evaluating if the returned target is reachable, reachable and unobstructed or a false match

6.2.3. Brass connector

The target generations pipeline was tested on a bin with 5 parts of the brass connector. The score distribution is presented in Figure 6.9, the average match score is approx. 8,98%. The matching results is shown in Figure 6.10. It can be seen that 11 runs and 6 part redistributions were needed to empty the bin. The matching procedure on average return approx. 1.3 good matches (27%) out of the 5 it is set to look for. It returns a good match 45% of the times, out of these all were unobstructed.

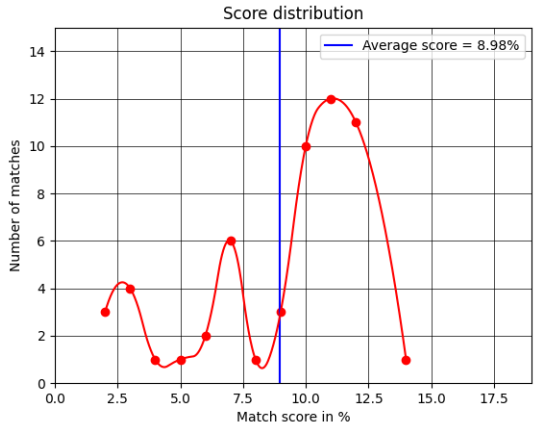


Figure 6.9.: Score distribution brass connector, the match score % tells how many of the points from the CAD model of the object is found in the scene

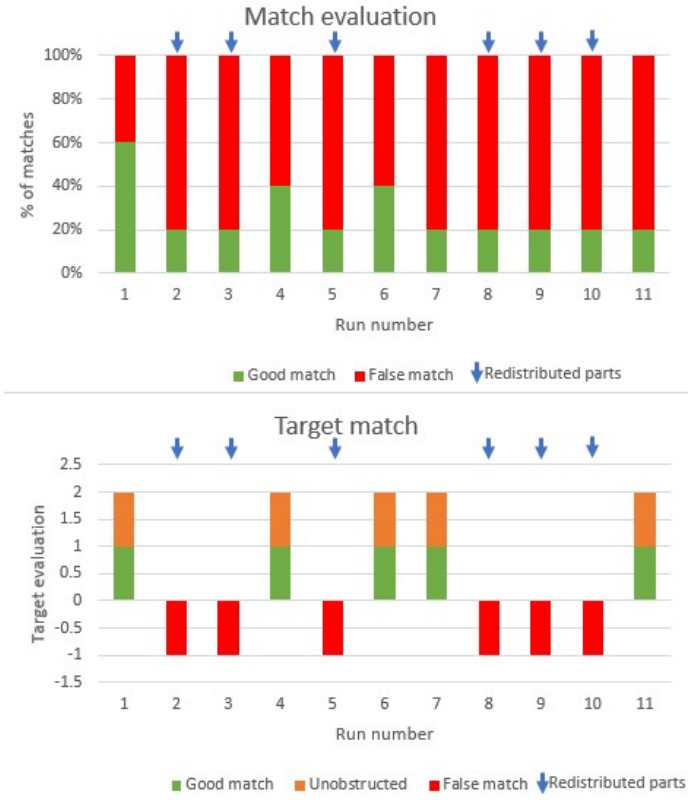


Figure 6.10.: Match results brass connector, top) % of good matches from the 5 returned matches of surface matching procedure, bottom) Evaluating if the returned target is reachable, reachable and unobstructed or a false match

6.2.4. Large lask

The target generations pipeline was tested on a bin with 4 parts of the large lask. The score distribution is presented in Figure 6.11, the average match score is approx. 24.94%. The matching results is shown in Figure 6.12. It can be seen that 8 runs and 3 part redistributions were needed to empty the bin. The matching procedure on average return approx. 1.25 good matches (25%) out of the 5 it is set to look for. It returns a good match 63% of the times, out of these all were unobstructed.

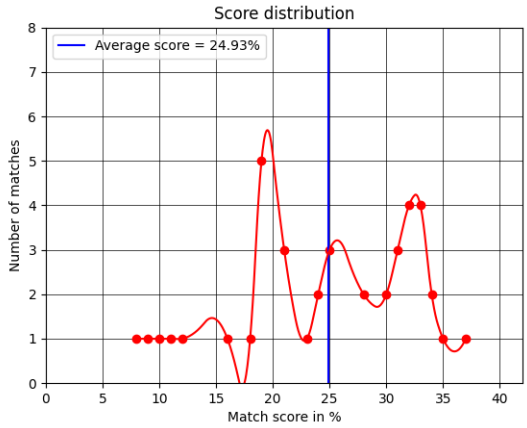


Figure 6.11.: Score distribution large lask, the match score % tells how many of the points from the CAD model of the object is found in the scene

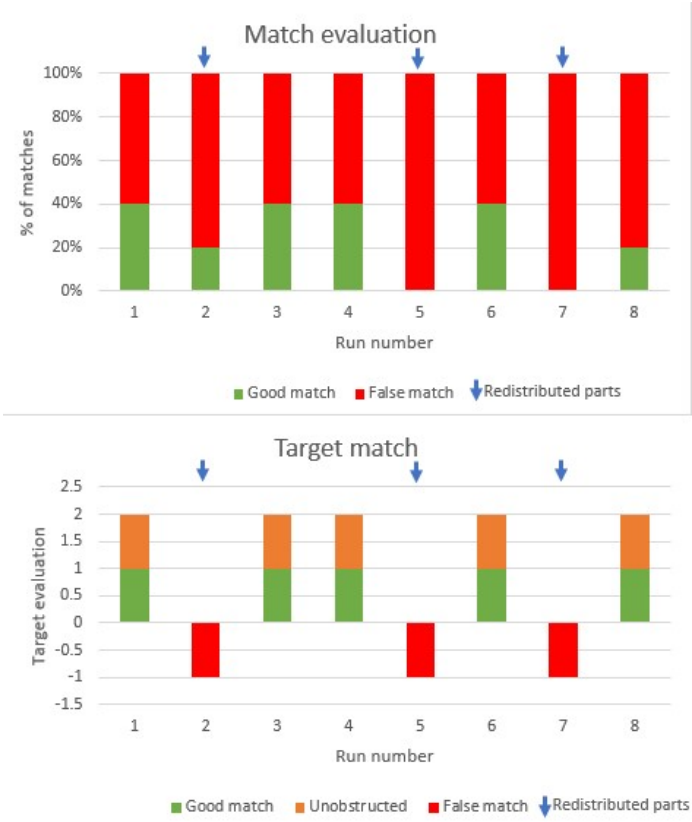


Figure 6.12.: Match results large lask, top) top) % of good matches from the 5 returned matches of surface matching procedure, bottom) Evaluating if the returned target is reachable, reachable and unobstructed or a false match

6.2.5. Small lask

The target generations pipeline was tested on a bin with 7 parts of the small lask. The score distribution is presented in Figure 6.13, The average match score is approx. 25.09%. The matching results is shown in Figure 6.14. It can be seen that 11 runs and 4 part redistributions were needed to empty the bin. The matching procedure on average return approx. 1.4 good matches (27%) out of the 5 it is set to look for. It returns a good match 72% of the times, out of these 88% were unobstructed.

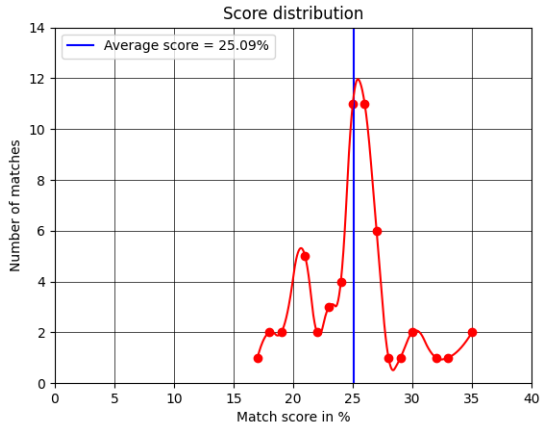


Figure 6.13.: Score distribution small lask, the match score % tells how many of the points from the CAD model of the object is found in the scene

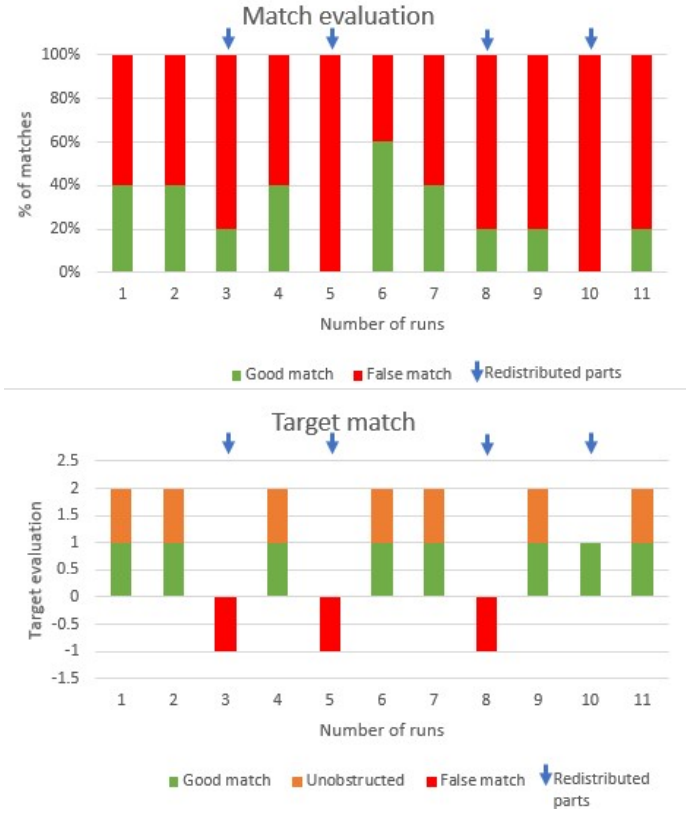


Figure 6.14.: Match results small lask, top) % of good matches from the 5 returned matches of surface matching procedure, bottom) Evaluating if the returned target is reachable, reachable and unobstructed or a false match

6.3. Gripper

6.3.1. Custom fingers

The custom printed fingers where tested for stiffness by holding the LFC, the holding force used is 20N for all 3 thicknesses and the 4mm thickness was also tested with 15N holding force. The reason for this test is to check how much the gripper bend while holding because this effects the stop position for the servo motors in the grippers which is specified in the `g_GripIn` command. If the end position is not within the tolerance because of flexing it would result in the error that tells the system the part is not picked. It would also increase the strain on the parts which increases the risk of failure for the fingers. The resulting flex is presented in Figure 6.15.

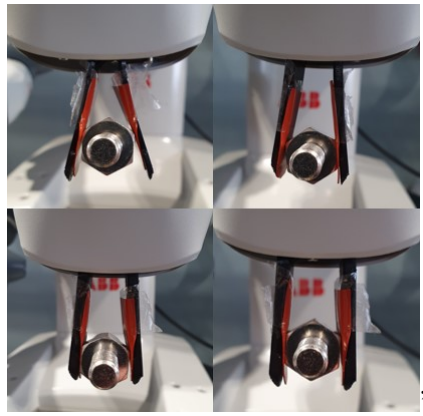


Figure 6.15.: Finger flex, top left) 2mm, top right) 3mm, bottom left) 4mm, bottom right)4mm 15N

Figure 6.16.: Vacuum gripper, from left:large lask, brass connector, elongated lask, triangle lask

6.3.2. Vacuum gripper

The vacuum gripper where tested to check which parts it could pick with regards to holding force and ability to get the necessary airtight seal. The result can be seen in Figure 6.16, it was able to pick the large lask, elongated lask, triangle lask and the brass connector. It was not able to pick the large and small female connector, nuts, washers and small lask.

6.4. System time

To evaluate the system time 10 runs where done on the LFC. The time used for the acquisition, .ply conversion, read the file, crop, matching, evaluation and target definitions were recorded. The rounded average of these runs are presented in the pie chart in Figure 6.17, the total time used is 11.15s. Acquisition, .ply conversion and reading of the file account for 93% of the time used. Time for

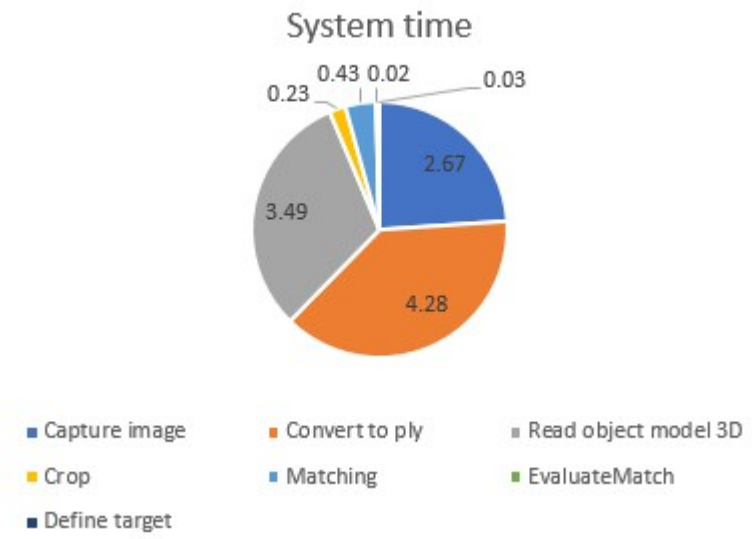


Figure 6.17.: Pie chart representation of the system time from acquisition to defined target

robot motion was not included in this evaluation.

6.5. Picking

To evaluate the performance of the system it was tested on each part by having 10 runs were the objective was to pick until failure. Number of picks and reason of failure were recorded. This was done to evaluate the applied picking rules and the ability to pick continuously. After each failure the parts were put back in the bin. The grip verification procedure described in subsection 5.5.4 was not used in the test and the parts in the foam cutout was only tested once. The failure causes are robot error which could be because of wrong `confdata` or the moving in wrong direction error shown in Figure 6.29. False match from the surface matching procedure, the robot cannot get a good grip or it drops the part, and a collision with neighboring parts or the bin. The result is presented graphically in Figure 6.18-Figure 6.22. Images of various picks are show in Figure 6.23-Figure 6.28.

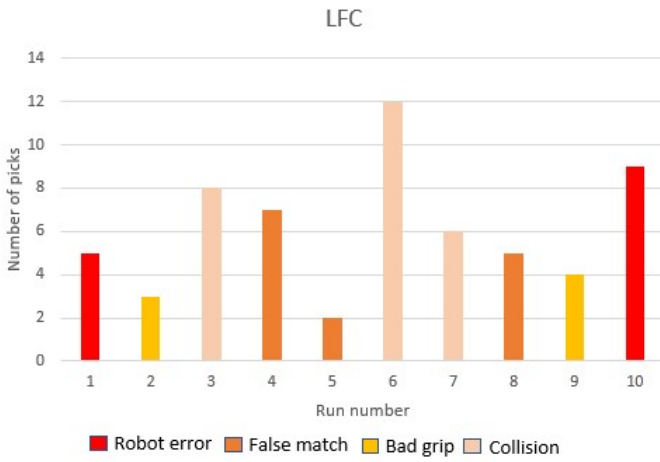


Figure 6.18.: Evaluations after 10 runs picking to failure with 51 LFCs in a bin using the custom 4mm gripper fingers. The resulting failure causes are robot error = joint limits or moving in the wrong direction, false match, bad grip, and collision with bin or neighbouring parts

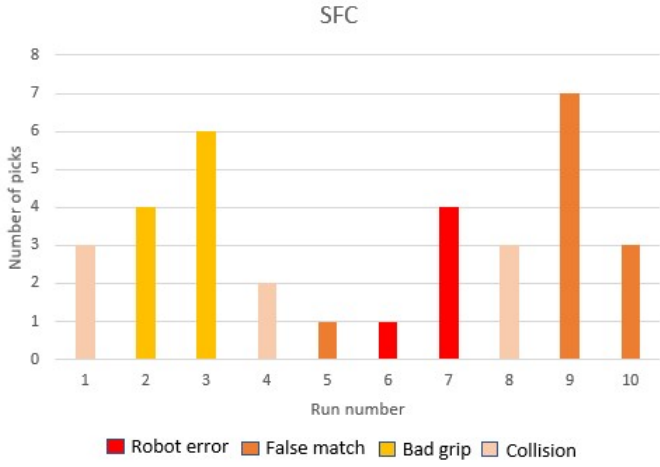


Figure 6.19.: Evaluations after 10 runs picking to failure with 62 SFCs in a bin using the custom 4mm gripper fingers. The resulting failure causes are robot error = joint limits or moving in the wrong direction, false match, bad grip, and collision with bin or neighbouring parts

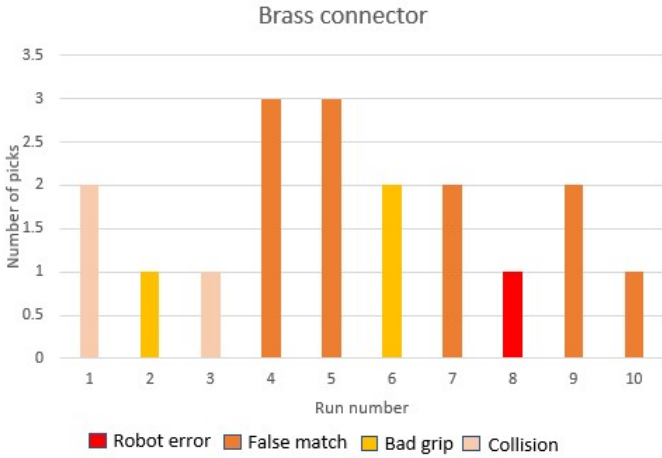


Figure 6.20.: Evaluations after 10 runs picking to failure with 5 brass connectors in a bin using the custom 4mm gripper fingers. The resulting failure causes are robot error = joint limits or moving in the wrong direction, false match, bad grip, and collision with bin or neighbouring parts

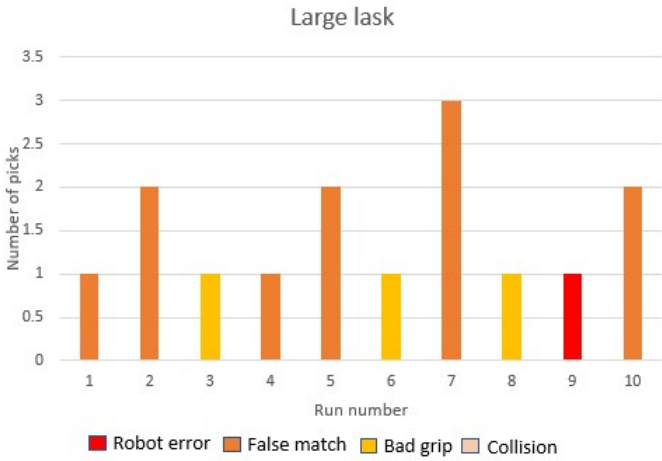


Figure 6.21.: Evaluations after 10 runs picking to failure with 4 large lasks in a bin using the custom 4mm gripper fingers. The resulting failure causes are robot error = joint limits or moving in the wrong direction, false match, bad grip, and collision with bin or neighbouring parts

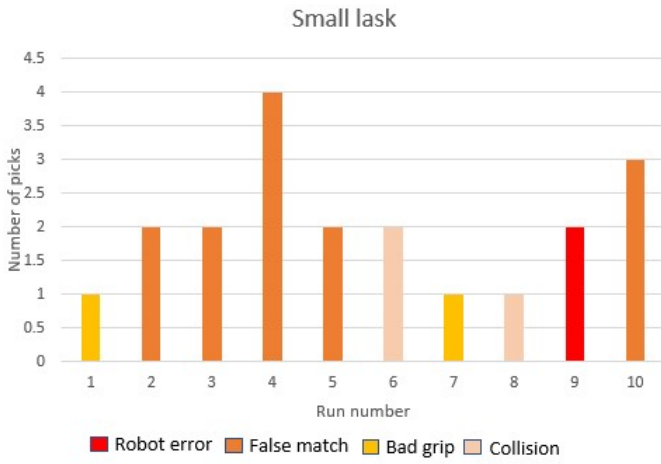


Figure 6.22.: Evaluations after 10 runs picking to failure with 7 small lasks in a bin using the custom 4mm gripper fingers. The resulting failure causes are robot error = joint limits or moving in the wrong direction, false match, bad grip, and collision with bin or neighbouring parts

Figure 6.23.: Picking LFC at the threaded end. left) approach above pick, middle) gripping, right) picking the part

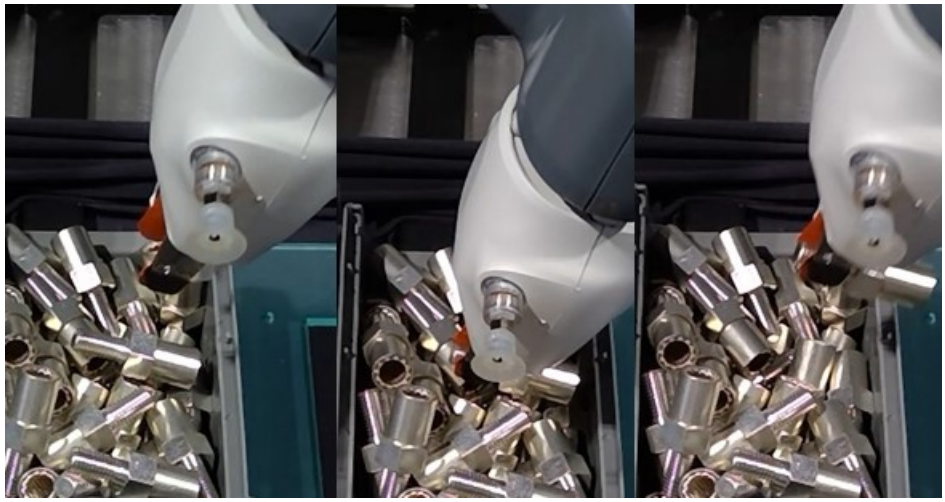


Figure 6.24.: Picking LFC at the threaded end. left) approach above pick, middle) gripping, right) picking the part

Figure 6.25.: Picking brass connector by the outer ring. left) approach above pick, middle) gripping, right) picking the part

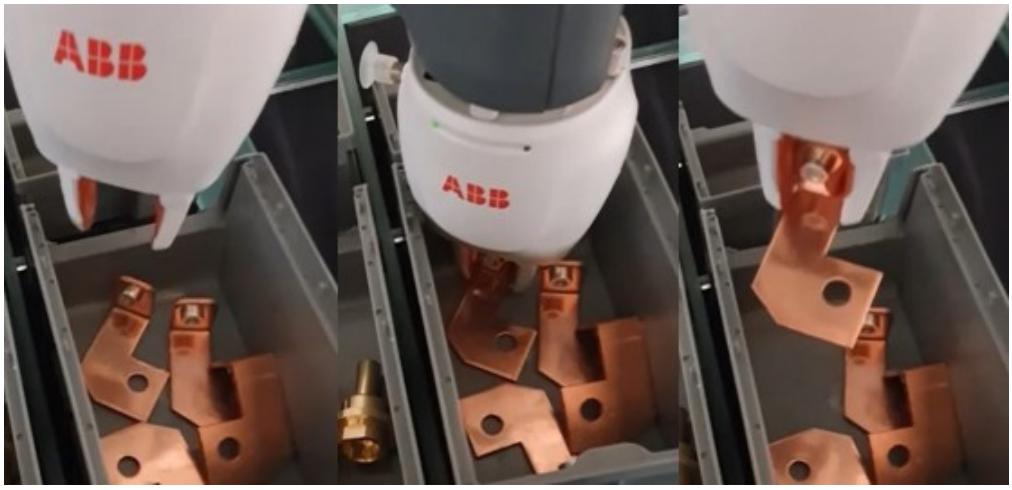


Figure 6.26.: Picking the large lask by the rules described in scenario 2. left) approach above pick, middle) gripping, right) picking the part

Figure 6.27.: Picking the elongated lask from the foam cutout. left) approach above pick, middle) gripping, right) picking the part

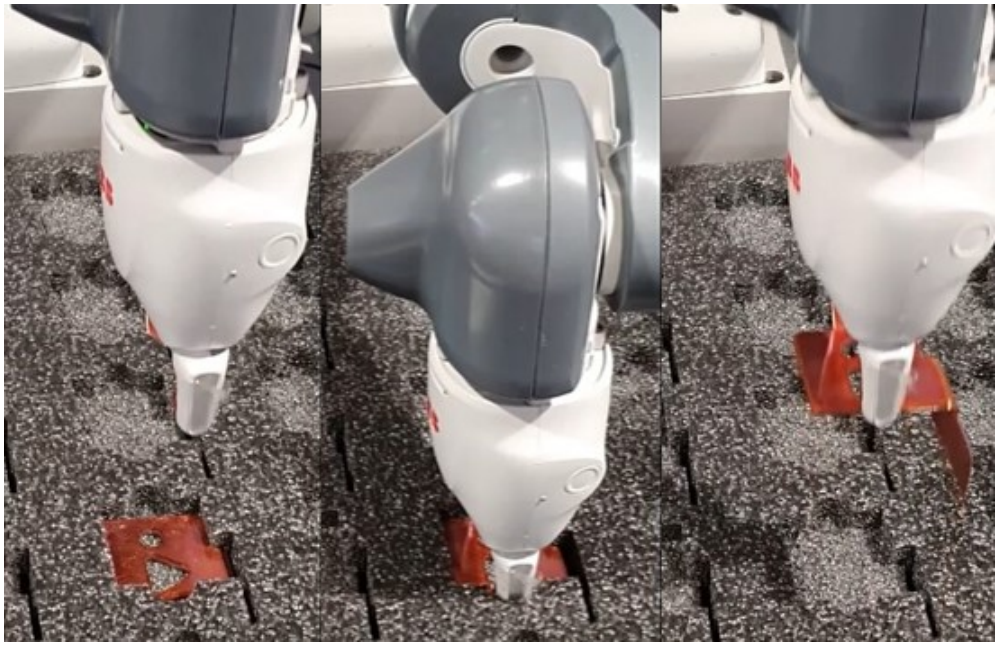


Figure 6.28.: Picking the triangle lask from the foam cutout. left) approach above pick, middle) gripping, right) picking the part

6.5.1. Errors

This section shows common errors experienced when picking parts. These are shown in Figure 6.29 - Figure 6.31.

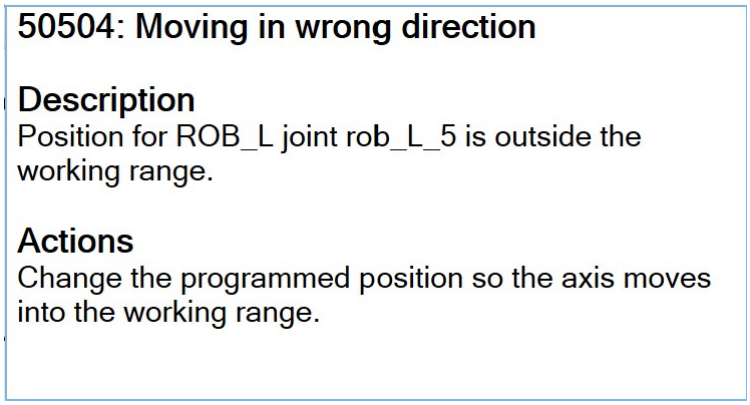


Figure 6.29.: Moving in wrong direction error, raised after picking the part trying to return to the abovePick pose.

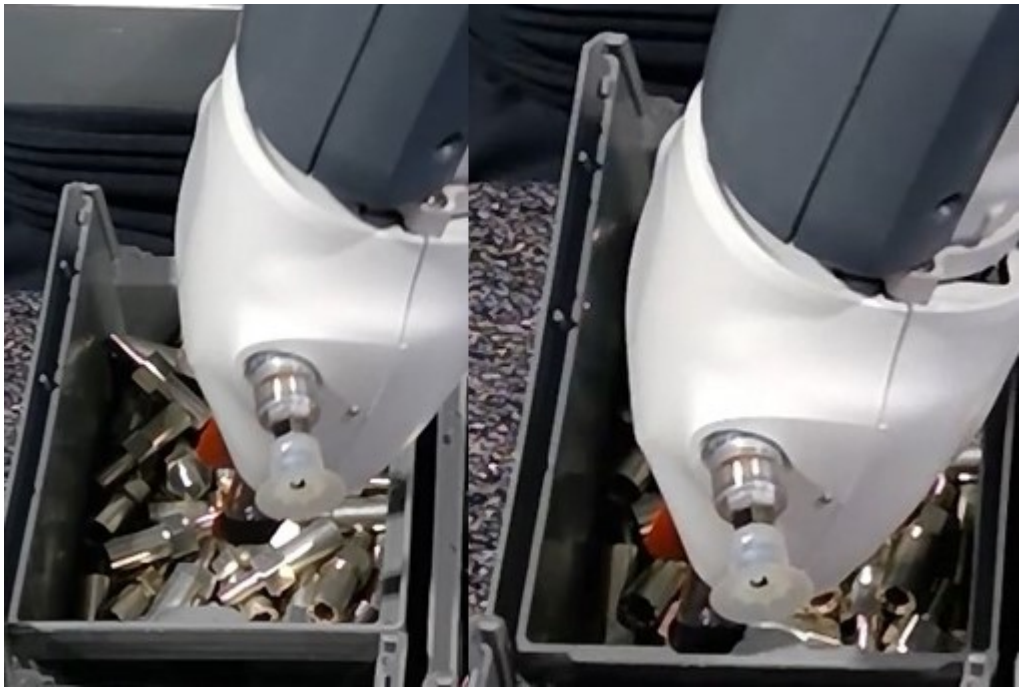


Figure 6.30.: Gripper fingers colliding whit bin because the part is to close to the wall causing joint limits. left) approach, right) collision

Figure 6.31.: Errors while picking lasks from the foam cutout where the parts are not released from the cutout. left) Elongated lask, right) triangle lask

Chapter 7.

Discussion

7.1. Acquisition settings

The acquisition settings used in section 6.1 are based on the settings from Zivids analyze and capture procedure, then adjusted to get the best results when dealing with shiny objects. Testing showed that the analyze and capture procedure changed the settings for different days and time of the day. The change could be from simple brightness and aperture to adding additional acquisitions. A probable cause of these changes is the natural lighting entering through the windows in the lab. Testing with an enclosure around the camera resulting in more homogenous settings throughout the day and over several days.

All the parts worked with in this project can be considered shiny, which in image acquisition and especially, when using an active stereo vision camera, requires additional consideration. Common problems are oversaturation caused by direct reflections from the projector into the image sensor, this can result in contrast distortions. This was handled using the settings and filters described in section 4.1 and following the procedure described in subsection 5.1.7, however there are some challenges that cannot be solved by this approach. The orientation of the parts impacts the severity of contrast distortion as described in section 2.4. In random bin picking the part orientation cannot be altered, so this effect is a constant issue. One approach to deal with this could be to create a filter that disregard matches that align perpendicular within a certain tolerance to the camera baseline.

7.1.1. Point cloud reflection error

The point cloud evaluation filter presented in subsection 5.3.2 removes erroneous acquisitions with "ghost planes" caused by reflections. These errors occurred randomly and in approx. 1/15 acquisitions. After discussion with a representative from Zivid it was concluded that the errors should be handled by the reflection

filter described in subsection 4.1.2, but for some reason were not. To establish the cause of the error an enclosure was used to block out light from the lab. After running 100 acquisitions in the enclosure no errors occurred and it can therefore be assumed that one of the ceiling lights in the lab or a change in the natural lighting was the cause. If this was a lab specific incident or related to using an older camera must be confirmed by testing the system in a different environment, but the key point is that this error should not occur, and it should therefore not necessary to enclose the system. The consequence of this error is show in Figure 7.1, the floating points in the image go towards the lens of the camera, i.e., the z-axis.

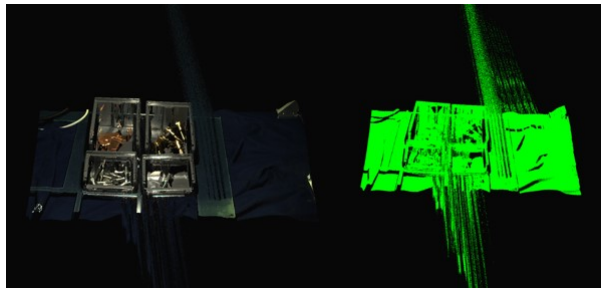


Figure 7.1.: Point cloud reflection error

7.1.2. Acquisition settings result

As can be seen from the result after using the analyze and capture feature there are few black areas without any points where the SNR values are below 7. These errors occur near highlights and on the side of parts with circular cross sections. Only 42.5 % of image has RGB values between 32-255, this is not optimal as this is a prerequisite for good data 3D data on the pixels. Evaluating the comparison between the initial image and the one where the projector is blocked the highlights does not disappear, so it can be concluded that they are not caused by the projector alone.

The first acquisition, the highlight acquisition, did not need be to adjusted as it showed a good result. Inspecting the RGB values by pixels after the second acquisition it can be seen that majority of the lower values, i.e. less than 32, are from the surroundings surfaces in the scene and not in the bins and on the parts. This is an important factor when evaluating the image with the histogram as it represents the whole scene and not the ROIs. So, a comprise was made in the second acquisition to get more pixels above 32 at the cost of a 2.6% increase in pixels in the 216-255 range. Following Zivids recommendation this was seen as the best solution. The final acquisitions were adjusted and the resulting adjustments showed 19.6% more of the image now resides in the range of values that Zivids

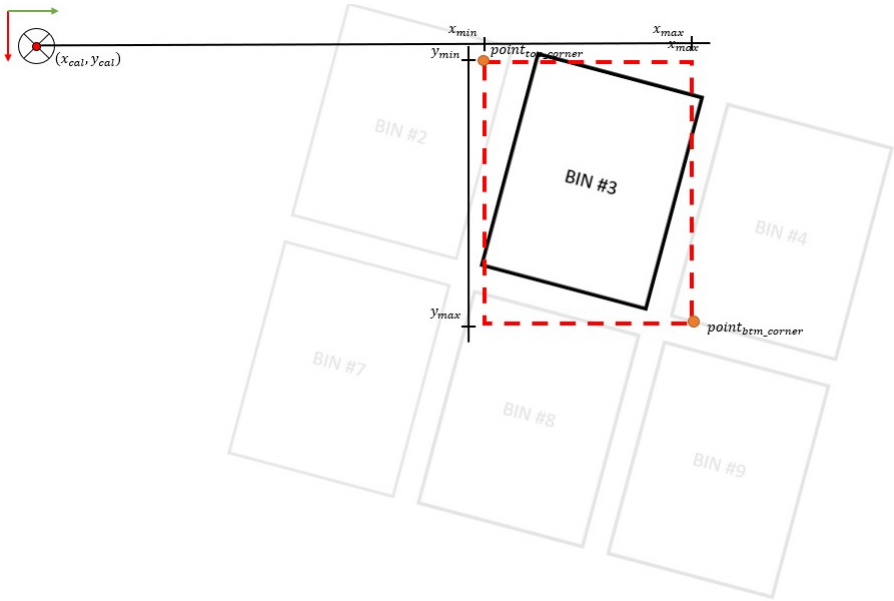


Figure 7.2.: Crop based on fixed location, not aligned bins

deems to give good 3D data. The gaussian smoothing filter was adjusted to 1.0 to preserve as many details in the scene as possible because the parts have small features, this was done per zivids recommendation [74]. The contrast distortion filter was also implemented with correction and removal, as it seen as better to remove erroneous points to reduce the risk of false matches. The Zivid One camera used does not feature the ability to change vision engine to stripe engine so the phase engine was used.

7.1.3. Define ROI

As described in subsection 5.3.3 two possible solution to defining the ROI was developed, cropped based on fixed locations was used. The reason for this was the simplicity and correlated speed of this solution, also since it was written in HALCON it was easier to implement. This solution worked well for this proof of concept and would be usable in future implementation of the system, but there are some drawbacks. Mainly, the HALCON procedure `select_points_object_model_3d()` that was used only allows for cropping in the principle axis direction, meaning the point cloud cannot be cropped diagonally. So for the system to work the bins must be aligned with the x and y axis of the camera. Figure 7.2 illustrated the consequences of not having the bins aligned.

The ROI is defined inside the bin walls to ensure no false matches with the walls

of the bin. Testing showed that this was a reoccurring problem. This does mean that some parts may not be visible for the matching procedure, but if there is a certain minimum of parts in the bin this should not be a problem.

ROI foam cutout

Running matching procedure on the parts in the foam cutout did require the ROI to be defined with a certain degree of precision because if too much of the flat surface surrounding the parts were included in the point cloud the matching procedure would in some cases return a false match. With the placement being constant it will not be a problem to implement that in the procedure.

7.2. Hand-eye Calibration

The resulting calibration from a previous thesis, with the same robot and camera, was used in the initial phase. Testing showed that the accuracy was not good enough, it required a larger gripper width to pick the parts and it was therefore decided to do a new calibration. The reason for choosing different approaches in the calibration was to have more data to evaluate with and because this system requires precision having more methods to confirm the result is important. The chosen calibration for the system was the one developed by Zivid. It was chosen because the evaluation metrics where better.

7.2.1. Calibration Zivid/Python

As mentioned in the description of the procedure it returns rotations and translations residuals to evaluate the score. The average score was 0.54 deg in rotation and 3.34 mm in translation, comparing this to the Zivid performance graph in Figure 5.7 and Figure 5.8, it can be seen that the obtained rotation error average is 0.5 deg bellow the 1 deg score the graph converges to. The average translation error is approx. 2 mm above the 1.5 mm translation error the graph converges to. From this it can be concluded that the result is precise, but there is potential for improvements. This was not prioritized as the result was good enough to get adequate precision on the targets.

7.2.2. 2D Calibration HALCON

The 2D calibration plate used in the calibration was predetermined by HALCON. The feature recognition result was visually inspected for each capture and a similar match with the correct orientation to Figure 6.3 was found for each image.

The projected result after calibration confirmed visually that there were no major errors in the calibration. The result from the procedure was good in terms of rotational error but the translational error were significant. To improve the procedure attempts where made to use more image/pose pairs but this did not show significant improvement.

7.2.3. 3D Calibration HALCON

The 3D calibration object used in the 3D calibration was the hand of the robot. This was chosen because this was what HALCON used in their demonstration of this calibration and because after researching the topic no better objects where found. A visual inspection of the surface matching result of the 3D calibration concluded that a correct match was found for each image with what visually looked like a good match. The calibration result shows significant error, an "inconsistent pose pair" error occurs for several of the poses. To handle this the inconsistent pose pairs where removed and program was restarted. This did improve the result, but it was still significantly worse then the other two. Another attempt to improve the procedure was done with new poses and acquisition but it gave similar result. Given that the two other procedures gave significantly better results a decision was made to not spend any more time improving this procedure.

7.3. Surface matching

7.3.1. Surface matching settings

The surface matching settings are based on a lot of trial and error trying to get good matches within a reasonable time frame. By returning 5 matches the theory is that this should result in several good matches and that they can be further evaluated to return the best one. Different number of matches where considered, but initial testing showed that the percentage of good matches did not improve with fewer or more parts.

When the parts bin is getting close to empty the occurrence of false matches increases. These false matches are found between the flat bottom of the bin and flat surfaces of the objects. A simple solution to this problem would be to refill the bins before they reach this state, this was discussed and proposed by Siemens. It still is a challenge of how the system could notify that it needs to refilled, with an estimate of the cycle time this refilling could be solved by following a time schedule or it could be filled it up every morning. A system solution could be to implement a counter that notifies the operator after a set amount of picks that a refilling is needed. If there is a desire to have the robot pick the box until it is completely empty then a having a rugged surface in the bottom of the bin could

possibly help reducing the amount of false matches. Choosing a black rubber surface should also be used to reduce reflection from the bottom of the bin when capturing images.

7.3.2. Surface matching results

The result of the surface matching pipeline shows that it is not perfect, i.e., it is most likely not possible to complete a full tray without implementing error handlers. The most common error is that a false positive match is returned even though good matches are found. Observations of the returned matches have shown that the return of a false match often occurs when most of the parts in the bin have below average scores. A handler for this could be to first check the score for each part, if it is below the already calculate average the parts in the bin should be redistributed.

Of the returned good matches for the LFC and SFC approx. 80% of them are unobstructed, i.e. there are no parts covering the returned match. During the testing there where a total of 3 cases for all part bins where the returned match was too low in the bin to be picked, further testing is needed to understand why these matches where returned. It might be the case they were the best match of the bunch or that the match evaluation procedure described in subsection 5.4.2 needs improvement. In the other cases the returned match was partly covered by another part causing a collision with gripper before it can reach the target. An evaluation procedure for this match could be to check if there are points in the point cloud above the found match, this would give an indication if there were parts obstructing it, if it were another match could be chosen.

The score distribution plots for the LFC and SFC approaches a bell cure with an average score of 8.74% and 8.08% respectively. This result can be used to define a threshold and evaluate the returned matches in the future. The rest of the score distribution plots for the other parts are difficult to take any real data from because the sampling size is small.

An issue with the testing is that there were few parts available of the brass connector, large lask and small lask, making it difficult to draw any conclusions based on the data obtained. The testing could have been repeated to get a more data, but this would give an inaccurate description of the system because the bins would always be close to empty. Due to the flat surface areas of the small and large lask, the matching procedure often finds a match with the bottom of the bin when there are few parts available. This was verified after reducing the ROI to only see the parts, which improved the result significantly. This indicates that with more parts in the bin the result will be better.

Another error, that is a lot less common, is the return of only false matches even though there are good matches in the scene. Most occurrences of this error have been when there are few parts in the bin, but there also exceptions. At run 25 on the LFC the returned matches where all bad even though there are 25+ parts left in the bin. This is seen as an anomaly at this point, further testing is needed to verify its relevance. A simple solution to the problem of false matches when there are few parts left if to refill it before the bin is empty. There is, at the current system requirements, no need to completely empty the bins.

7.4. Improvements

7.4.1. Camera upgrades

The Zivid camera used in this system is the Zivid One and was obtained by NTNU to be used in research. Since this camera was introduced Zivid has released and updated version called Zivid One + which comes in three variations, small, medium and large and a second-generation camera called Zivid Two. The zivid two camera is small and lightweight(880g) camera that is ideal for eye-to-hand configurations. Compared to the Zivid One + it has decreased the acquisition time from 85ms to 60ms and the trueness error from <1% to <0.2% which result in a 10x more accurate hand-eye calibration. In addition the new cameras comes with a new feature called Artifact Reduction Technology (ART) which helps remove noise from point clouds of shiny object, which has been an issue. Furthermore the new cameras has the additional stripe engine which is well suited for shiny objects [90].

7.4.2. Software upgrade

The HALCON version used in this system is 18.11 Steady, MvTec have had biannually releases of HALCON each May, and November and the current newest version is 21.05. The surface based matching procedure have been improved three times [91] compared to the version used in this system. The 19.05 version had improvements making it more robust against noisy point clouds by allowing user to control the impact of surface and edge information via multiple min-scores [92], the 20.05 version had improvements in the use of features for matching making it more robust [93], the 20.11 version had improvements in speed and edge-supported surface based matching [94]. The newer version of HALCON supports python. This does suggest that upgrading HALCON to a newer version would improve the matching and with python make it easier to use. It would also make the python script developed for defining ROI easy to implement.

7.4.3. Robot

The YuMi robot has 7-DOF, which generally is more difficult to program, but since ABB solves the kinematics, this was not an issue in this project. Having not worked with 6-DOF robots it is difficult to conclude how they would perform in this system. However, 6-DOF robots are used in various bin picking applications, based on that the assumption is that this will also work.

Based on the configuration change and Siemens preference for KUKA robots, KUKAs AGILUS or LBR iiwa series would be the recommended robot solution for this application. Both series features robots with a load capacity > 3kg and good versatility. They offer the required performance and precision, as well as flexibility in mounting which offers possibilities in further optimizing the robot cell. There is difference between the series, the LBR iiwa has greater load capacity, reach, precision and it has 7-DOF while the AGILUS has 6-DOF, these advantages does come with a higher price tag. The price of KUKA KR AGILUs 23,000-33,000€ + MVA and the price of KUKA LR iiwa is 70,000 – 100,000€ + MVA

7.4.4. Configuration eye-on-hand

As stated in the introduction, the setup with the yumi robot and the zivid camera in a fixed position was predetermined. The main limitations was the size of the working space, which was determined by the FOV of the camera. Based on the experienced gained from this project the recommendation would be to alter the configuration to eye-on-hand and the change the robot to a more conventional single arm configuration with a greater load capacity. This would greatly increase the workingspace as the robot reach would be the limit for what the camera can see. Another advantage to this configuration is that the camera can be placed in the optimal position to get the best images and the placement of the parts can be verified using the same camera. These advantages are illustrated in Figure 7.3

7.5. Picking

7.5.1. Picking evaluation

The robot's ability to pick the different parts were decent, the picking rules implemented proved to be versatile, they worked regardless of the part orientation creating the correct target. With the LFC and SFC an additional target transformation was implemented to move the target to the threaded end. This was done to reduce the risk of collision between the gripper fingers and the neighboring parts because here the cross section is smallest which leaves more clear air

Figure 7.3.: Benefits of eye-on-hand configuration, robot reach defines workspace and the camera can be moved to the optimal position [95]

between the parts. This was confirmed by visually inspecting the bin and through testing.

The evaluation done on the picking in showed that it was not able to pick continuously, as expected after evaluating the surface matching procedure. The best the data is from testing with the LFC and SFC because here the bins are full. It is this result that can be used to give an indication on how the system performs. The system managed 13 and 8 picks respectively before failure. The were no common errors for these parts, the collisions were caused when the gripper hit neighboring parts and the false matches were the same errors experienced when testing the matching procedure. The robot error were the moving in wrong direction error shown in Figure 6.29. Bad grips were when the part slipped out of the gripper when moved, most likely caused by the flex in the custom fingers.

The number of parts available when testing with brass connector, small lask and large lask must be taken into consideration. At least 50% of the failures for these parts where due to false matches, the collision failures where between the hand and the bin wall. These failures are caused because the matching procedures find false matches with the bottom of the bin and the gripper fingers are not long enough to reach the bottom without having the whole gripper hand in the bin. Subsequent testing on a flat surface with precision crop of the ROI showed significant improvements, indicating that a fuller bin and longer would reduce these problems. The cause of bad grips for the small and large lask were that the parts were picked at slight angle causing them to rotate and slip out of the fingers. This was partly due to the vulcanized rubber slipping in the fingers because it did not attach properly. The decision to pick the brass connector at the outer ring in the cases were it was oriented close to parallel with the xy base plane of the robot worked. The grip on the part was decent and risk of collision with neighboring

parts were reduced. However,the bad grip failure was due to strength of the grip, it was not strong enough if the parts were partly covered by neighboring parts.

When picking parts from the foam cutout there were only a few parts available. This does not make a difference for the elongated lask as they are placed separately, but for the triangle lask which is stacked vertically. Testing with two parts stacked showed that it could pick them, but further testing should be done on full stacks.

7.5.2. Picking errors

There were some errors and challenges that occurred when picking the parts. In some cases, the gripper would collide with the bin if the target parts were too close to wall or the angle of attack where to steep. As can be seen in Figure 6.30 the gripper fingers hit the top of the bin. Another errors is the "moving in wrong direction" shown in Figure 6.29. This occurred when the robot moved from the pick pose to the above pick pose, this may be caused because `MoveL` is used between these poses. Some testing were done using `MoveJ` and the error did not occur, but further testing is needed to verify this.

When picking parts from the foam cutout an issue that did occur was that the cutout came up with the parts when picking the last one, as shown in Figure 6.31. This happened when no external force was used to hold the cutout in place. Therefore, some way of fastening/holding is needed to ensure clean picks. With the elongated lask there was also issue with part slipping to some degree when held by the gripper. This was due to its design, it needed to be picked at one end creating a moment at the other end. This makes it difficult to place it directly after picking, an intermediate step needs to be defined before it can be placed.

7.5.3. Stock grippers

The stock grippers were proven capable of picking various parts, while only requiring a slight modification. Having a gripper with finger gives good versatility and it is a good starting point for testing a bin picking system. The main problem with the stock fingers is the thickness, when picking from a bin the fingers collided with neighboring parts when moving to the target as shown in Figure 7.4, this resulted in joint limitations due to too much torque in the motors. There are different approaches that can be used to overcome this problem, one could be to implement functionality that monitors the joints and stops the problem before the joint limit is reached. From then, either a new target must be chosen, the box could be shaken, or gripper could be used to move into "clearer air" and a new image can be captured.

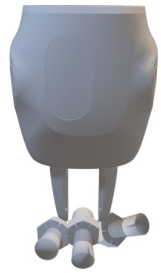


Figure 7.4.: Collision with neighbouring parts

7.5.4. Custom gripper design

As stated the main problem with the gripper is that they collide with neighboring parts before they can reach the target. To overcome problem of collision between fingers and neighboring parts, a new set of fingers where created that have slimer and narrower design that should easier slide into the gaps between the parts. The 2mm finger thickness proved to be too flexible allowing the servomotors of the grippers to go to full close when holding the part. They were not able to get a good grip on the part an either lost it or never managed to pick it up in most cases. The 3mm fingers were also quite flexible and the fingers would bend, but were still be able to hold the part. The 4mm fingers also experienced some degree of bending when holding the part, but much less sever.

The testing showed that having thinner fingers reduced the occurrence of collisions with neighboring parts, but the amount of bending does mean they need to redesigned to increase the stiffness. A solution to this could be to change the material to metal as recommended by ABB in their smart gripper manual [83]. Another solution could be to increase the support structure between the finger and base to make the finger stiffer. Testing also showed that when angle of attack was to steep or the parts were close to the wall and deep in the bin, the hand of the gripper could collide with the bin. To overcome this a new set of longer fingers should be designed so that hand does not go down in the bin, only the fingers.

7.5.5. Vacuum gripper

Testing showed the limitations of the stock vacuum gripper. It was not able to get an airtight seal on the small flat surfaces or the cylindrical surfaces on the connectors. It was also not able to get a seal on the small lask, washers and nuts due to the size of the surface areas on these parts. This result shows the limitations when dealing with small parts with holes and curved parts. However,

the advantage of using the vacuum gripper is that it picks from above, i.e. the risk of colliding with neighboring parts is significantly reduced compared to gripper fingers. This advantage merits further development, an important note is that rubber suction seal used is the one that comes stock with gripper and has been on the robot for some time. So, the plastic has aged which impacts it performance negatively by making the rubber harder. The size of the rubber seal is also not optimized for the task. By improving the system which new and smaller rubber seal it could be possible to pick more off the parts. .

7.6. Placement

One solution to place the parts is based on having the sorting tray in a fixed location and hard-coding the placement poses. A 2D camera could be used to verify that the placement were correct. Another is to use ArUco markers to identify each placement location and use the same procedure developed to identify them in cropping method. Since the trays have fixed heights only the x and y coordinates are needed, so a 2D camera would work. There have only been some testing to verify the accuracy of the robot which showed promising results, but this was not implemented in the system due to prioritizing picking and limited time.

7.6.1. Intermediate step, funnel

Funnel solution for an intermediate step before placing is proposed as solution to the problem off each pick not being identical. This removes previous grasping restriction imposed by the defined placement of the parts. The basic concept is that each cylindrical part, i.e., brass, and large and small female connector can, after picking, be moved to a fixed location above a funnel. At this point only the general direction of the y-axis is important and because it is aligned with the y-axis of the gripper it is easy to define. The funnel correctly orients the parts and guides it to a retainer ring where the robot can pick the part again. This time the pose of the part is always the same and the robot target pose can be hard-coded.

7.7. System

7.7.1. System time

From the evaluation of the system time shown in Figure 6.17, it can be seen that 93% of the time is used to capture the image, convert to .ply and read the file.

The time used to capture the image is based on the set capture time, reducing this would reduce image quality is therefore not recommended to alter. To convert the captured point cloud to .ply in HALCON the procedure `write_object_model_3d` is used. This procedure is considerably slower compared to when the .ply file is created in Zivid Studio after an image is captured. This gives an indication that the procedure can be improved to reduce time for the overall system. Reading the file is done with the HALCON procedure `read_object_model_3d`, it does not have any parameters for time so this cannot be reduced. The overall cycle time of the system can be improved by looking for several parts in each picture to reduce the down time when the robot waits for a new target after a new image have been captured. This was not prioritized for this project as it would complicate the interface between HALCON and ABB. The time used for robot motion was evaluated in depth, but testing showed that the robot reached the ready pose before the a new target was created.

7.7.2. System cost

The cost of a bin picking varies depending on the system requirements. The cost of a similar system used in this projects is $6,875 for HALCON, 6,500€ for Zivid One+ or for Zivid Two 7000€, and $80,000 for ABB YuMi [96]–[98]

7.8. Challenges

7.8.1. Testing

Testing the system was very time consuming with a lot of unexpected challenges. One of the main issues time wise was connecting to the Zivid camera in HALCON, a rough estimate is that in 8/10 attempts it could not connect to the camera because of an `open_framegrabber` error where the device could not be initialized. The cause of this error was not found, there was no issue when connecting to the camera in Zivid studio and it did work after repeated attempts. A suggestion from a representative from Zivid suggested it might be a driver error with GenICam, so this was updated, but the problem remained.

7.8.2. Equipment problems and COVID restrictions

The testing time was reduced because of an expired license for ABB RobotStudio in the period 16.12.20 – 10.02.21. This license was provided by a subcontractor for NTNU so renewing it took a while. After the license was renewed an “internal position error” with robot caused a new delay in the period 04.03.21 – 24.03.21. This was fixed after consultation with the RobotNorge, the supplier of the robot

and a representative from ABB. The solution was to do a fine calibration of the robot. In total these errors reduced the planned lab work with two months, limiting the time to test the system. Additionally, there was a COVID outbreak in the middle may which lead to the school be closed for two weeks which further restricted the access to the lab.

Chapter 8.

Conclusion and further work

8.1. Conclusion

This thesis has presented the development of an automated system for unstructured bin picking using 3D vision and machine learning. All the functionality is programmed in MvTec HALCON, except for robot motion that is written in RAPID in ABB RobotStudio.

To set up the system, Zivid and HALCONs methods for hand-eye calibration were implemented and evaluated. The best result was obtained using Zivids method and the precision obtained gave a good accuracy when picking. The acquisition settings were based on Zivids assisted mode and adjusted to work better with shiny objects. This was confirmed by comparing the point clouds before and after adjustment. Different grippers were tested. The stock grippers had issues with collision between fingers and neighboring parts when picking the target part. Therefore, a set of custom fingers were designed that were thinner so they could get into the small gap between the parts. This significantly reduced collisions. The vacuum gripper was tested to evaluate which parts it could pick. The result showed that they were limited to parts with large flat surface areas. Grasping analysis were conducted on all the parts to determine the best way to define their coordinate system and the rules governing how the match pose should be transformed to create a valid robot target. The analysis was experimentally performed by holding the parts in the gripper in various poses and evaluate the grip. The picking rules proved to be reliable. Evaluation and testing showed that they correctly transformed the pose.

The system uses a Zivid One 3D camera to capture point clouds of bins. The captured point cloud was cropped to only contain the bin of interest using a local procedure developed in MvTec HALCON. This procedure proved to be accurate and fast. Pose estimation is done using point pair features in a surface matching

procedure. The resulting poses were evaluated in a local match evaluation procedure based on match score and where the part is in the bin. The procedure was evaluated for each part. The best result obtained was for the small and large female connector which had a success rate of 82% and 76%. Due to the number of parts available for testing the result for the other parts only gives an indication on the performance, but it did confirm the reliability of the picking rules.

The ABB robot performs its own motion planning. Predefined configurations were used as intermediate steps and the target pose configurations were determined based on iterating through a list of joint configurations. This was evaluated for all parts and in most cases the cause of failure was either collision between the gripper fingers and parts in the bin or a false match. The robot and computer running HALCON used TCP/IP to send strings containing messages or poses. Some error handlers were implemented to shut down the procedure if errors occur.

The main objectives for this thesis have been achieved. The developed bin picking system can pick shiny objects from unsorted bins using machine learning and 3D vision, grasping analysis and gripper development have been done, and the system is experimentally verified in the lab. However, it is not yet capable of picking continuously over longer periods of time without operator assistance due to false matches and errors not handled by the software, but it is a good start that with further development should be capable of more consistent performance.

8.2. Further work

Further work on the system would be to improve the surface matching result and evaluation to correctly identify good matches for each run so that the system can work continuously. This can done by upgrading the hardware, software and configuration as described in the discussion. Furthermore, the code must be expanded to handle all kinds of errors that occur in these systems. The main part of the system time is spent capturing, converting and reading the image, this should be improved. The grippers were evaluated and though they did work, the discussed improvements of a smaller diameter vacuum suction cup, and longer and stiffer fingers should be investigated further to decrease the risk of collisions with the bin and other parts. Testing where done, but due to the discussed problems with the equipment, COVID and few available parts, more testing is needed to further verify the system, and to discover unforeseen issues and errors. The additional variables added to the grasping command to verify that the parts were picked was only tested to verify that it worked. Yhis must be integrated further. The placement of parts was not implemented to a large degree. The discussed solution with an intermediate step where the parts are correctly oriented must be further developed and tested. The choice of robot for the actual cell used for this ap-

plication need to be researched further. This project gives a general evaluation of performance and price of the different robots, but not enough to make a final decision.

References

[1] S. Energy. (). "Siemens energy," [Online]. Available: https://www.siemens-energy.com/global/en.html (visited on 10/03/2020).

[2] ——, (). "Energy storage solutions," [Online]. Available: https://www.siemens-energy.com/global/en/offerings/renewable-energy/energy-storage-solutions.html (visited on 10/03/2020).

[3] INTEK. (). "Intek," [Online]. Available: Siemens%20Energy:%20Energy%20storage%20solutions (visited on 10/03/2020).

[4] H. D. Berven, "Development of a virtual bin picking system," *Delivery specialization project, TPK4560*, 2020.

[5] T.-T. Le and C.-Y. Lin, "Bin-picking for planar objects based on a deep learning network: A case study of usb packs," *Sensors*, vol. 19, no. 16, p. 3602, 2019.

[6] S. Lee and Y. Lee, "Real-time industrial bin-picking with a hybrid deep learning-engineering approach," in *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2020, pp. 584–588.

[7] A. Zeng, K. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1386–1383.

[8] D. Li, N. Liu, Y. Guo, X. Wang, and J. Xu, "3d object recognition and pose estimation for random bin-picking using partition viewpoint feature histograms," *Pattern Recognition Letters*, vol. 128, pp. 148–154, 2019.

[9] R. König and B. Drost, "A hybrid approach for 6dof pose estimation," *arXiv preprint arXiv:2011.05669*, 2020.

[10] T. Hodan, M. Sundermeyer, B. Drost, Y. Labbe, E. Brachmann, F. Michel, C. Rother, and J. Matas, "Bop challenge 2020 on 6d object localization," *arXiv preprint arXiv:2009.07378*, 2020.

[11] Pickit. (). "Pickit," [Online]. Available: https://www.pickit3d.com/ (visited on 06/25/2020).

[12] K. M. Lynch and F. C. Park, *MODERN ROBOTICS: MECHANICS, PLANNING, AND CONTROL*. Cambridge University Press, 2017.

[13] CHRobotics. (). "Understanding quaternions," [Online]. Available: http://www.chrobotics.com/library/understanding-quaternions (visited on 11/25/2020).

[14] R. I. Alex Owen-Hill. (2016). "3 types of robot singularities and how to avoid them," [Online]. Available: https://robohub.org/3-types-of-robot-singularities-and-how-to-avoid-them/ (visited on 11/25/2020).

[15] Wikipedia. (2020). "Gimbal lock," [Online]. Available: https://en.wikipedia.org/wiki/Gimbal_lock (visited on 11/25/2020).

[16] MathsPoetry, *No gimbal lock*. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=5948979.

[17] ——, *Gimbal lock*. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=5949077.

[18] N. Tardella. (Sep. 2019). "Reasons why high-dof robots haven't caught on," [Online]. Available: https://www.therobotreport.com/reasons-why-high-dof-robots-havent-caught/. (accessed: 14.01.2021).

[19] J. M. Hollerbach, "Optimum kinematic design for a seven degree of freedom manipulator," in *Robotics research: The second international symposium*, MIT Press Cambridge, MA, vol. 2, 1985, p. 215.

[20] P. O. Egeland, "Robot vision," Tech. Rep., 2020.

[21] Ø. T. Zivid Knowledge Base. (). "Detectable light intensity in a camera capture," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98795895/Detectable+Light+Intensity+in+a+Camera+Capture (visited on 10/21/2020).

[22] Wikipedia. (). "Color temperature," [Online]. Available: https://en.wikipedia.org/wiki/Color_temperature (visited on 10/21/2020).

[23] S. S. Zivid Knowledge Base. (). "Color balance," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/484081806/Color+Balance (visited on 10/21/2020).

[24] J. B. Margaret Rouse. (). "Signal-to-noise ratio (s/n or snr)," [Online]. Available: https://searchnetworking.techtarget.com/definition/signal-to-noise-ratio (visited on 10/21/2020).

[25] M. Levoy. (). "Depth of field," [Online]. Available: https://graphics.stanford.edu/courses/cs178/applets/dof.html (visited on 10/20/2020).

[26] Ø. T. Zivid Knowledge Base. (). "Depth of focus," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/99877070/Depth+of+Focus (visited on 10/20/2020).

[27] ——, (). "Exposure time," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/99450881/Exposure+Time (visited on 10/20/2020).

[28] S. S. Zivid Knowledge Base. (). "Aperture," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/451149880/Aperture (visited on 10/20/2020).

[29] Wikipedia. (). "High dynamic range imaging," [Online]. Available: https://en.wikipedia.org/wiki/High-dynamic-range_imaging (visited on 08/30/2020).

[30] Ø. T. Zivid Knowledge Base. (). "Introduction to stops," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98763232/Introduction+to+Stops (visited on 10/21/2020).

[31] Wikipedia. (). "F-number," [Online]. Available: https://en.wikipedia.org/wiki/F-number (visited on 10/21/2020).

[32] digitizationguidelines. (). "Gain (image)," [Online]. Available: http://www.digitizationguidelines.gov/term.php?term=gainimage (visited on 10/21/2020).

[33] N. Mansurov. (). "What is iso? the complete guide for beginners," [Online]. Available: https://photographylife.com/what-is-iso-in-photography (visited on 10/21/2020).

[34] Ø. T. Zivid Knowledge Base. (). "Gain," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/100007959/Gain (visited on 10/21/2020).

[35] ——, (). "Blooming - bright spots in the point cloud," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98861207/Blooming+-+Bright+spots+in+the+point+cloud (visited on 10/22/2020).

[36] S. S. Zivid Knowledge Base. (). "Reflection filter," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/482410590/Reflection+filter (visited on 10/21/2020).

[37] A. P. Zivid Knowledge Base. (). "Contrast distortion artifact," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/502202755/Contrast+distortion+artifact (visited on 10/21/2020).

[38] Zivid. (). "Dealing with highlights and shiny object," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98796122/Dealing+with+Highlights+and+Shiny+Objects.

[39] Wikipedia. (). "Epipolar geometry," [Online]. Available: https://en.wikipedia.org/wiki/Epipolar_geometry (visited on 10/25/2020).

[40] K. Kitani, "Triangulation," Tech. Rep., 2020. [Online]. Available: http://www.cs.cmu.edu/~16385/s17/Slides/11.4_Triangulation.pdf.

[41] Wikipedia. (). "Triangulation," [Online]. Available: https://en.wikipedia.org/wiki/Triangulation_(computer_vision)/ (visited on 10/25/2020).

[42] A. P. Zivid Knowledge Base. (). "Structured light," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/427996/Structured+Light (visited on 10/22/2020).

[43] S.-A. Dragly, "How structured light works," 2020. [Online]. Available: https://blog.zivid.com/how-structured-light-works-part-1.

[44] roboticstomorrow. (). "What is structured light imaging," [Online]. Available: https://www.roboticstomorrow.com/article/2018/04/what-is-structured-light-imaging/11821 (visited on 04/24/2018).

[45] N. Madali, "Structured light scanning," 2020. [Online]. Available: https://towardsdatascience.com/structured-light-scanning-c125d5e06c41.

[46] P. C. Library. (). "About pcl and point clouds," [Online]. Available: https://pointclouds.org/about/ (visited on 09/21/2020).

[47] C. Thomson. (). "Common 3d point cloud file formats & solving interoperability issues," [Online]. Available: https://info.vercator.com/blog/what-are-the-most-common-3d-point-cloud-file-formats-and-how-to-solve-interoperability-issues/ (visited on 09/21/2020).

[48] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 998–1005. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5540108.

[49] robots.com. (). "The future of automated random bin picking," [Online]. Available: https://www.robots.com/blogs/the-future-of-automated-random-bin-picking (visited on 07/30/2020).

[50] Wikipedia. (). "Bin picking," [Online]. Available: https://en.wikipedia.org/wiki/Bin_picking (visited on 08/30/2020).

[51] M. Ingvaldsen. (). "Understanding the importance of 3d hand-eye calibration," [Online]. Available: https://blog.zivid.com/importance-of-3d-hand-eye-calibration (visited on 11/25/2020).

[52] M. Ulrich, A. Heider, and C. Steger, "Hand-eye calibration of scara robots," *OGRW2014*, p. 117, 2014.

[53] ZIVID, *Zivid one - quick user guide*.

[54] Zivid. (). "Zivid studio," [Online]. Available: https://www.zivid.com/zivid-studio.

[55] ——, (). "Zivid software developer kit," [Online]. Available: https://www.zivid.com/sdk.

[56] ——, (). "Zivid knowledge base," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/overview?mode=global#!spacehome.

[57] ——, (). "Zivid blog," [Online]. Available: https://blog.zivid.com/.

[58] A. P. Zivid Knowledge Base. (). "Zivid studio guide," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/330957112/Zivid+Studio+Guide?__hstc=&__hssc=&hsCtaTracking=0a6ee901-0aaf-45d5-8487-e29dcc50fa17%7C8ce7633b-0347-47e4-bba2-4a0ccec2efba (visited on 10/22/2020).

[59] Zivid. (). "Getting the right exposure for good point clouds," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/99844206/Getting+the+Right+Exposure+for+Good+Point+Clouds.

[60] R. K. T. Zivid. (). "Zivid vision engine," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/1091665963/Vision+Engine.

[61] MvTec, "Mvtec halcon," [Online]. Available: https://go.mvtec.com/acton/attachment/43208/f-ec077ca5-60f6-4dc7-9fb1-a5c517e9aceb/1/-/-/-/-/halcon_20.11_steady_brochure_en.pdf (visited on 11/28/2020).

[62] M. HALCON. (). "Programming with halcon," [Online]. Available: https://www.mvtec.com/products/halcon/work-with-halcon/programming/ (visited on 11/25/2020).

[63] A. robotics, *Yumi - irb 14000 product specification*. [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=3HAC052982-001&LanguageCode=en&DocumentPartId=&Action=Launch (visited on 11/25/2020).

[64] ——, *Yumi - irb 14000 data sheet*. [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=9AKK106354A3254&LanguageCode=en&DocumentPartId=&Action=Launch (visited on 11/25/2020).

[65] ——, *Yumi - irb 14000 collaborative robot*. [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=9AKK106354A3256&LanguageCode=en&DocumentPartId=&Action=Launch (visited on 11/25/2020).

[66] ——, (). "Yumi - irb 14000 collaborative robot," [Online]. Available: https://new.abb.com/products/robotics/industrial-robots/irb-14000-yumi (visited on 11/25/2020).

[67] ——, (). "Irc5," [Online]. Available: https://new.abb.com/products/robotics/controllers/irc5 (visited on 11/25/2020).

[68] ——, *Irc5 technical data*. [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=9AKK106354A3254&LanguageCode=en&DocumentPartId=&Action=Launch (visited on 11/25/2020).

[69] ——, *Irc5 with flexpendant operating manual*. [Online]. Available: https://abb.sluzba.cz/Pages/Public/IRC5UserDocumentationRW6/en/3HAC050941%20OM%20IRC5%20with%20FlexPendant%20RW%206-en.pdf (visited on 11/25/2020).

[70] ——, *Robotstudio features*. [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=9AKK107045A3920&LanguageCode=en&DocumentPartId=&Action=Launch (visited on 11/25/2020).

[71] ——, (). "Irc5," [Online]. Available: https://new.abb.com/products/robotics/robotstudio (visited on 11/25/2020).

[72] S. S. Zivid Knowledge Base. (). "Depth of focus calculator," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/452034580/Depth+of+Focus+Calculator (visited on 10/20/2020).

[73] Ø. T. Zivid Knowledge Base. (). "Brightness," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98796105/Brightness (visited on 10/20/2020).

[74] ——, (). "Adjusting filters," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/99942562/Adjusting+Filters (visited on 10/22/2020).

[75] S. S. Zivid Knowledge Base. (). "Noise filter," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/484016139/Noise+filter (visited on 10/21/2020).

[76] ——, (). "Outlier filter," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/482345089/Outlier+filter (visited on 10/21/2020).

[77] Wikipedia. (). "Gaussian filter," [Online]. Available: https://en.wikipedia.org/wiki/Gaussian_filter (visited on 10/21/2020).

[78] S. S. Zivid Knowledge Base. (). "Gaussian smoothing," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/483950634/Gaussian+smoothing (visited on 10/21/2020).

[79] ——, (). "Contrast distortion filter," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/484081687/Contrast+Distortion+filter (visited on 10/21/2020).

[80] Ø. T. Zivid Knowledge Base. (). "How to get good 3d data on a pixel of interest," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/98992363/How+to+get+Good+3D+Data+on+a+Pixel+of+Interest (visited on 10/01/2020).

[81] M. HALCON, *Pose estimation using surface-based 3d matching*. [Online]. Available: http://download.mvtec.com/halcon-12.0-solution-guide-iii-c-3d-vision.pdf (visited on 09/15/2020).

[82] ABB, *Technical reference manual rapid instructions,functions and datatypes*. [Online]. Available: https://library.e.abb.com/public/b227fcd260204c4dbeb8a58
Rapid_instructions.pdf?x-sign=f79v/883X1nHGc8fqH+WAJ2F30y/M6TZfYUuPuQpP+jeMBygouyGg+WSj8A9Otry (visited on 10/05/2020).

[83] A. robotics, *Product manual - irb 14000gripper*. [Online]. Available: https://abb.sluzba.cz/Pages/Public/IRC5RoboticsDocumentationRW6/Robots/Collaborative%20Robots/en/3HAC054949-001.pdf (visited on 11/25/2020).

[84] S. S. Zivid Knowledge Base. (). "Calculate fov and imaging distance," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/502202922/Calculate+FOV+and+imaging+distance (visited on 10/22/2020).

[85] Ø. T. Zivid Knowledge Base. (). "Working distance and camera positioning," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/99876931/Working+Distance+and+Camera+Positioning (visited on 10/22/2020).

[86] S. S. Zivid. (). "Zivid hand-eye calibration," [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/72450049/Hand-eye+calibration.

[87] Zivid. (). "Importance of 3d hand-eye calibration," [Online]. Available: https://blog.zivid.com/importance-of-3d-hand-eye-calibration (visited on 03/25/2021).

[88] M. HALCON, *Create_caltab procedure*. [Online]. Available: https://www.mvtec.com/doc/halcon/12/en/create%5C_caltab.html (visited on 01/15/2021).

[89] ZIVID, "Hand-eye calibration residuals," 2021. [Online]. Available: https://zivid.atlassian.net/wiki/spaces/ZividKB/pages/95944793/8.+Hand-eye+calibration+residuals.

[90] Zivid. (). "Compare and select camera," [Online]. Available: https://www.zivid.com/compare-and-select-3d-cameras?hsCtaTracking=56247cea-ac9f-4f5f-8ca2-48e5f65ff6a9%7Cc0f80eff-6dfd-4f0a-ad20-d4afcbb405dd (visited on 03/25/2021).

[91] M. HALCON. (). "Halcon newest features," [Online]. Available: https://www.mvtec.com/products/halcon/newest-features#c16511 (visited on 03/22/2020).

[92] ——, (). "Halcon newest features 19-05," [Online]. Available: https://www.mvtec.com/products/halcon/newest-features/halcon-19-05 (visited on 03/22/2020).

[93] ——, (2020). "Halcon newest features 20-05," [Online]. Available: https://www.mvtec.com/products/halcon/newest-features/halcon-20-05 (visited on 03/22/2020).

[94] ——, (). "Halcon newest features 20-05-1," [Online]. Available: https://www.mvtec.com/products/halcon/newest-features/halcon-20-05-1 (visited on 03/22/2020).

[95] A. Zivid. (). "Webinar: On-arm 3d vision robotics," [Online]. Available: https://info.zivid.com/on-arm-webinar?utm_campaign=Partner&utm_medium=email&_hsmi=135280168&_hsenc=p2ANqtz-8252PPUek3B6Nkun8j_n1--s4nTaiVR-QtXeIlceNcFJ_vRKMCvaY_H4TvghisOOOKV67pxEXe2BnwcKukKKaMyOKWw utm_content=135233969&utm_source=hs_email (visited on 03/05/2021).

[96] J. Automation. (). "Halcon price," [Online]. Available: https://www.jmakautomation.com/halcon-vs-cognex-visionpro (visited on 03/05/2021).

[97] Zivid. (). "Industrial 3d developer kits," [Online]. Available: https://shop.zivid.com/collections/kits?utm_campaign=RFQ&utm_medium=email&_hsmi=124926682&_hsenc=p2ANqtz--fx3JqZVwdypiVrb8G3nDCqo1zpO35AsHz_CBbAHRdijlMsoeHi2DrUPTGHfsFViDrXHzgi_ET5i56lpuU4oEy2cki2w&utm_content=124888145&utm_source=hs_email (visited on 03/21/2021).

[98] Bots.co.uk. (). "Yumi robot cost," [Online]. Available: https://bots.co.uk/yumi-robot-cost/ (visited on 03/22/2021).

Appendix A.

Name of Appendix

A.1. Troubleshooting/Errors

A.1.1. Internal position error ABB YuMi robot

Got the error of internal position error when running RAPID programs on the robot. This was solved with a fine calibration.

A.1.2. Robot limits because of ABB quaternion precision

ABB uses quaternions where the norm is 1, an important note is that ABB also requires this to be true to a 5 decimal point precision. If this is not true then we ABB raises a robot limit. In contrast HALCON only gives this with a 4 decimal point precision. To ensure the quaternion would meet ABB requirement a while loop has been developed in RAPID that increments the q_1 and q_2 with +- 0.00001 until the sufficient precision is reached.

A.2. PDF

TECHNICAL DATA SHEET

Prusament PETG by Prusa Polymers

PETG is one of the most commonly used filaments. It is an excellent choice for printing mechanically stressed parts. Compared to PLA, it is more heat resistant, more flexible and less brittle.

APPLICATIONS: The typical use of PETG is printing functional and mechanical parts. Thanks to good layer adhesion it is also suitable for waterproof prints.

NOT SUITABLE FOR: Not suitable for tiny parts

POST-PROCESSING: When post-processing PETG, it's possible to use both dry and wet sanding.

IDENTIFICATION:

Trade name	Prusament PETG
Chemical name	Copolyester
Usage	FDM 3D printing
Diameter	1.75 ± 0.02 mm
Manufacturer	Prusa Polymers, Prague, Czech Republic

RECOMMENDED PRINT SETTINGS:

Nozzle Temperature [°C]	250 ± 10
Heatbed Temperature [°C]	80 ± 10
Print Speed [mm/s]	up to 200

TYPICAL MATERIAL PROPERTIES:

Physical Properties	Typical Value	Method
Specific Gravity [g/cm3]	1.27	ISO 1183
Moisture Absorption 24 hours [%](1)	0.2	Prusa Polymers
Moisture Absorption 7 days [%](1)	0.3	Prusa Polymers
Moisture Absorption 4 weeks [%](1)	0.3	Prusa Polymers
Heat Deflection Temperature (0,45 MPa) [°C]	68	ISO 75
Tensile Yield Strength Filament [MPa]	46 ± 1	ISO 527

MECHANICAL PROPERTIES OF PRINTED TESTING SPECIMENS(2):

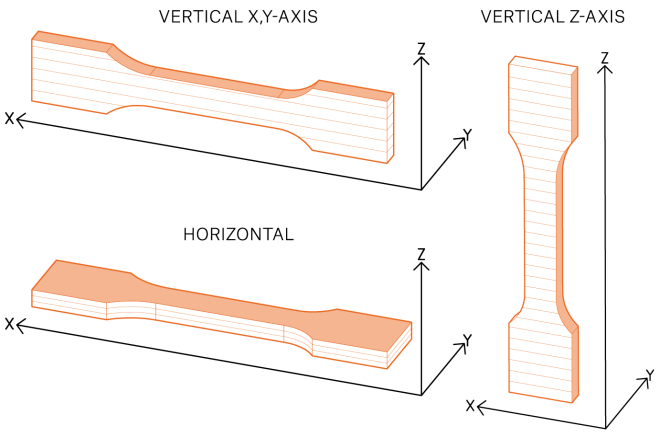
Property / print direction	Horizontal	Vertical X,Y-Axis	Vertical Z-Axis	Method
Tensile Yield Strength [MPa]	47 ± 2	50 ± 1	30 ± 5	ISO 527-1
Tensile Modulus [GPa]	1.5 ± 0.1	1.5 ± 0.1	1.4 ± 0.1	ISO 527-1
Elongation at Yield Point [%]	5.1 ± 0.1	5.1 ± 0.1	2.5 ± 0.5	ISO 527-1
Impact Strength Charpy(3) [kJ/m2]	NB(C)(4)	NB(4)	5 ± 1	ISO 179-1

(1) 30 °C; humidity 30 %

(2) Original Prusa i3 MK3 3D printer was used to make testing specimens. Slic3r Prusa Edition 1.40.0 was used to create G-codes with following settings:Prusa PETG Filament; Print settings 0,20mm FAST (layers 0,2mm); solid layers Top:0 Bottom:0; Infill 100% Rectilinear, infill print speed 100mm/s; extrusion multiplier 1.07; extruder temperature 260°C all layers; bed temperature 90°C all layers; other parameters set default

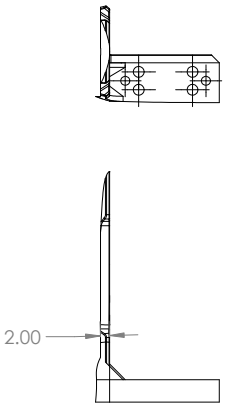
(3) Charpy unnotched - Edgewise direction of blow according to ISO 179-1

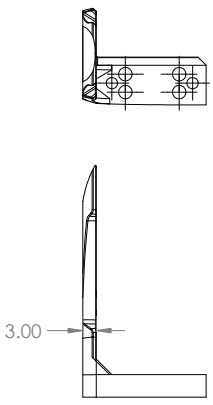
(4) NB (no break); C (complete break) in brackets second most frequent type of failure > 1/3

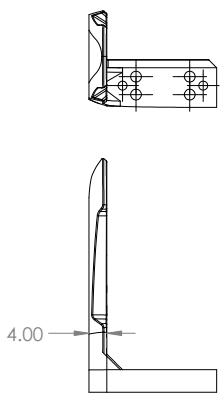


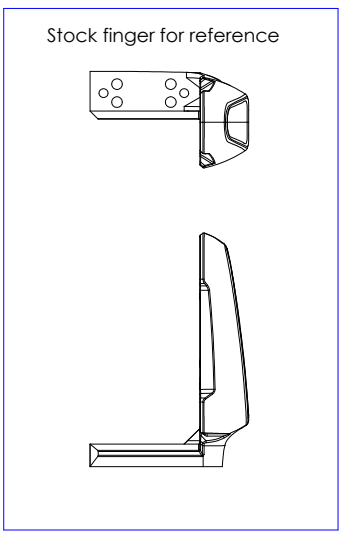
Dislamer

The results presented in this data sheet are just for your information and comparison. Values are significantly dependent on print settings, operators experiences and surrounding conditions. Everyone have to consider suitability and possible consequences of printed parts usage. Prusa Polymers can not carry any responsibility for injures or any loss caused by using of Prusa Polymers material.

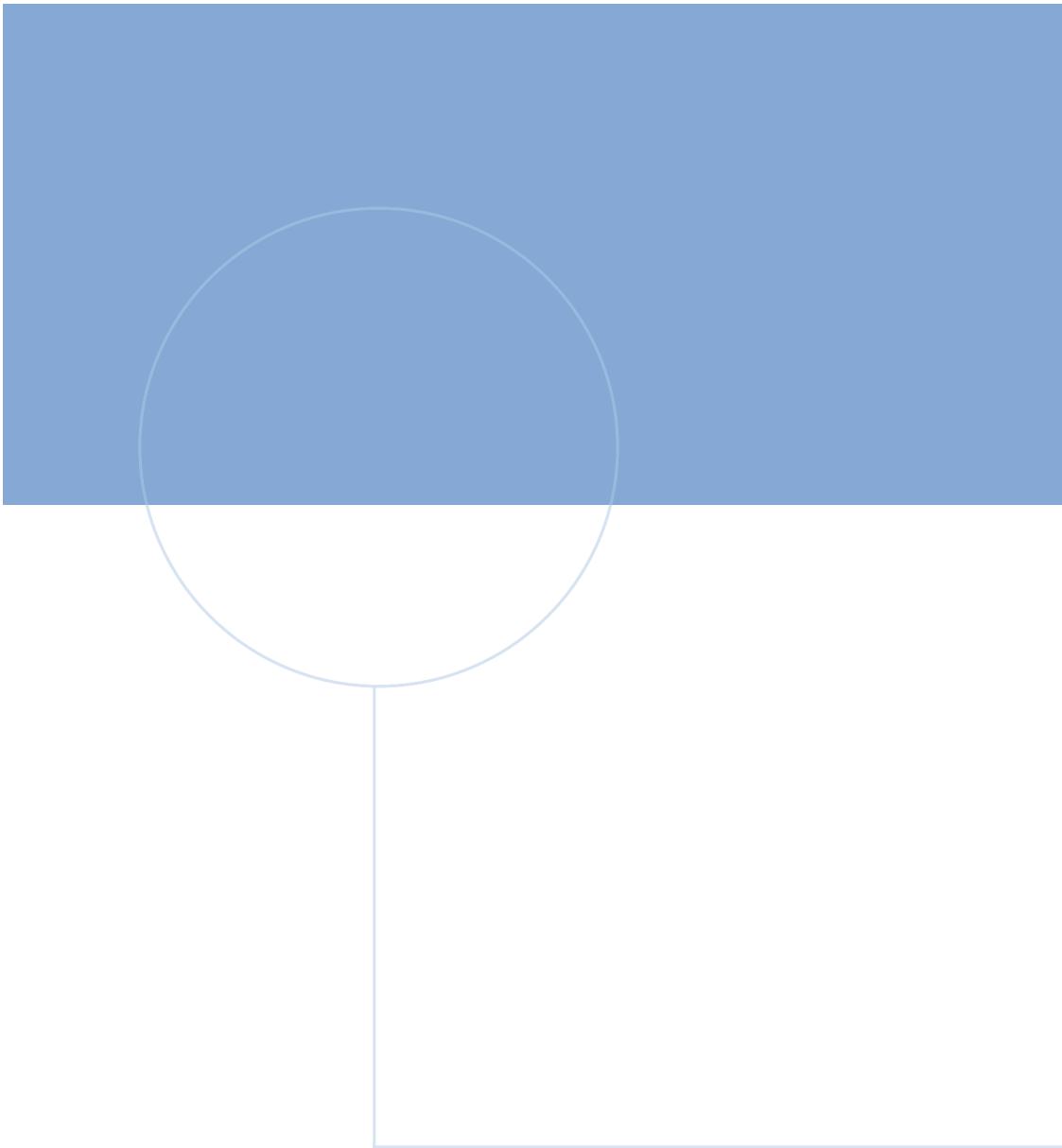








UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:		DEBURR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION	
	NAME	SIGNATURE	DATE			
DRAWN						TITLE:
CHK'D						
APPV'D						
MFG						
Q.A				MATERIAL:	DWG NO.	A4
					gripper_finger_drawing	
				WEIGHT:	SCALE:1:1	SHEET 1 OF 1



NTNU

Norwegian University of
Science and Technology