HO CHI MINH UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF CONTROL ENGINEERING AND AUTOMATION

LÊ HOÀNG ANH TÀI – VY ĐỨC KIÊN

GRADUATION THESIS

BOLT BIN PICKING ROBOTICS SYSTEM

BACHELOR OF CONTROL ENGINEERING AND AUTOMATION

HO CHI MINH CITY, MAY 2023

HO CHI MINH UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF CONTROL ENGINEERING AND AUTOMATION

LÊ HOÀNG ANH TÀI – 1951091

VY ĐỨC KIÊN – 1951010

GRADUATION THESIS

BOLT BIN PICKING ROBOTICS SYSTEM

BACHELOR OF CONTROL ENGINEERING AND AUTOMATION

INSTRUCTOR

Dr. NGUYỄN VĨNH HẢO

HO CHI MINH CITY, MAY 2023

GRADUATION THESIS REVIEWS OF THE THESIS INSTRUCTOR

Project name:

BOLT BIN PICKING ROBOTICS SYSTEM

Group members:

Lê Hoàng Anh Tài 1951091
Vy Đức Kiên 1951010

Instructor:

Dr. Nguyễn Vĩnh Hảo

Thesis review:

1. Thesis report:

Number of pages: _____ Number of chapters: _____

Number of tables: _____ Number of figures: _____

Number of documents: _____ Product: _____

Reviews of the format of the report:

(*Comments on format, report writing style, content distribution, are the chapter headings appropriate, etc.)*

.....
.....
.....
.....

2. Thesis content:

(Comments on the knowledge and methods that students have researched, evaluated the advantages and disadvantages.)

.....
.....
.....
.....

3. Thesis applicability:

(Comments on the construction of demo applications, evaluation of advantages and disadvantages.)

.....
.....

.....
.....

4. Working attitude of the students:

(Comments on the attitude, strengths, and weaknesses of each participating student.)

.....
.....
.....
.....

General assessment: The thesis meets/does not meet the requirements of a graduation thesis with a low/medium/good grade.

Scores of each student:

Lê Hoàng Anh Tài: ……./10

Vy Đức Kiên: ……/10

INSTRUCTOR

GRADUATION THESIS REVIEWS OF THE THESIS ASSESSOR

Project name:

BOLT BIN PICKING ROBOTICS SYSTEM

Group members: **Instructor:**

Lê Hoàng Anh Tài 1951091 Dr. Nguyễn Vĩnh Hảo

Vy Đức Kiên 1951010

Thesis review:

1. Thesis report:

Number of pages: _____ Number of chapters: _____

Number of tables: _____ Number of figures: _____

Number of documents: _____ Product: _____

Reviews of the format of the report:

(*Comments on format, report writing style, content distribution, are the chapter headings appropriate, etc.)*

.....

.....

.....

.....

2. Thesis content:

(Comments on the knowledge and methods that students have researched, evaluated the advantages and disadvantages.)

.....

.....

.....

.....

3. Thesis applicability:

(Comments on the construction of demo applications, evaluation of advantages and disadvantages.)

.....

.....

.....

.....

4. Working attitude of the students:

(Comments on the attitude, strengths, and weaknesses of each participating student.)

.....

.....

.....

.....

General assessment: The thesis meets/does not meet the requirements of a graduation thesis with a low/medium/good grade.

Scores of each student:

Lê Hoàng Anh Tài: ……./10

Vy Đức Kiên: ……/10

ASSESSOR

ACKNOWLEDGEMENT

First of all, we would like to sincerely express our deepest gratitude to the esteemed teachers of the University of Technology - Vietnam National University Ho Chi Minh City in general and the Automation and Control Department in particular, who have helped us since our early days at school. Thanks to that, we have a solid knowledge foundation to complete this graduation thesis research topic.

Our luck is that this graduation thesis is guided by Mr. Nguyễn Vĩnh Hảo - Head of the Automation and Control Department. His help and constructive comments played an extremely important role throughout the process of forming ideas until completing our thesis. Hereby, we would like to express our sincere thanks and gratitude to him. For us, the learning experience from him plays a very important role in the learning period and career path later on.

Continuing our thanks, we would like to dedicate it to our family and loved ones who have loved and supported us, making a solid rear base for us to pursue an educational path and have a future ahead. In addition, we would like to sincerely thank Yaskawa Company for sponsoring equipment, creating conditions for us to carry out this thesis.

We send our love to friends who have studied and worked together in a school, faculty, department and especially brothers in laboratory 205B3 who have always been enthusiastic in supporting and helping us throughout the process of completing the graduation thesis.

Ho Chi Minh city, 18 May 2023

Student

THESIS OUTLINE

SUBJECT: BOLT BIN PICKING ROBOTICS SYSTEM.

Instructor: Dr. Nguyễn Vĩnh Hảo

Implementation time: From 26/12/2022 to 18/05/2023

Students:

Lê Hoàng Anh Tài – 1951091

Vy Đức Kiên - 1951010

Project content:

Goal:

Research and build a program for detecting, segmenting and estimating position and pose of the bolts.

Research and build a program for controlling robot to perform bin picking task with the given 6D information (3D coordinates and orientation angles).

Design a software with friendly-user interface for controlling and monitoring the whole working process of the system.

Scope of implementation:

Picking object: Bolt M10x25.

Robot: Yaskawa MotoMini.

Camera: Intel RealSense D435.

Implementation condition: Laboratory 205B3 – Ho Chi Minh University of Technology.

Methods:

Robot bin picking: Communicate with the computer through a high-speed Ethernet UDP socket packet provided by the manufacturer, then write moving commands to the controller with necessary information (position variable and speed variable) to move the designated position to pick object and place object.

Detecting, segmenting and estimating object position and 3D pose:

Preparation stage: Build a YOLOv5 model by training from a collected and pre-processed dataset to desired performance.

Implementation stage: Use the YOLOv5 model to detect and segment the object to narrow down the working space, then calculate the position and orientation angles of the object based on the information output retrieved from the YOLOv5.

Interface: Use the given tool of Qt Creator to program with C++ language.

Expected results:

Build and design a software to control and monitor the robot and the bin picking system.

Result precision reach as the following criteria:

	Object detection	**Picking object**
Expected precision	Above 90%	Above 90%

Implementation Plan

Stage 1:

Student: Lê Hoàng Anh Tài.

Work: Build a program to communicate and control Yaskawa MotoMini robot.

- ❑ Research about the controller YRC1000micro.
- ❑ Research about the UDP socket to connect the computer to the controller.
- ❑ Build a library with a packet of commands to control Yaskawa Robot in different versions.

Student: Vy Đức Kiên.

Work: Program the gripper with STM32 microcontroller (MCU).

- ❑ Research about the datasheet of the STM32 MCU.
- ❑ Learn about the UART connection between the computer and the STM32 MCU.
- ❑ Programming the necessary commands for MCU to rotate the servo to open and close the gripper.

Stage 2:

Students: Lê Hoàng Anh Tài, **Vy Đức Kiên**.

Work: Build a YOLOv5 model to detect and segment the object.

- ❑ Research about the YOLOv5 structure and its usage.
- ❑ Build dataset for the training process.
- ❑ Train the YOLOv5 model with the desired version to get desired performance.

Stage 3:

Students: Lê Hoàng Anh Tài, **Vy Đức Kiên**.

Work: Calculate position and pose of the object.

- ❑ Research about PCA methods and some image-processing techniques.
- ❑ Extract the necessary information from the result of the previous stage.
- ❑ From that information, calculate the position and 3D pose of the object.

Stage 4:

Students: Lê Hoàng Anh Tài, **Vy Đức Kiên**.

Work: Calibrate the camera with respect to the robot base.

- ❑ Research about the Hand-eye calibration methods.
- ❑ Calculate Forward kinematics and Inverse kinematics.
- ❑ Perform camera calibrations.

Stage 5:

Students: Lê Hoàng Anh Tài, **Vy Đức Kiên**.

Work: Design user interface and complete the system.

- ❑ Research about Qt Creator.
- ❑ Design the user interface and complete the whole software.

Stage 6:

Students: Lê Hoàng Anh Tài, **Vy Đức Kiên**.

Work: Evaluate the system performance.

- ❑ Research about YOLOv5 evaluation methods.
- ❑ Research about Hand-eye calibration evaluation methods.
- ❑ Perform evaluation tasks for different stages.

Confirmation of the instructor	Ho Chi Minh city, 18 May 2023 **Student 1** **Student 2**

LIST OF THESIS DEFENSE COMMITTEE

The graduation thesis committee is established according to the Decision No. ..... on ..... of the Rector of Ho Chi Minh University of Technology.

1. ..... - Chairman.
2. ..... - Secretary.
3. ..... - Examiner.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS & ABBREVIATIONS

CNN: Convolutional Neural Network

CSP: Cross Stage Partial

D-H: Denavit – Hartenberg

DMA: Direct Memory Access

GUI: Graphic User Interface

PAN: Path Aggregation Network

PCA: Principal Component Analysis

PWM: Pulse Width Modulation

SPP: Spatial Pyramid Pooling

YOLO: You Only Look Once

GUI: Graphical User Interface

ABSTRACT

Object grasping system using robots is an interesting topic that attracts special attentions in the robotic field. This thesis aims to solve two problems related to small-sized objects, in this case M10x25 bolts: 6-dimenstional position and orientation estimation of objects in 3D space and object grasping. The main algorithm of our system, which is a series of specialized sub-algorithms to determine the positional and rotational information of each M10x25 bolt. Object detection is also an important stage of the thesis. Therefore, we propose the use of a deep learning model (YOLOv5) with segmentation, which offers the striking benefit of eliminating unnecessary workspace thereby improving calculation time and accuracy. For the communication between the robot and the computer, this system is built upon a UDP-based protocol developed by the manufacturer, which was rewritten by the group in the form of a C++ library. Regarding the connection between the camera and the robot, we use hand-to-eye calibration method to find the homogeneous transformation matrix from the camera frame to the robot's base frame. Besides, we develop Graphics User Interface (GUI) based on Qt Creator platform and C++ to supervise robot's tasks more conveniently. Finally, we perform an evaluation of the implementation. With respect to object detection, our results demonstrated a mean average precision as follows: 0.988 for mAP50 and 0.822 for mAP50-95 for bounding box detection. For mask detection, we achieved values for mAP50 and mAP50-95 of 0.987 and 0.69, respectively. For hand-to-eye calibration, we attained the mean error of 0.62 mm, -0.79 mm, -1.98 mm, $-0.7°$, $0.96°$, $-1.21°$ for x, y, z, Rx, Ry, and Rz, respectively. For object estimation with respect to camera, the error of the estimation is ± 3 for x, y, ± 4 for z, $\pm 3°$ for Ry, and $[-6°, +2°]$ for Rz. On the other hand, for the estimation of the object relative to the robot base, the group achieved an accuracy of $\pm 3°$ for R_z angle, and an accuracy of $\pm 1.5°$ for R_y angle. Furthermore, a success rate of 94% was attained for object grasping possibility.

Chapter 1. INTRODUCTION OF THE THESIS

1.1. Motivations for choosing the topic

In the era of industrialization and modernization of the country, the technical field in general, or manufacturing industries in particular, are facing great design challenges in order to improve production processes that are capable of the following tasks:

- Support and replace human labor wherever human labor becomes insufficient and ineffective.
- Improve the product quality and yield, creating a supply for the already great demand existing within the market.
- Diversify products to meet different needs and uses of a wide range of customers.
- Become more flexible, ready for various operating procedures.
- Minimize the physical operating area of production lines, thus optimize the use of real estate and nature resources.

Nowadays, the estimation of position and orientation of real-life objects in two dimensions is widely applied in robots that have only a few degrees of freedom such as the SCARA robots. However, the strong development of 6 degree-of-freedom (DOF) robots has revolutionized the industrial automation, with one of the most impacted applications being the pick-and-place robots. The advantages of pick-and-place robots in industries are indeed apparent, namely in their ability to operate at very high speeds, and their utmost precision in every motion. Moreover, 6-DOF pick-and-place allows for more flexibility for complex tasks that resembles human actions. However, actual production lines mainly focus on pick-and-place robotic applications that are mostly capable of working comfortably with objects of medium to large size, whereas objects of small size such as bolts, screws and other miniature physical parts are yet to receive widespread attention, mostly due to the astronomical budget reserved for research and development (R&D) and high deployment cost. Keyence, for example, is one

of the market leaders capable of producing cutting-edge robot visions systems for advanced robot applications, offers a price of 20,000 USD for a sophisticated system consisting of four industrial depth cameras and many other sensors, combined with complex algorithms to solve the pose estimation with near-perfect efficiency and accuracy.

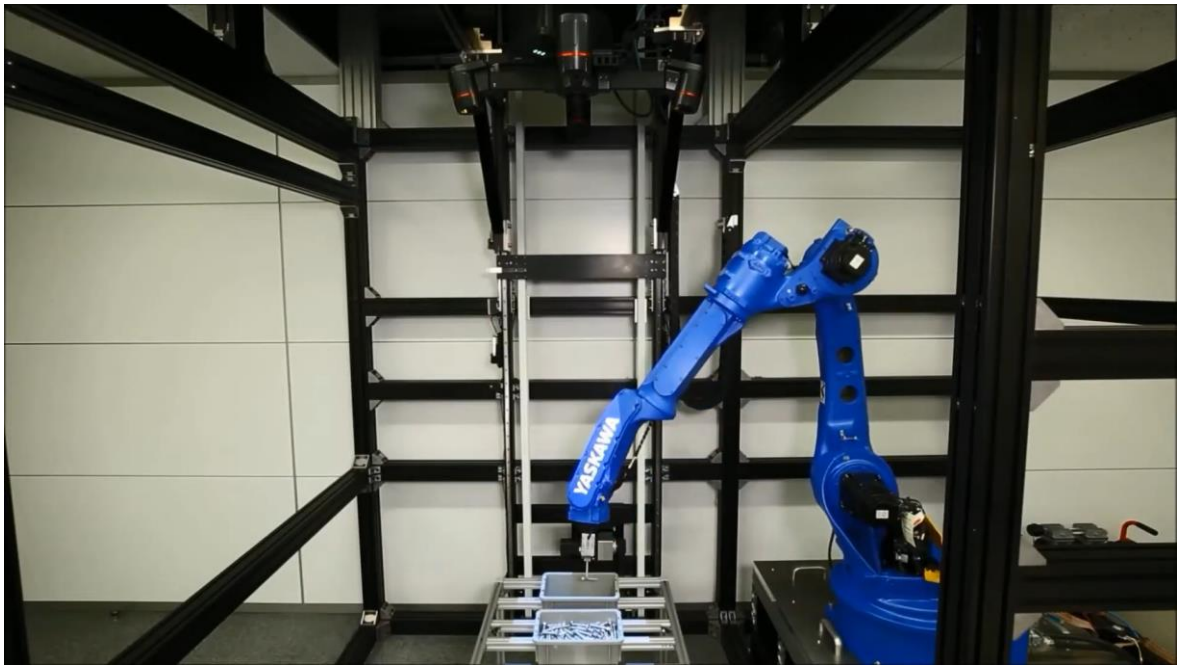


Figure 1.1. Keyence RB Series 3D VGR Camera Vision together with a Yaskawa robot performing bolts and screws bin picking activity.

Consequently, in light of the aforementioned considerations, the objective of this thesis is to propose an equivalent model of state-of-the-art systems that not only incurs lower costs but also aims to delivers comparable performance without industrial-grade equipment. The model advanced herein addresses three salient challenges in the design of a bin-picking robotic system: object detection, pose estimation, and grasping. A comprehensive account of the system's design and implementation is presented in this report.

1.2. Related works in the field

By the time the group started to work on this thesis, there had been several scientific research papers that focused on designing pick and place systems using industrial robots and specialized gripper for manipulating small-sized objects

combined with computer vision for object detection and pose estimation. In this section, some of the past studies and their approaches will be summarized.

Nowadays, 6-DOF robots are being widely applied in the industry. Market leaders in the robotic industry such as ABB, Yaskawa, KUKA, etc. all consider 6 DOF robots as a top technical priority both for competition and for commercialization. Thanks to the significant investment into research and development, 6 DOF robots now have a multitude of applications in various industries that could bring significant improvement to quality. As a major robotic company, Yaskawa offers a wide range of robot products used in automotive or electronic industry (assembling and manufacturing), health care, logistics (classification of goods), education, as well as in other applications, for example in [1] and [2].

For object detection problem, YOLO model is widely recognized as a deep learning approach that effectively balances accuracy and processing speed. Through research and development, there have been continuous improvements of YOLO through different versions of mentioned in [3], [4] [5], and [6].

The problem of object pose estimation has been addressed in various methods. Some researches rely on CAD models of the object to determine its pose, previous research by [7], for example, attempted to solve this problem by segmenting the point cloud of the detected object and then applying a fully convolutional network on the point cloud to produce its pose information (x,y,z,Rx,Ry,Rz). Other research by [8] proposed a more robust model which also makes use of a machine learning model to identify an object's pose using only RGB images and without its CAD model being seen. This machine-learning-based approach towards object pose estimation, despite being powerful in most cases, fails to see a vast usage in real-life applications, due to the labor-intensive training stage, the slow processing time, and the high computational cost requiring high-powered, industrial-grade equipment, which combined altogether is unlikely to deliver remarkable return on investment. More importantly, such approach becomes less reliable and exhibits

limitations when it comes to certain conditions, such as when the object is of a rotational symmetric shape, or when the object lacks texture.

On the other hand, within the national scope, specifically at the Ho Chi Minh City University of Technology, the Yaskawa Mechatronics Laboratory which was established and inaugurated on October 1st, 2019 provides students with the opportunity to conduct research utilizing Yaskawa's 6-DOF robot. Our team has consulted various articles and scientific research conducted by graduate students of the university, including notable theses by Mr. Thai Phat Trien and Mr. Huynh Duc Dung [9]. The research aims to utilize a single stereo camera to facilitate an algorithm for estimating the 6D position and orientation of objects. The results have been evaluated as highly reliable, with an accuracy of 5° and 5cm for the ADD standard, reaching up to 98.7%. The robot's success rate for picking up objects is up to 97%. However, the approach of this method is irrelevant to ours, due to the hardware limitation of the camera that is insufficient to capture a full point cloud of a small image to facilitate the point-cloud based algorithm. Based on our consultation and selection process, our team has chosen to employ a methodology in our graduation thesis that includes: utilizing deep learning models to recognize and classify objects. Subsequently, we will employ a specialized and customized algorithm based on the physical properties of the object to estimate its position and orientation. Finally, we will utilize a robot in conjunction with a gripper to pick up objects.

1.3. Objectives of the research

1.3.1. Subjects of the research

In this thesis, the group has defined the objects to be grasped and the necessary equipment needed for implementation as below:

- Object of interest: M10x25 bolts. Multiple instances of the object are placed within the working area of the robot.
- Robot: Yaskawa MotoMini Robot and YRC1000micro controller.
- Camera: Intel RealSense D435 depth camera.

- Main Interface: implemented in Qt, running on Windows OS.

1.3.2. Purposes of the research

In this research, the group aim to design and implement a system based on the Yaskawa robot to perform pick-and-place operations on the M10x25 bolts. A computer software which consists of object detection and estimation, as well as robot control algorithms is designed in order to control and monitor the robot system. After the implementation steps, evaluations and tests are run to verify the design choices and to help the group make proper corrections accordingly. In brief, the system is expected to achieve a 90% or higher success rate when performing object detection, and also a 90% or higher success rate in object grasping.

1.4. Problems to be solved in the thesis

The thesis is aimed to solve the following problems:

Firstly, in order to solve the problem of object recognition and pose estimation, a deep learning model is used, with its input being 2D RGB images retrieved from the camera. The output of this model, combined with the intrinsic parameters of the camera will produce positional information (x,y,z) of the recognized object. Then, by applying Principal Component Analysis (PCA) techniques and trigonometry, the rotational information (Rx,Ry,Rz) of the object will be obtained. This is also the key algorithm of the thesis.

For object grasping, the group build a hardware system consisting of a robot arm, a depth camera and a separate gripper that is attached to the end-effector of the robot. The group also builds the interface protocol between the main computer and the robot based on the datasheet of the manufacturer. Another important task is the calibration between the robot and the camera, which helps determine the correct coordinate and orientation of the object relative to the base frame of the robot before being manipulated by the gripper.

Afterwards, the group constructed a ground truth set to evaluate errors throughout the implementation process.

Finally, to enable users to control and monitor the robot's execution process in an intuitive manner, the group constructed an interface consisting of appropriate features.

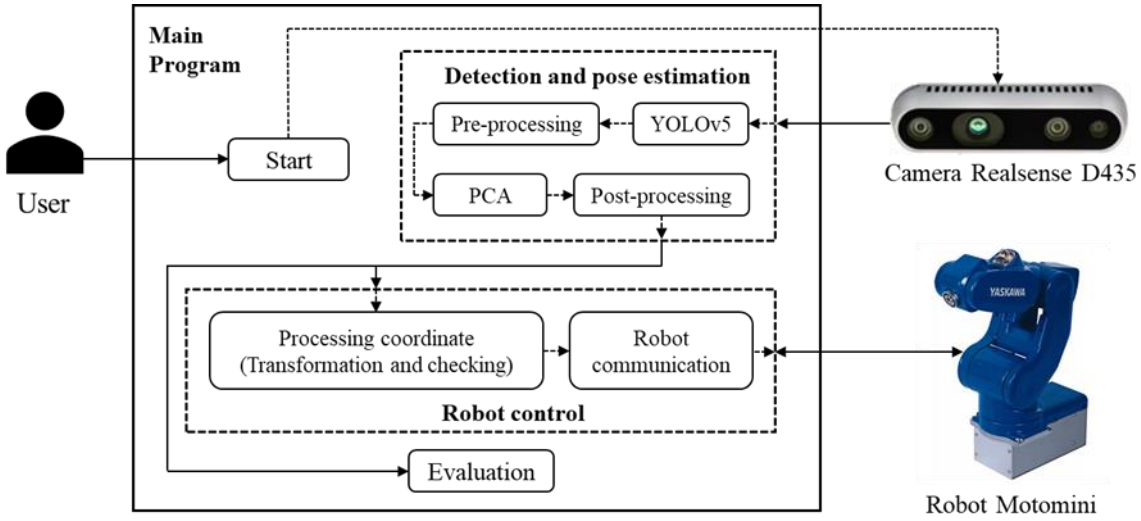


Figure 1.2. Overview of the system

Chapter 2. BASIC THEORIES

2.1. YOLOv5

For the task of detecting and classifying objects, the team chose YOLOv5 deep learning models to undertake, using YOLOv5 with high processing speed, creating object bounding box to localize the detecting area, and optimize for later estimation problems. YOLO is a convolutional neural network (CNN) first developed by Joseph Redmon for the purpose of solving object detection and classification problems (Object Detection), more specifically, YOLOv5 is a version of YOLO, which is an improved version from previous ones, YOLOv1-YOLOv4. It should be more complete (most notably, the speed of computation). The network architecture of YOLOv5 consists of 3 parts: Backbone, Neck and Head.

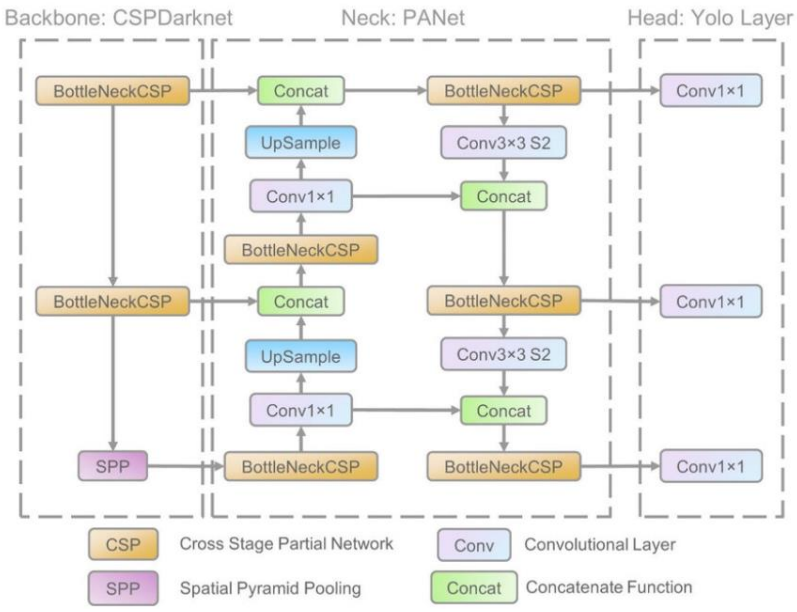


Figure 2.1. YOLOv5 network architecture. From: ResearchGate.

2.1.1. Backbone

The backbone network for object recognition is usually pretrained using the ImageNet dataset. Pre-trained has the lattice weights adjusted to identify abstract features in an image, although they will be fine-tuned for the new task of object detection. YOLOv5 is a combination of Cross Stage Partial and Darknet53: CSP-Darknet53 has higher accuracy in task object detection than CSP-ResNet, although

CSP-ResNet has higher accuracy in task classification but CSP-Darknet53 offers the following advantages:

- ❑ The input size of the network is higher, thus improving the ability to detect small objects.
- ❑ Using multiple 3x3 filters to accommodate larger input network sizes.
- ❑ Increasing the number of parameters makes it possible to detect multiple objects with different scale in the image.

2.1.2. Neck

Neck has many functions and combines the feature maps acquired through multiple feature extraction (backbone) and detecting process. YOLOv5 uses Path Aggregation Network (PAN) [10] combined with Spatial Pyramid Pooling (SPP) for neck network.

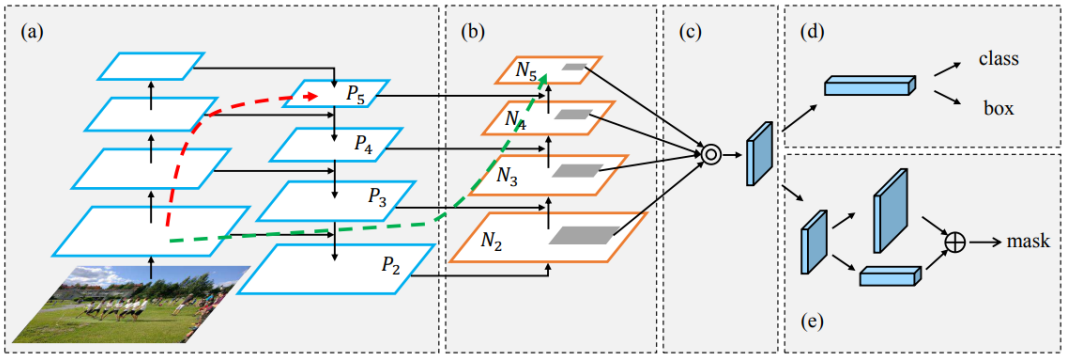


Figure 2.2. Demonstration of PAN architecture.

PANet augment another bottom-up path to make low-layer information easier to propagate to deeper layers. Therefore, with the newest release of YOLOv5, CSP-PAN is used to speed up the transmission of feature information and feature fusion.

Spatial Pyramid Pooling layer (SPP) is layer which feature map will be processed through Max Pooling layer with 4 kernels of different sizes and later flattened into a vector with fixed length.

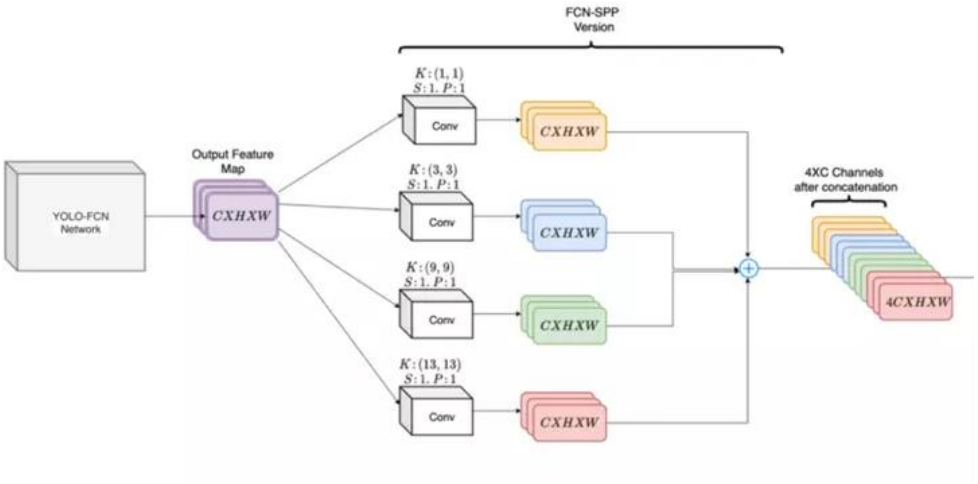


Figure 2.3. SPP layer architecture.

From the release 6.0, SPP is replaced by SPPF layer (SPP-Fast), which is a faster version, which used sequential Max Pooling instead of parallel one in SPP layer.

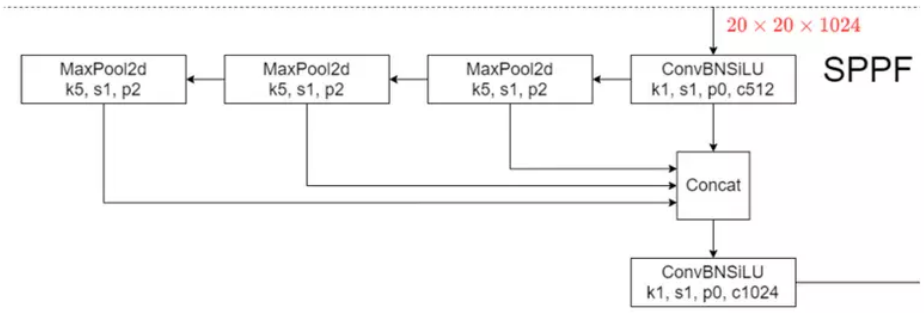


Figure 2.4. SPPF layer architecture.

2.1.3. Head

YOLOv5 uses the same head from YOLOv4 with anchor box to detect object at different scale in image.

Anchor boxes are pre-defined bounding boxes. Each grid cell will consist of 9 different anchor boxes with different sizes.

- 13x13 grid cell: (116x90), (156x198), (373x326).
- 26x26 grid cell: (30x61), (62x45), (59x119).
- 52x52 grid cell: (10x13), (16x30), (33x23).

During training session, model will take into account the loss between ground truth and anchor box and update set of values $[x, y, w, h]$ of the bounding box. From that, model will learn the feature of the object.

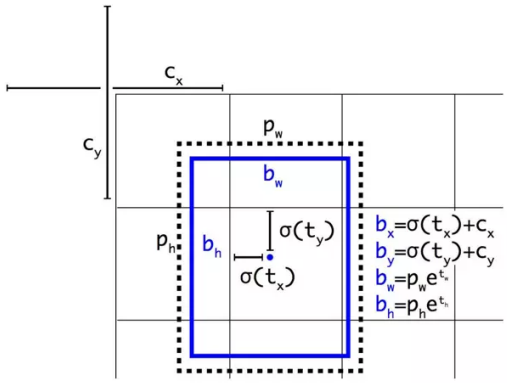


Figure 2.5.Anchor box calculation for fitting ground truth of YOLO.

2.2. PCA

Principal component analysis (PCA) is a popular technique for analyzing large datasets containing a high number of dimensions/features per observation, increasing the interpretability of data while preserving the maximum amount of information, and enabling the visualization of multidimensional data. Formally, PCA is a statistical technique for reducing the dimensionality of a dataset.

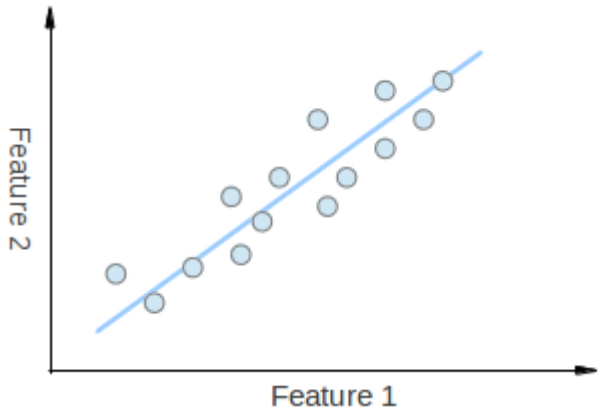


Figure 2.6. Scatter plot of an abitrary 2-dimensional dataset.

Consider a set of 2-dimensional points as it is represented in a scatter plot in Figure 2.6 as an example. Each dimension corresponds to a feature that is of interest. The points in the dataset are initially set in a random order. By

observation, there is a linear pattern (indicated by the blue line) which is hard to dismiss, indicating a positive linear correlation between feature 1 and feature 2.

A key point of PCA is the Dimensionality Reduction. Dimensionality Reduction is the process of reducing the number of the dimensions of the given dataset. For example, in the above case it is possible to approximate the set of points to a single line and therefore, reduce the dimensionality of the given points from 2D to 1D. Upon closer observation, the points tend to vary the most along the blue line, more than they do on the two axes. This mean more information can be extracted from any given point lying along the blue line, than any points along the Feature 1 or Feature 2 axes. Hence, the direction along which data vary the most can be determined with PCA. In reality, the result of running PCA on the set of points in the diagram consist of 2 vectors called eigenvectors which are the principal components of the data set. The size of each eigenvector is encoded in the corresponding eigenvalue and indicates how much the data vary along the principal component. The beginning of the eigenvectors is the center of all points in the dataset. Applying PCA to N-dimensional dataset yields N N-dimensional eigenvectors, N eigenvalues and 1 N-dimensional center point. In the current example. a two-dimensional dataset produces 2 eigenvectors, the length of each representing its corresponding eigenvalue, and 1 center point from which the 2 eigenvectors propagate.

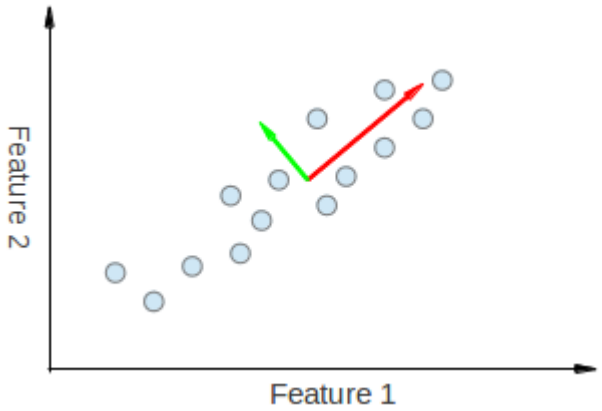


Figure 2.7. Eigenvectors represented geometrically.

The general calculation of the eigenvectors and eigenvalues are conducted through the following steps:

Step 0: Organize the dataset so that it ignores unnecessary factors like labels.

Step 1: Calculate the empirical mean for each dimension the dataset.

Step 2: From the calculated mean, compute the covariance matrix of the dataset.

Step 3: Compute eigenvectors and the corresponding eigen-values.

Step 4: Sort the eigenvectors by decreasing eigenvalues and choose k eigen-vectors with the largest eigenvalues to form a $d \times k$ dimensional matrix W.

Step 5: Represent the eigenvectors in matrix W geometrically.

In this thesis, PCA is applied to determine the body of a M10x25 bolt based on its retrieved mask during the detection stage, as well as its rotation around the z-axis, a.ka. the R_z angle.

2.3. Forward and Inverse kinematics of Robot

2.3.1. Forward kinematics

For forward kinematics calculation, the necessary parameters of Denavit – Hartenberg (D-H) table of the robot are acquired based on Figure 2.8 and are summarized in Table 2.1.

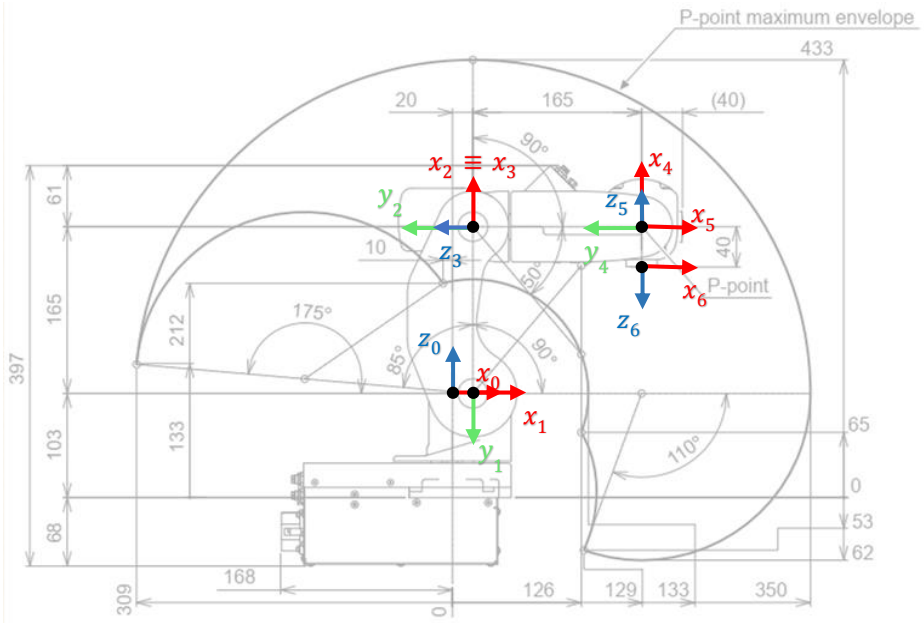


Figure 2.8. Working coordinates of each joints of the Motomini.

Table 2.1. D-H table of the Motomini.

Joint	a (mm)	α (deg)	d (mm)	θ (deg)
1	20	-90	0	θ_1
2	165	180	0	θ_2
3	0	-90	0	θ_3
4	0	90	-165	θ_4
5	0	-90	0	θ_5
6	0	180	-40	θ_6

The final result of forward kinematics is a homogeneous transformation matrix with format:

$$^0T_6 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} ^0R_6 & ^0p_6 \\ 0 & 1 \end{bmatrix} = A_1 \times A_2 \times A_3 \times A_4 \times A_5 \times A_6 \quad (2.1)$$

With:

$$A_i = {}^{i-1}T_i = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.3.2. Inverse kinematics

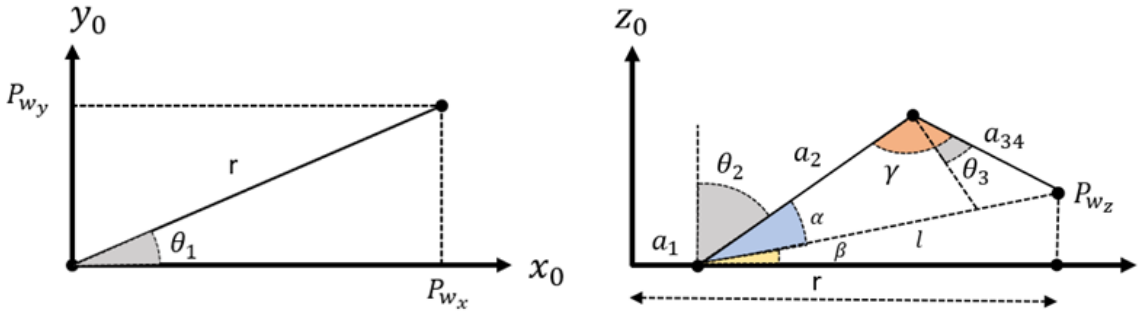


Figure 2.9. Inverse kinematic variable demonstration.

For inverse kinematics problem, the values of joint set $\{\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6\}$ need to be calculated.

We can easily find the position P_w as below:

$$P_w = [P_{w_x} \quad P_{w_y} \quad P_{w_z}]^T = {}^0p_6 - d_6 \times {}^0R_6 \times [0 \quad 0 \quad 1]^T \tag{2.2}$$

Finding θ_1:

$$\rightarrow \theta_1 = atan2(P_{w_y}, P_{w_x}) \tag{2.3}$$

Finding θ_2:

$$a_{34} = \sqrt{a_3^2 + d_4^2}$$

$$r = \sqrt{P_{w_x}^2 + P_{w_y}^2}$$

$$\beta = \begin{cases} acos\left(\dfrac{r - a_1}{\sqrt{P_{w_z}^2 + (r - a_1)^2}}\right) \ with \ (P_{w_z} \geq 0) \\ -acos\left(\dfrac{r - a_1}{\sqrt{P_{w_z}^2 + (r - a_1)^2}}\right) \ with \ (P_{w_z} < 0) \end{cases} \tag{2.4}$$

$$\alpha = acos\left(\frac{a_2^2 + l^2 - a_{34}^2}{2a_2 l}\right) \tag{2.5}$$

$$\rightarrow \theta_2 = {}^{\pi}/_2 - \alpha - \beta \tag{2.6}$$

Finding θ_3:

$$\gamma = acos\left(\frac{a_2^2 - l^2 + a_{34}^2}{2a_2 a_{34}}\right) \tag{2.7}$$

$$\rightarrow \theta_3 = \gamma - {}^{\pi}/_2 \tag{2.8}$$

Finding $\theta_4, \theta_5, \theta_6$:

From input point $[x \ y \ z \ R_x \ R_y \ R_z]$, the 4x4 homogeneous transformation matrix based on Roll-Pitch-Yaw angles are calculated as below:

$$ {}^{0}T'_{6} = T_{RPY} = \begin{bmatrix} R_{RPY} & [x\ y\ z]^T \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{2.9} $$

With:

$$ R_{RPY} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

$r_{11} = \cos_{Rz} \cos_{Ry}$

$r_{12} = -\sin_{Rz} \cos_{Rx} + \cos_{Rz} \sin_{Ry} \sin_{Rx}$

$r_{13} = \sin_{Rz} \sin_{Rx} + \cos_{Rz} \sin_{Ry} \cos_{Rx}$

$r_{21} = \sin_{Rz} \cos_{Ry}$

$r_{22} = \cos_{Rz} \cos_{Rx} + \sin_{Rz} \sin_{Ry} \sin_{Rx}$

$r_{23} = -\cos_{Rz} \sin_{Rx} + \sin_{Rz} \sin_{Ry} \cos_{Rx}$

$r_{31} = -\sin_{Ry}$

$r_{32} = \cos_{Ry} \sin_{Rx}$

$r_{33} = \cos_{Ry} \cos_{Rx}$

We have:

$$ {}^{0}T'_{6} = {}^{0}T'_{3} \times {}^{3}T'_{6} \rightarrow {}^{3}T'_{6} = ({}^{0}T'_{3})^{-1} \times {}^{0}T'_{6} = ({}^{0}T'_{3})^{-1} \times {}^{0}T'_{6} $$

$$ \rightarrow {}^{3}T'_{6} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}T_{6} = ({}^{2}T_{3})^{-1} \times ({}^{1}T_{2})^{-1} \times ({}^{0}T_{1})^{-1} \times {}^{0}T_{6} = \begin{bmatrix} n_x & s_x & a_x & x \\ n_y & s_y & a_y & y \\ n_z & s_z & a_z & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.10} $$

We have:

$$ r_{33} = -\sin\theta_5 = a_z $$

In order for the retrieved angle not to be too different from the previous robot position, the comparison between them need to be taken into account to get the appropriate values of θ_5 and θ_6:

$$\rightarrow \theta'_5 = \text{asin}(-a_z)$$

$$\rightarrow \theta_5 = \begin{cases} \theta'_5 \ with \ \theta'_5 - \theta_{5(old)} < {}^{\pi}/_{2} \\ \pi - \theta'_5 \ with \ \theta'_5 - \theta_{5(old)} > {}^{\pi}/_{2} \end{cases} \quad (2.11)$$

We have:

$$\begin{cases} r_{13} = -\cos\theta_4 \cos\theta_5 = a_x \\ r_{23} = -\sin\theta_4 \cos\theta_5 = a_y \end{cases}$$

$$\rightarrow \theta_4 = \begin{cases} atan2(-r_{23}, -r_{13}) \ with \ (cos\,\theta_5 > 0) \\ atan2(r_{23}, -r_{13}) \ with \ (cos\,\theta_5 < 0) \end{cases} \quad (2.12)$$

We have:

$$\begin{cases} r_{31} = -\cos\theta_6 \cos\theta_5 = n_z \\ r_{32} = -\sin\theta_6 \cos\theta_5 = s_z \end{cases}$$

$$\rightarrow \theta'_6 = \begin{cases} atan2(-r_{23}, -r_{13}) \ with \ (cos\,\theta_5 > 0) \\ atan2(r_{23}, -r_{13}) \ with \ (cos\,\theta_5 < 0) \end{cases} \quad (2.13)$$

$$\rightarrow \theta_6 = \begin{cases} \theta'_6 \ with \ \theta'_6 - \theta_{6(old)} < {}^{3\pi}/_{4} \\ \pi - \theta'_6 \ with \ \theta'_6 - \theta_{6(old)} > {}^{3\pi}/_{4} \end{cases} \quad (2.14)$$

2.4. UDP communication

High-Speed Ethernet Server is an application layer protocol on UDP/IP that allows connection, control, and transmission of data between an Ethernet-connected device and the YRC1000micro controller. In which, the controller acts as a server, and external devices (PC) act as clients. For each request sent from the client, the server will process and send back the corresponding response.

There are two ways to program computer applications using the Highspeed Ethernet Server protocol. One is to use the manufacturer's MotoCom library, which provides an API for users to build external application layers. The second is to build a communication library based on the protocol published by the

manufacturer, from which to write an application layer. The group will employ the second method to implement in this thesis. This will be demonstrated more specifically in section 3.2.2.

Chapter 3. DESIGN AND IMPLEMENTATION

The following figure shows the whole implementation process of the thesis:

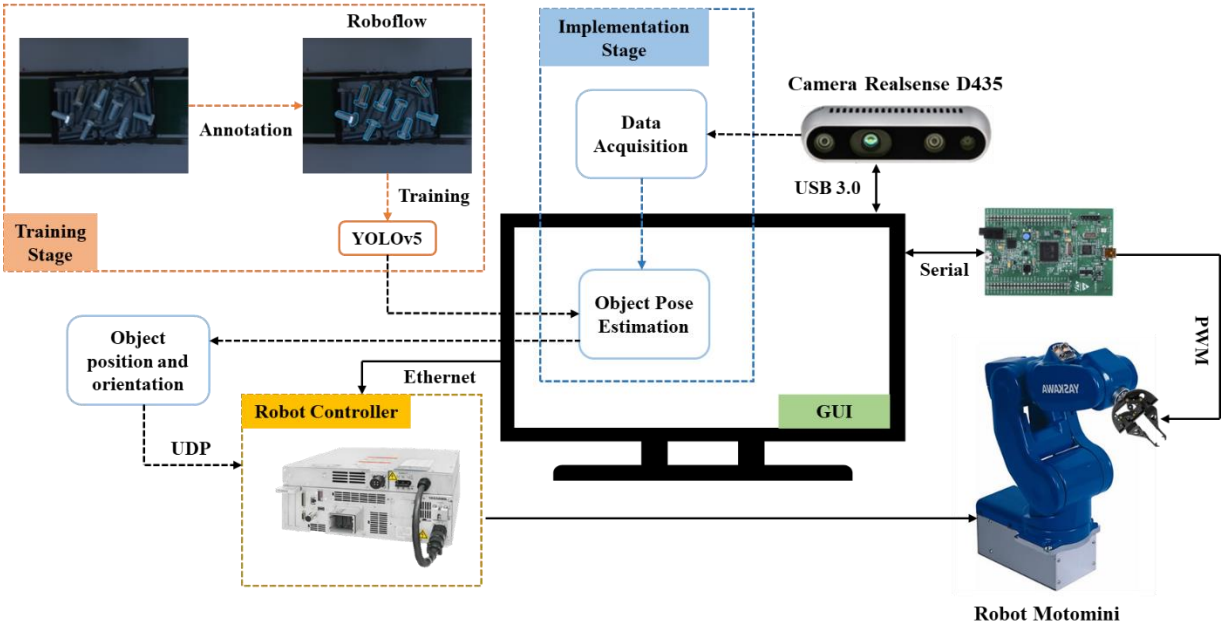


Figure 3.1. Overview of system design and implementation process.

As can be seen from the Figure 3.1, the implementation content of the thesis can be divided into 3 main parts. The first part consists of 2 main stages: training stage for YOLOv5 model and implementation stage for object pose estimation. Next part is developing a program for controlling between the computer and the robot. Eventually, the last part is to design and develop the system software, which is also a graphical user interface (GUI) for users to observe and control the whole working process.

3.1. Position and orientation estimation of the object

3.1.1. Training stage

3.1.1.1. Research object

The group selected medium-sized bolt samples gripping, specifically length of the bolt should not be too high so that the grasping point of the bolt can still balance itself inside the gripper and the diameter of the bolt must not exceed 50mm since the maximum opening of the gripper is 54mm.

Based on those factors, a M10x25 bolt with 10mm diameter and 25mm length for the thread part, including the stainless-steel bolts and the iron ones, is chosen to be the target object as seen in Figure 3.2.

Figure 3.2. Stainless-steel and iron M10x25 bolts inside a container.

3.1.1.2. Training YOLOv5

For detecting and segmenting the bolt object so as to grasp it, the group conducts training model YOLOv5. For the collecting data stage, numbers of picture sets on different backgrounds are taken. Those pictures are then uploaded to Roboflow for labeling to create dataset for training set, specifically the labeling instances are only those that have high grasping possibilities, which means they are fully visible and on the top layer. Furthermore, in order to have the best mask after detection, the object should be labeled carefully fit as much as possible.

Figure 3.3. Demonstration of labelling process in Roboflow.

The original dataset with 1071 640x640 images has 3337 instances. The group continue to use the augmentation tool provided by Roboflow for the dataset to expand the volume of dataset to 2035 images with the following augmentation options:

Table 3.1. Augmentation options of Roboflow's augmentation tools.

Augmentation methods	Settings
Flip	Horizontal, Vertical
90° Rotate	Clockwise, Counter-Clockwise, Upside Down
Crop	0% Minimum Zoom, 10% Maximum Zoom
Shear	±5° Horizontal, ±5° Vertical
Hue	Between −10° and +10°
Saturation	Between −20% and +20%
Brightness	Between −20% and +20%
Exposure	Between −10% and +10%

Blur	Up to 1px
Noise	Up to 1% of pixels

The dataset is split into 95% for training set and 5% for validation set, which is used to improve the model based on the performance feedback during the training process.

Training session is conducted on Google Colab since it gives users free and powerful GPU with 16GB VRAM for maximum GPU usage duration around 4 hours. The users just need to program in Python language to implement the necessary commands.

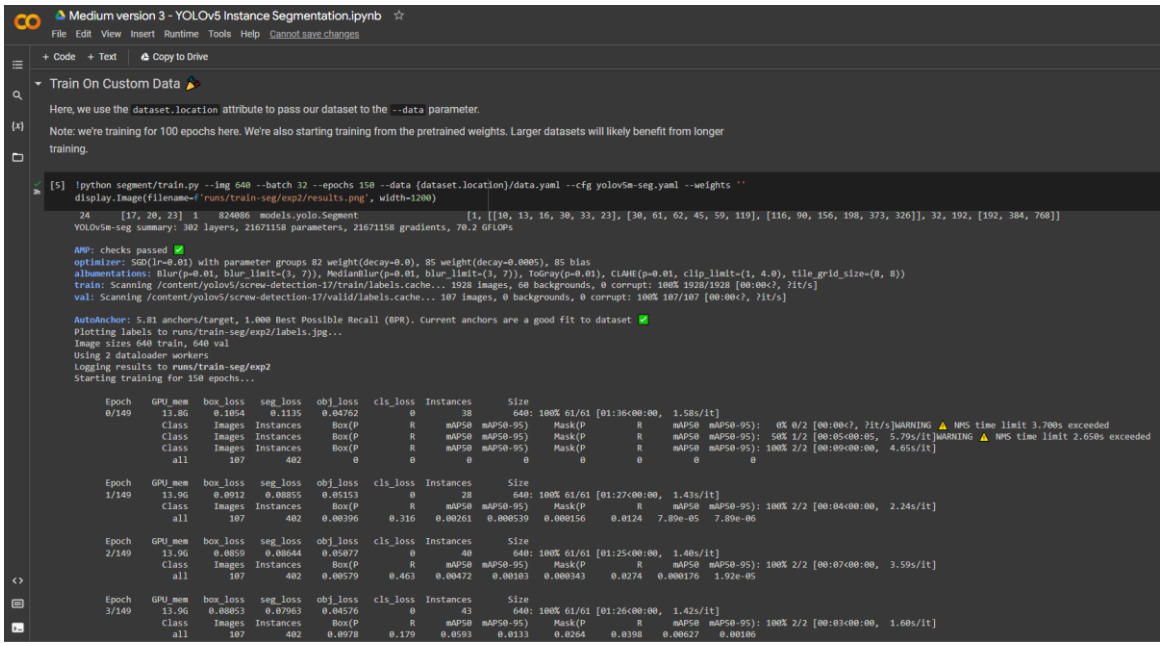


Figure 3.4. Training process of YOLOv5 in Google Colab.

The training session is initialized with 150 epochs, 32 for batch size, version config is medium-segmentation version.

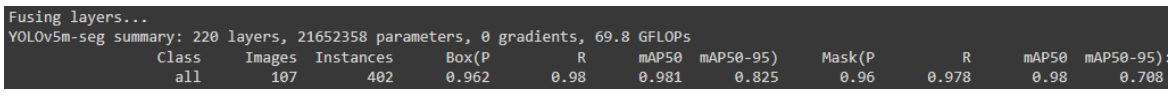


Figure 3.5. Training result of YOLOv5.

3.1.2. Implementation stage

Implementation stage is the stage to conduct calculating algorithms for the final point so as the robot can grasp precisely and can be observed and controlled from the GUI. The input data are the frames collected from the camera with the configuration files and weights of trained model acquired from the previous stage. The flow chart of the whole implementation stage is as follows:

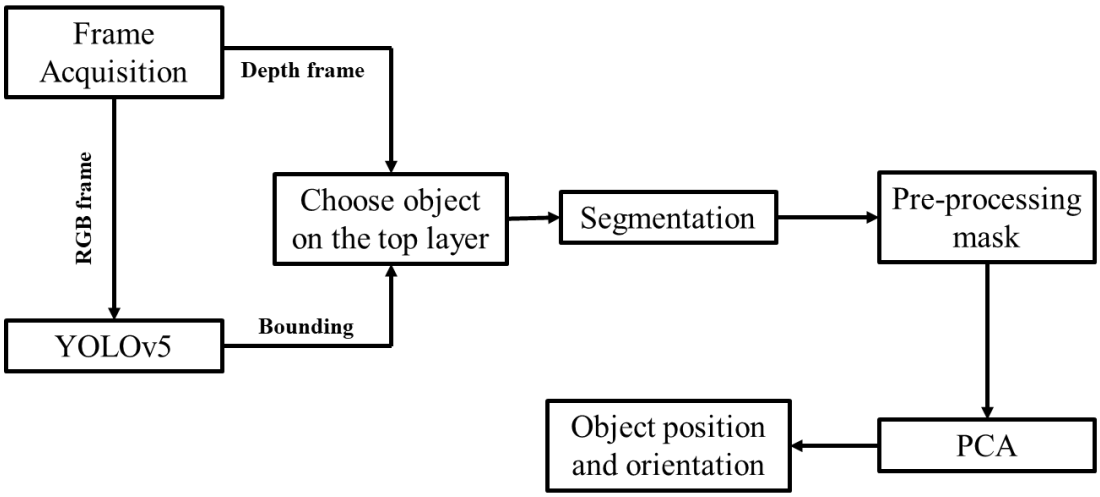


Figure 3.6. Flow chart of pose estimation process.

3.1.2.1. Data acquisition

Data acquired from the Intel RealSense D435 camera consists of 2 image frames: color frame (RGB) and depth frame (Z16). The frame is configured with the 640x480 resolution and some modification for the depth frame.

Depth value of frame is stored as Z16 value type, which is a 16-bit unsigned integer. The actual value of z (in meter) of 2D image is calculated as follows:

$$z = DepthUnit \times Z(m)$$

According to the datasheet of the manufacturer, the smaller value of $DepthUnit$ of RealSense D400 series the better surface feature extracted from the camera when working in the lower 50 cm range depth environment. The value of $DepthUnit$ selected for the project is $10\mu m$, which means that the maximum value of the depth, which can be acquired is $10\ \mu m$: $z_{max} = 10\mu m \times 65535 =$

65.535 (cm). This is an appropriate value for the current requirements of this project.

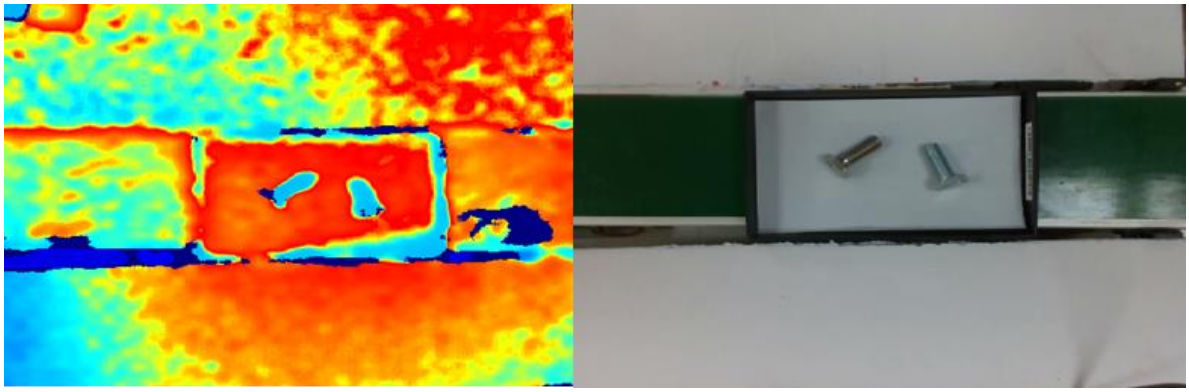


Figure 3.7. Depth image (left), and the RGB image (right) retrieved from the camera.

3.1.2.2. YOLOv5 data extraction

The RGB frame extracted from the camera RealSense D435 is then transferred into neural network YOLOv5 model, which is trained in the previous stage. The outcomes of the model are the bounding boxes of the objects with the masks covering the whole objects surface and the confidence score of the objects, which is the certainty of detected object.

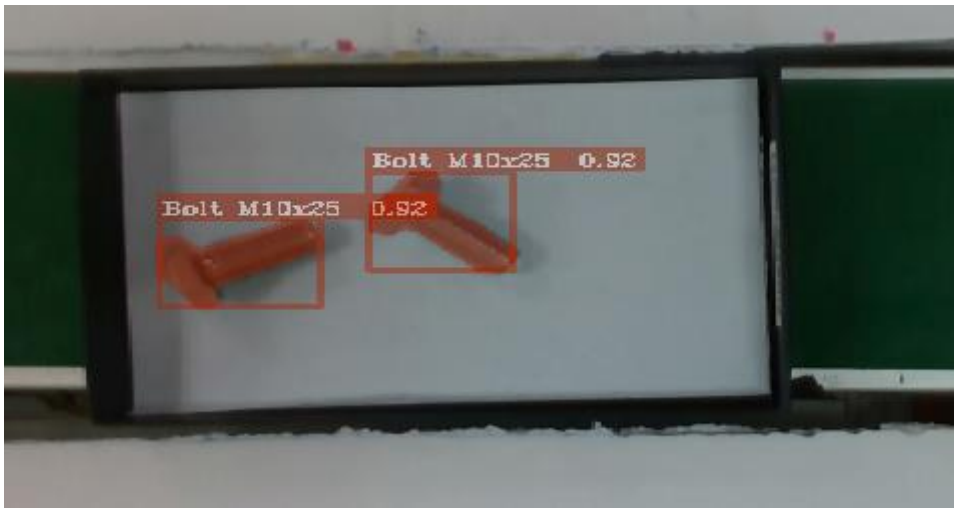


Figure 3.8. Detection result of the M10x25 bolts in YOLOv5.

For each object detected, the output information retrieved comprises the values of the bounding box description, including the top-left point of the bounding box in a 2D image and its width and height, represented as [x y w h]. Additionally,

the binary image of the object's mask is retrieved for utilization in subsequent calculations during later stages.

Figure 3.9. A mask of the object retrieved by YOLOv5.

3.1.2.3. Position and orientation estimation

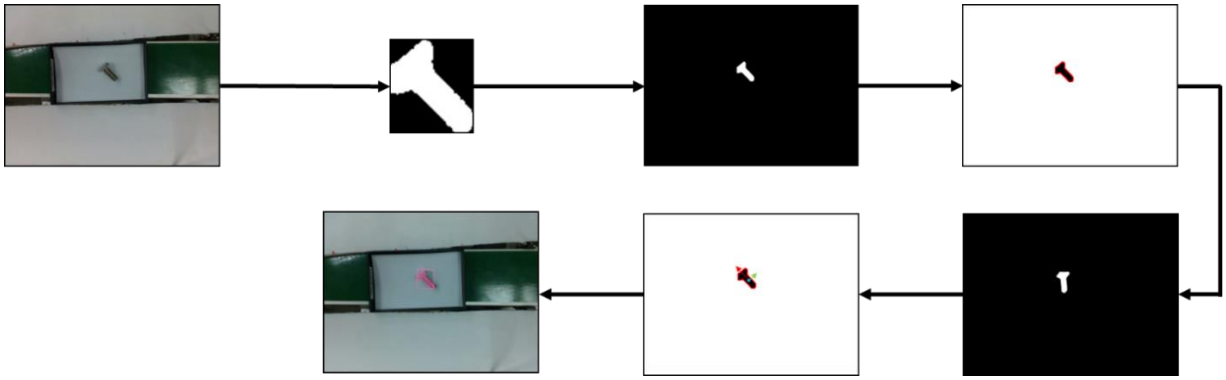


Figure 3.10. Graphical desmonstration of the pose estimation algorithm.

From the extracted mask from the outcome of the YOLOv5 model with the description information of the bounding box of the detected object [x y w h], the mask is then processed to be brought back at the exact position of the object in the RGB image to a black background image as a binary image, as presented in Figure 3.10 c).

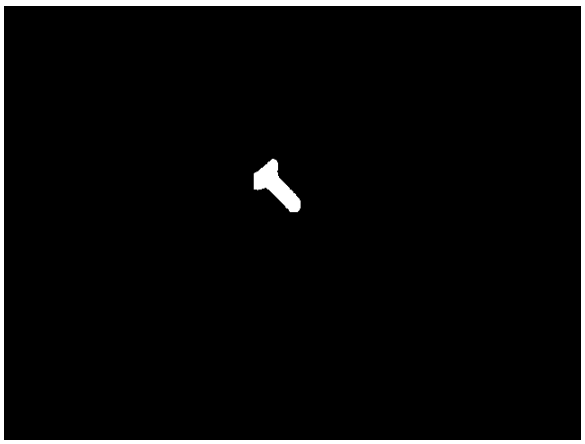


Figure 3.11. Binary image of detected object.

Then, the largest contour of the object is identified using OpenCV library, andis drawn as the result in Figure 3.12.

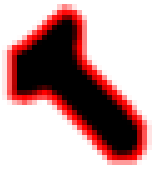


Figure 3.12. Contour of the binary mask.

Next, the group perform PCA to calculate the R_z angle with respect to camera as follows:

1. Arrange the dataset (in this case the set of n points forming a contour) into 2 columns, x and y.

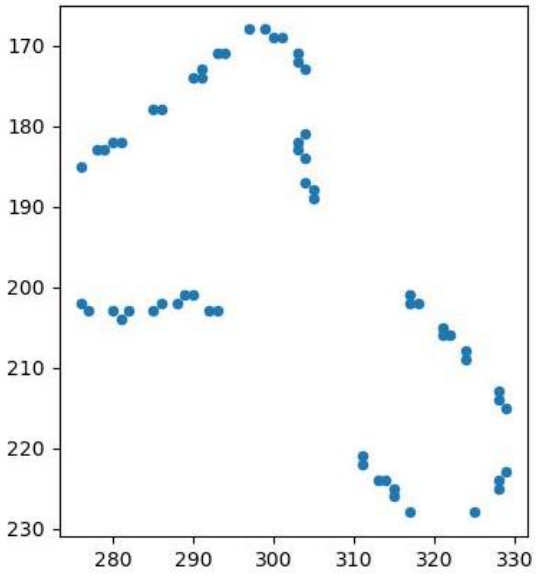


Figure 3.13. Scatter plot of the pixels forming the previously retrieved contour.

Table 3.2. Preparation of pixel data before applying PCA

x	y
x_1	y_1

x_2	y_2
x_3	y_3
...	...
x_n	y_n

In the form of matrix, the contour is represented as below:

$$A = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}$$

2. Calculate the center point of the mask, which is also the mean of each dimension of the matrix above. The mean matrix would be:

$$\bar{A} = [\bar{x} \quad \bar{y}] \tag{3.1}$$

To normalize the dataset, each row of the matrix A would be subtracted by the mean matrix, giving:

$$X = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ x_2 - \bar{x} & y_2 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \tag{3.2}$$

3. After demeaning the original data matrix, the next step is to calculate the covariance matrix C of the data. This step is also called the normalization of matrix X.

$$C = \begin{bmatrix} cov(x,x) & cov(y,x) \\ cov(x,y) & cov(y,y) \end{bmatrix} = \begin{bmatrix} var(x) & cov(y,x) \\ cov(x,y) & var(y) \end{bmatrix} \tag{3.3}$$

With

$$cov(x,y) = \frac{1}{n}\sum_{i=1}^{n}(x - \bar{x})(y - \bar{y})$$

4. The eigenvalues λ are then obtained by solving the characteristic equation. Since the data is 2-dimensional, the result would be a matrix containing 2 values of λ.

$$\det(C - \lambda I) = 0$$

Finally, for each λ calculated above, the eigenvectors are the roots of the equation

$$Cv = \lambda v \tag{3.4}$$

After solving the equation and sorting the absolute value of eigenvalues by decreasing order, a matrix W containing 2 eigenvectors is obtained as below, corresponding to the magnitude of each λ.

$$W = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} \tag{3.5}$$

Here, the eigenvectors $[x_1 \quad x_2]^T$ is chosen to be the first principal component of the dataset, which corresponds to the greatest absolute value of λ, or $|\lambda_{max}|$. This eigenvector also represents the greatest variation of the data, which apparently matches with the body of the bolt, therefore, using the first principal component, the orientation of the bolt can be determined.

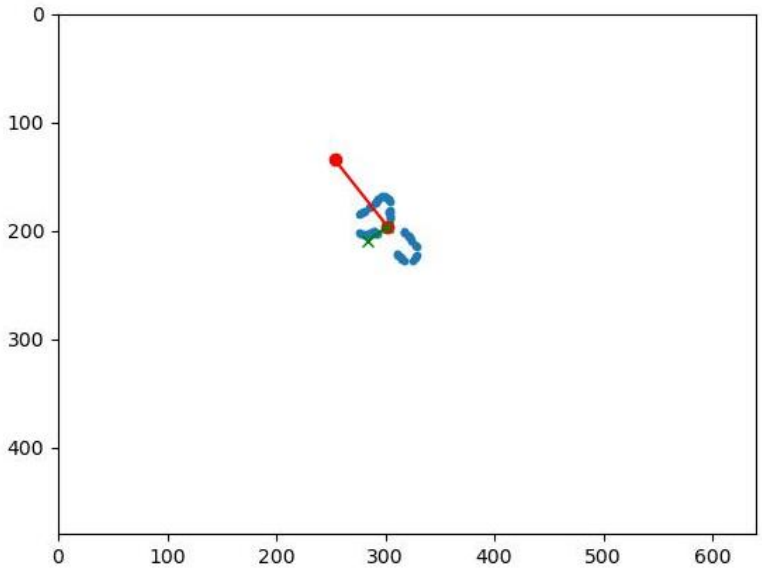


Figure 3.14. Principle components plotted on the scatter plot.

Remark: The application of Principal Component Analysis (PCA) for determining the orientation of a bolt around the z-axis has been shown to yield highly accurate results. However, it is important to note that the efficacy of this method is contingent upon the shape of the mask produced by the image detection stage. Any distortion in the mask may adversely impact the direction of the eigenvector, resulting in an erroneous calculation of the R_z angle.

From the obtained eigenvector corresponding to the maximum eigenvalue $|\lambda_{max}| - \vec{v}(x, y)$, we calculate rotate angle of the bolt about the z-axis:

$$rotate\_angle = atan2(y, x) \tag{3.6}$$

Then we calculate the angle to rotate the bolt into vertical state as follows:

$$img\_rotate\_angle = \begin{cases} -90 - rotate\_angle, \ with\, rotate\_angle < 0.0 \\ 90 - rotate\_angle, \ with\, rotate\_angle > 0.0 \end{cases} \tag{3.7}$$

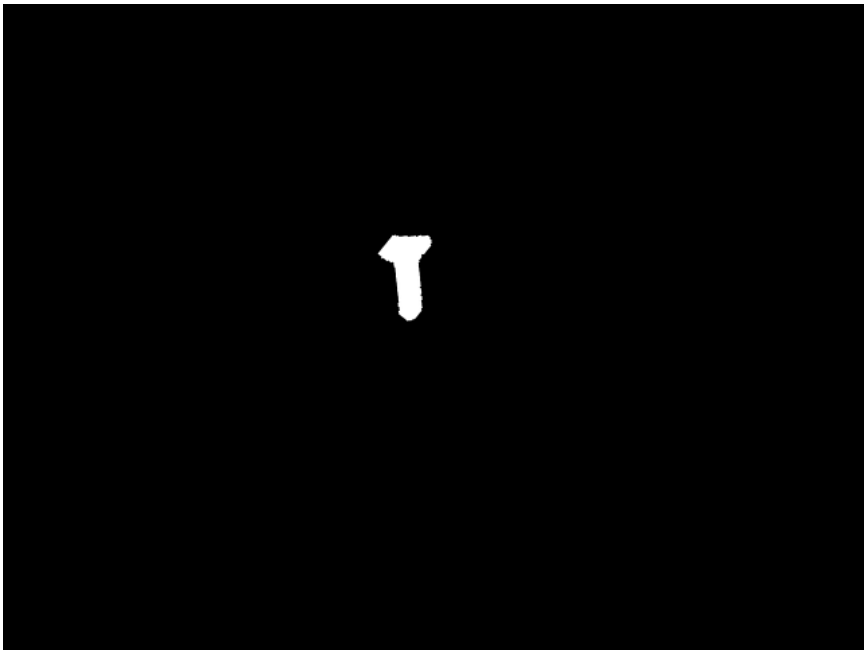


Figure 3.15. The mask after being rotated and repositioned to its original coordinates.

With the rotated image, the team continues to find the head of the bolt and determine if it is above or below.

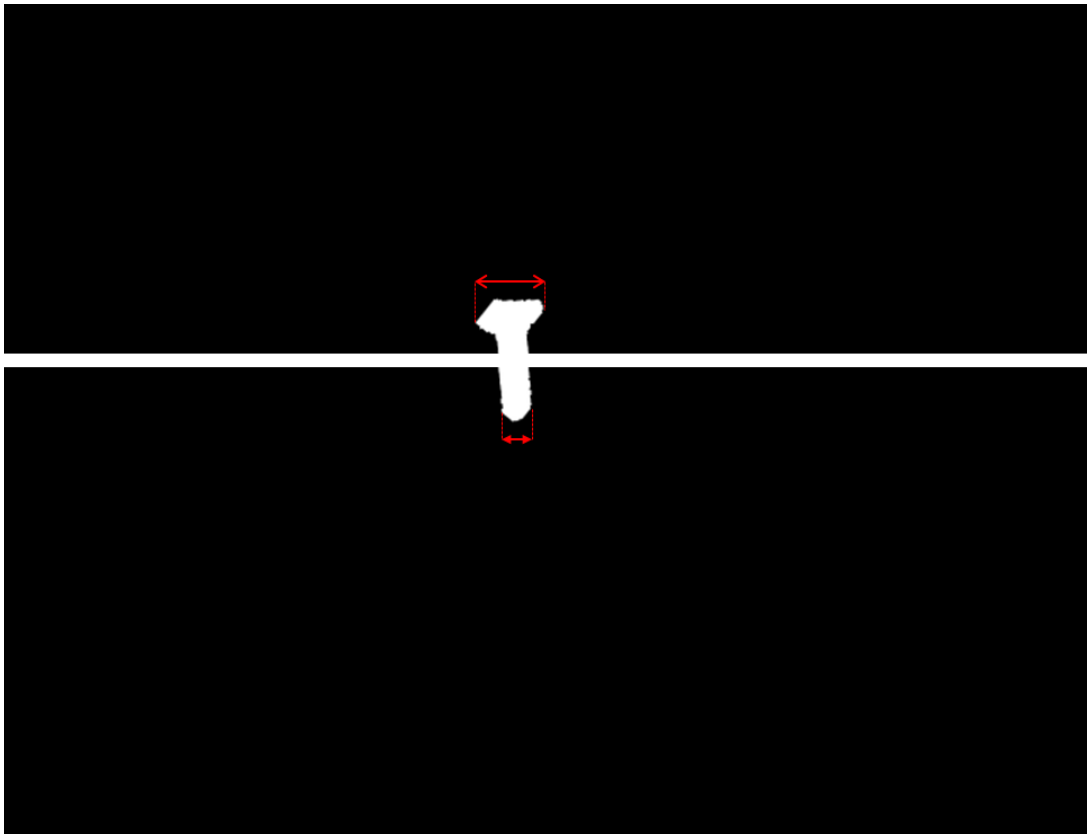


Figure 3.16. The mask is splitted to find the head based on the distance between pixels.

In the process of determining the position of the bolt head in an image, the first step involves rotating the image and splitting it into two halves based on the center founded in the previous step. The bolt head is located in one of these halves and can be identified by comparing the horizontal distance, Δx, between the white pixel with the minimum x value, x_{min}, and the one with the maximum x value, x_{max}. To accomplish this, the number of white pixels in each half of the image is counted and stored in an array. These white pixels are identified by their pixel value of 255. Once this data has been collected, a sorting algorithm is applied to determine the values of, x_{min} and, x_{max} from the array. The distance $\Delta x = |x_{min} - x_{min}|$ is then calculated for each half of the image and compared to determine which half contains the bolt head. This process allows for accurate identification of the bolt head's position within the image.

Eventually, the R_z angle with respect to camera is calculated as follows:

If head_is_up **then:**
    If $rotate\_angle \geq 0$ **then:**
        $Rz = -rotate\_angle$
    Else:
        $Rz = -rotate\_angle - 180.0$
    End if
Else if head_is_down **then:**
    If $rotate\_angle \geq 0$ **then:**
        $Rz = 180.0 - rotate\_angle$
    Else:
        $Rz = -rotate\_angle$
    End if
End if

Figure 3.17. Calculated R_z angle w.r.t. the camera frame.

From the eigenvalue $|\lambda_{max}|$ and the center point calculated above, we find the line along the bolt:

$$\begin{cases} p1(u_1, v_1) - center\ point \\ p2(u_2, v_2) - |\lambda_{max}| \end{cases}$$

$$\rightarrow y = ax + b$$

This line equation will help to get head point and end point along the bolt.

Figure 3.18. The grasping point (yellow) between the head and end point.

From these two points, the grasping point will be the center between them.

For any 2D point (u, v) of RGB image can be transformed into 3D point (x, y, z) with intrinsic parameters of the camera $[c_x\ c_y\ f_x\ f_y]$ provided by the manufacturer as follows:

$$\begin{cases} z = DepthUnit \times Z(u,v) \\ x = (u - c_x) \times \frac{z}{f_x} \\ y = (u - c_y) \times \frac{z}{f_y} \end{cases} \tag{3.8}$$

For R_y angle calculation, the formulas are shown below:

$$length = \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2} \tag{3.9}$$

$$Ry = \begin{cases} atan2(abs(z_1 - z_2), length), & z_1 \geq z_2 \\ -atan2(abs(z_1 - z_2), length), & z_1 < z_2 \end{cases} \tag{3.10}$$

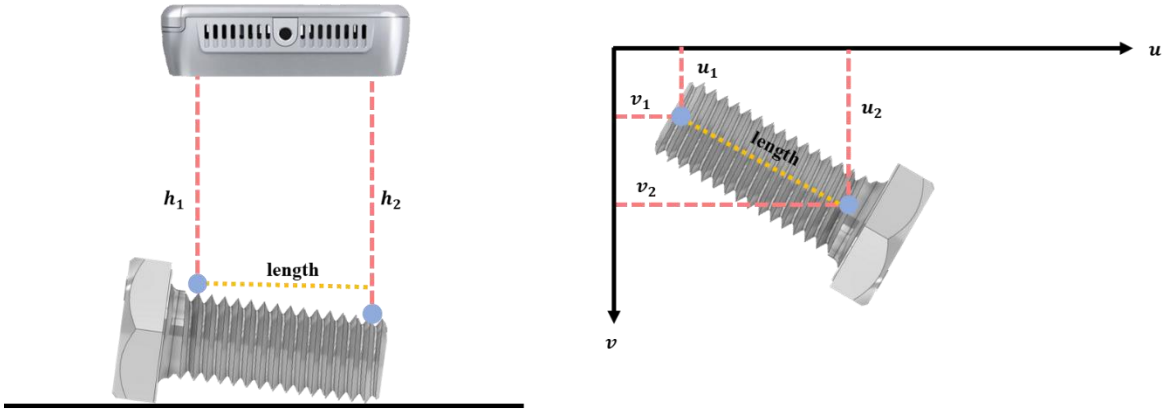


Figure 3.19. Calculation of Ry angle in geometric representation.

For R_x angle calculation, due to the symmetric geometry of the bolt, the team will assign a constant value for R_x if the bolt is near walls of container for collision avoidance and for normal cases R_x will be 0.

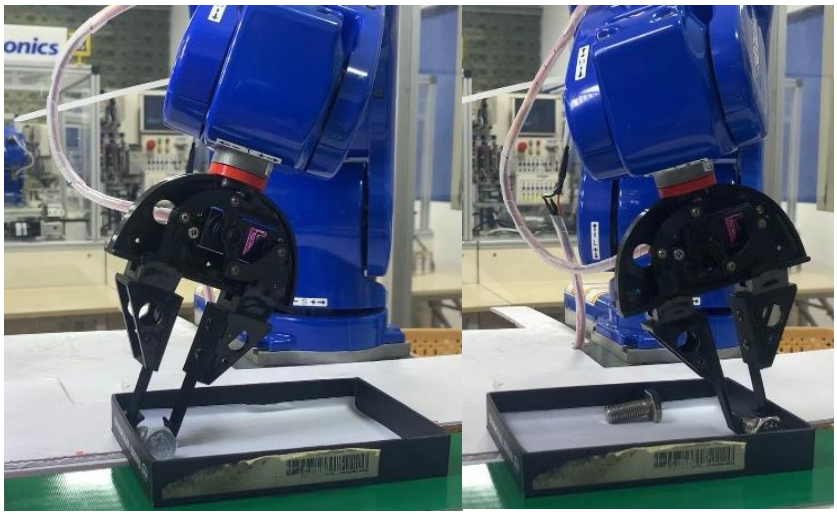


Figure 3.20. Rx angle is modified to avoid collison against container wall.

3.2. System hardware

3.2.1. Overview of system hardware

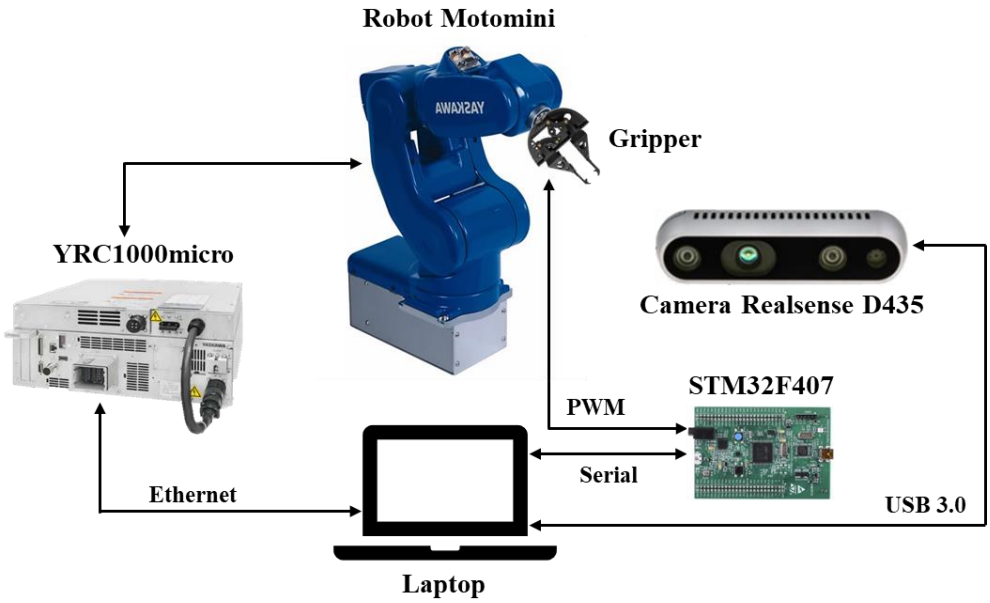


Figure 3.21. Overview of the system hardware.

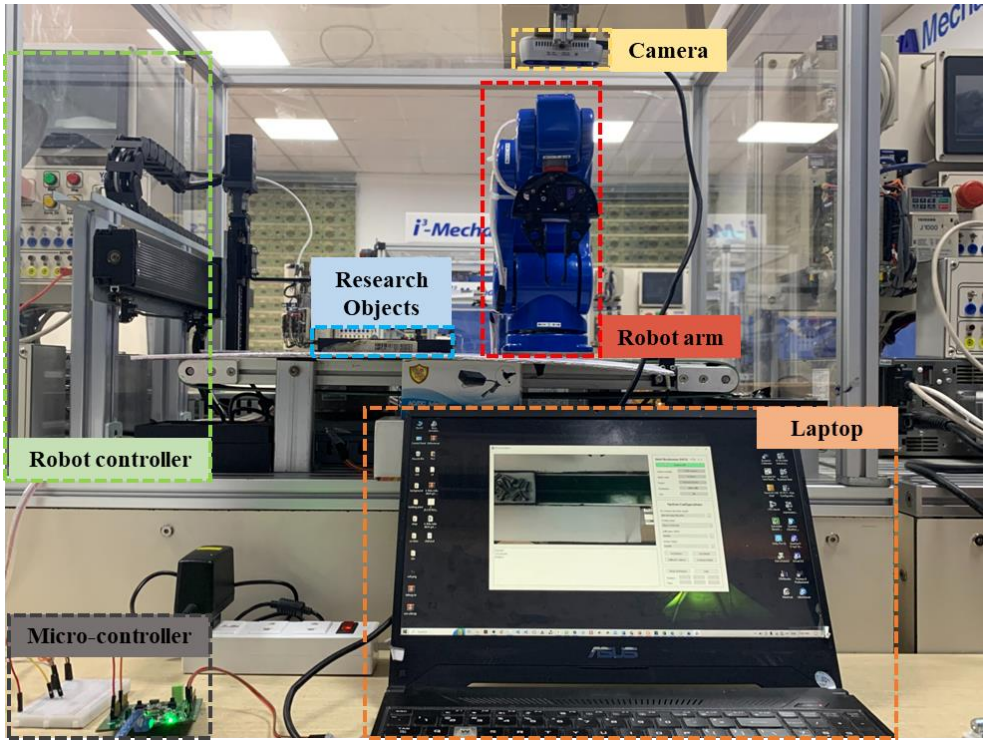


Figure 3.22. Actual layout of system hardware.

The hardware system is built in basic conditions in the 205B3 Yaskawa Robot Laboratory - Ho Chi Minh City University of Technology with the Yaskawa MotoMini Robot and the controller provided by the manufacturer, the main sensor of the program is RealSense D435 camera chosen by the group and some other components to serve the operation of the robot.

3.2.1.1. Motoman MotoMini Robot

The MotoMini robotic arm boasts the highest acceleration in its class, outpacing comparable small robots by 20% in terms of speed. This results in reduced cycle times and enhanced productivity. The incorporation of internal cabling and air lines mitigates interference with other process equipment. Additionally, the availability of multiple mounting options, including surface, wall, and ceiling mounts, facilitates integration into high-density factory layouts. Engineered for exceptional agility and high-speed performance across a broad spectrum of applications, the MotoMini's compact, quiet, and precise operation can elevate small-part processes to new heights.

Figure 3.23. Yaskawa Motoman MotoMini Robot.

The following figures and table represent the technical specifications of MotoMini Robot:

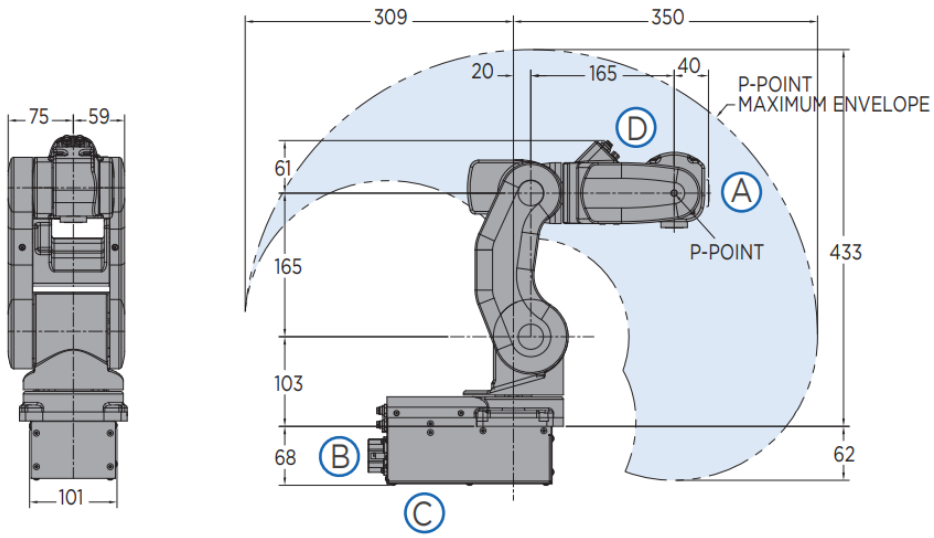


Figure 3.24. Operation dimension of the MotoMini.

Table 3.3. Specification paramemeters of the MotoMini

Axes	Maximum motion range	Maximum speed	Allowable moment	Allowable moment of inertia
	degrees	°/sec	N•m	kg•m²
S	±170	315	-	-
L	+90/-85	315	-	-
U	+120/-175	420	-	-
R	±140	600	0.42	0.00378
B	+210/-30	600	0.42	0.00378
T	±360	600	0.37	0.00299

Mounting Options: Floor, Wall, Tilt, Ceiling
* MLX300 fieldbus cards, I/O cards and vision equipment must be purchased separately from the supplier. All peripherals are programmed using a PLC.

Item	Unit	MotoMini
Controlled axes		6
Maximum payload	kg	0.5
Repeatability	mm	0.02
Horizontal reach	mm	350
Vertical reach	mm	495
Weight	kg	7
Internal user I/O cable		9 conductors w/ ground
Internal user air line		2 x M5 connection
Power requirements		1- or 3-phase; 200/230 VAC
Power rating	kVA	0.5

3.2.1.2. YRC1000micro Controller

The Motoman YRC1000micro is a powerful and precise, ultra-compact robot controller for small robots up to 12 kg payload (GP Series, HC Series, MotoMini and SG-Series Scara, most of them supporting 230V/3-phase power supply), capable to control a robot with up to 2 additional external axes. It's small footprint and lightweight housing is ideal for space-saving stand-alone installation, or it can be installed horizontally or vertically in a 19" rack or a machine control cabinet.

Figure 3.25 YRC1000micro Controller

Technical specifications of the YRC1000micro controller is shown below:

Table 3.4. Specifications of the YRC1000micro Controller.

Specifications Controller YRC1000micro	
Configuration	Open structure
Dimensions	425 (W) x 125 (H) x 280 (D) mm – excluding protrusions
Weight	10.5 kg (without top mount box)
Protection class	IP20
Cooling system	Direct cooling
Ambient temperature	During operation: 0°C to +40°C During storage: –10°C to +60°C
Relative humidity	90 % max. (non-condensing)
Power supply	Single-phase 200/230 VAC (+10 to –15%), 50/60 Hz (±2 %)[1]; three-phase 220/220 VAC (+10 to –15 %), 50/60 Hz (± 2%)
Digital I/Os	External IO (hardware, standard max): 8 inputs / 8 outputs (transistor 100 mA)[2] Standard signal allocation: Gerneral Purpose IO: 1 input / 1 output Specific IO: 7 inputs / 7 outputs
Programming capacity	JOB: 200,000 steps, 10,000 instructions; CIO ladder: 1,500 steps
Expansion slots	PCI express, 2 slots
LAN (Connections to host)	1 (10BASE-T/100BASE-TX)
Safety category	4, PLd

Specifications programming pendant	
Dimensions	152 (W) x 299 (H) x 53 (D) mm
Weight	0.730 kg
Material	Reinforced plastics
Operation device	Select keys, axis keys, numerical/application keys, mode selector switch with keys (mode: teach, play, and remote), emergency stop button, enable switch, SD card interface device, USB port (1 port)
Display	5.7-inch color LCD, touch panel 640 x 480 pixels
IECpProtection class	IP54

3.2.1.3. Intel RealSense D435 Camera

The Intel® RealSense™ D435 offers the widest field of view of all Intel cameras, along with a global shutter on the depth sensor that is ideal for fast moving applications. The D435 is a stereo solution, offering quality depth for a variety of applications. It's wide field of view is perfect for applications such as robotics or augmented and virtual reality, where seeing as much of the scene as possible is vitally important. With a range up to 10m, this small form factor camera

can be integrated into any solution with ease, and comes complete with Intel RealSense SDK 2.0 and cross-platform support.

The integration of an Intel module and vision processor into a compact form factor yields a solution that is well-suited for both research and development and rapid product creation. When coupled with highly customizable software, the D435 offers an economical, lightweight, and powerful solution that facilitates the development of next-generation sensing technologies capable of comprehending and interacting with their environments.

Depth camera D435 is part of the Intel® RealSense™ D400 series of cameras, a lineup that takes Intel's latest depth-sensing hardware and software packages them into easy-to-integrate products. Perfect for developers, makers, and innovators looking to bring depth sensing to devices, the Intel RealSense D435 camera offers simple out-of-the-box integration and enable a whole new generation of intelligent vision-equipped devices.

Figure 3.26. Intel RealSense D435 Camera.

3.2.1.4. MG996R Servo

The MG996R servo motor represents an enhancement over its predecessor, the MG995 motor, in terms of rotational speed, torque, and precision. Capable of rotating up to 180 degrees, the MG996R is primarily utilized in applications that require gripping motion and directional control. The motor's durability is bolstered by its metal alloy gearbox and integrated circuitry that regulates the rotation angle of the motor shaft via pulse width modulation (PWM).

Table 3.5 below presents the key specifications of the MG996R motor:

Table 3.5. Specifications of the MG99R Servo.

Specification	Value
Type	Analog RC Servo
Input voltage	4.8 - 6.0 VDC
Torque	3.5 kg-cm (180.6 ozin) (4.8V-1.5A) 5.5 kg-cm (208.3 ozin) (6V-1.5A)
Rotation speed	0.17 sec / 60 degrees (4.8V, no load) 0.13 sec / 60 degrees (6.0V, no load)
Dimension	40 mm x 20 mm x 43 mm
Net weight	55 g
Pins	Vcc (Red) GND (Brown) PWM (Orange)

In this thesis, the shaft of the MG996R motor is attached to the gripper and is used to control the motion of the gripper based on the rotation angle of the shaft.

3.2.1.5. Gripper

G5 Gripper is preferred for this project due to its compactness, light-weight structure, and reasonable price. Moreover, this type of gripper can support various types of customized claws attached to its fingers. For objects that are small in shape with the shape at grip point smaller than 50 millimeters, the gripper is an ideal choice. The specification of the gripper is showed in Table 3.6 below:

Table 3.6. Specifications of the G5 Gripper.

Characteristics	Value
Length	98 mm
Width	103 mm
Largest gap between two fingers	54 mm

Figure 3.27. G5 Gripper.

Due to the contact surface of the gripper is quite big compared to the bolt M10x25 for gripping without affecting the whole workspace, the team decided to design a more optimal gripping part for the gripper and attach it to the gripper. The 3D part is designed in Autodesk Inventor platform with 30mm length addition to the gripper.

Autodesk Inventor is an engineering design software developed by Autodesk. Autodesk Inventor is computer-aided design (CAD) software developed by Autodesk for 3D mechanical modeling, simulation, visualization and documentation. Inventor enables the integration of 2D and 3D data in a shared environment, creating a virtual representation of the final parts that allows users to verify the its form, fit and function before it is built. Autodesk Inventor includes parametric, direct editing and free-form modeling tools, as well as multi-CAD

translation capabilities in its standard DWG (DraWinG) computer drawings. It uses ShapeManager, Autodesk's proprietary geometric modeling core. This software competes directly with SolidWorks, Solid Edge and Creo.

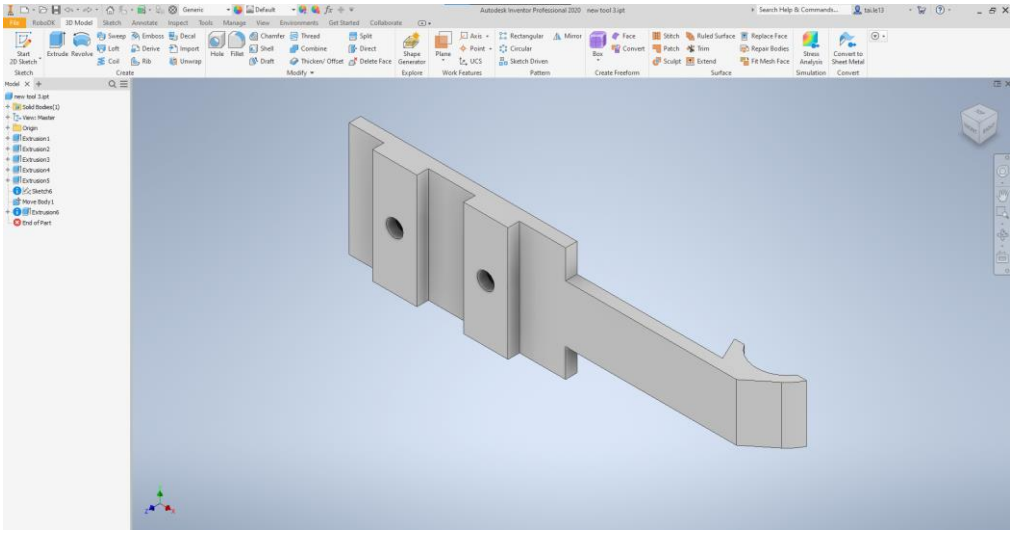


Figure 3.28. Design of the new gripping part in Autodesk Inventor.

Figure 3.29. New gripper parts installed on the G5 Gripper.

3.2.1.6. STM32F407VGTx

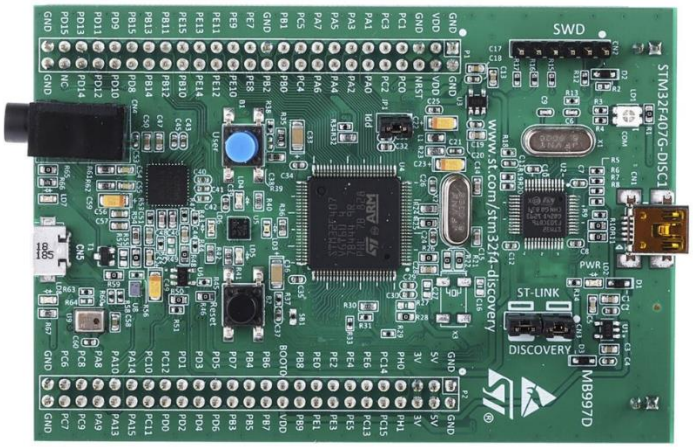


Figure 3.30. STM32F407VGTx Microcontroller

The STM32F407VGTx kit acts as a communication bridge that receives commands from the main computer via serial protocol to control the gripper motion. The STM32F407 microcontroller inside offers high clock speed (168 MHz), PWM timers for motor control, and fast, simultaneous serial communication with the use of its Direct Memory Access (DMA) controllers.

3.2.1.7. Laptop

The computer has the most important task, bearing the responsibility of processing various information from the communication system (commands from people, data from cameras, controlling robots and servo motors to perform tasks) so it needs a computer with a configuration that can handle multiple streams. Therefore, this project needs a laptop with a configuration as presented in Table 3.7.

Table 3.7. Specifications of the laptop.

System Model	Operating System	CPU	GPU	RAM
TUF Gaming FX505DT_FX505DT	Windows 10	AMD Ryzen 7 3750H	NVIDIA GeForce GTX 1650	16GB

3.2.2. Communication between computer and robot

Through high-speed Ethernet communication server function, user can send/receive the YRC1000micro internal data, monitor the manipulator, and control the manipulator by operating from the PC. Connection between PC and robot is made through UDP communication with the utility of the corresponding transmission packet regulated from the manufacturer. There are 2 ports for different usage, which are 10040 and 10041 for sending command, receiving data callback and managing file, respectively.

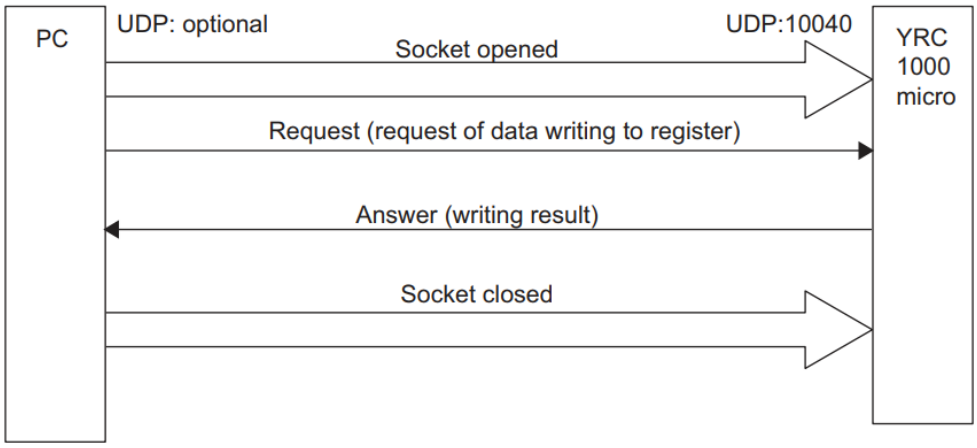


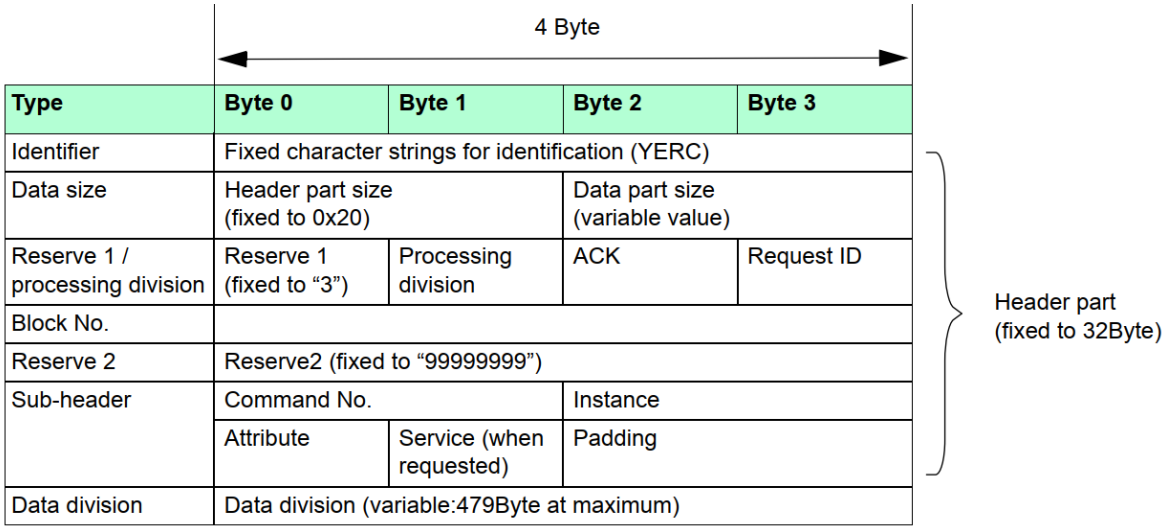
Figure 3.31. Communication between the main computer and the robot controller.

Transmission packet of high-speed Ethernet server function is composed of header part (32 bytes) and data part (flexible, with a maximum of 479 bytes.)

The transmission packet consists of “request”, which transmits the data from the PC to the YRC1000micro, and “answer”, which transmits the data from the YRC1000micro to the PC.

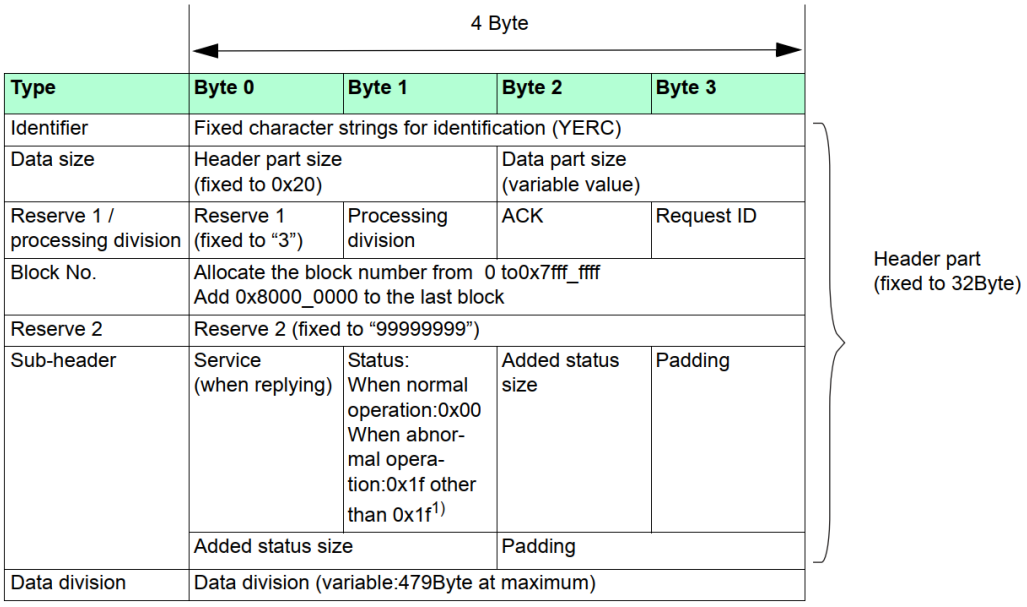
The sub-header setting composition of “request” and “answer” are different. And the setting value of the “answer” varies in accordance with the replying contents.

Followings are the format of each packet.



Type	Byte 0	Byte 1	Byte 2	Byte 3
Identifier	Fixed character strings for identification (YERC)			
Data size	Header part size (fixed to 0x20)		Data part size (variable value)	
Reserve 1 / processing division	Reserve 1 (fixed to “3”)	Processing division	ACK	Request ID
Block No.				
Reserve 2	Reserve2 (fixed to “99999999”)			
Sub-header	Command No.		Instance	
	Attribute	Service (when requested)	Padding	
Data division	Data division (variable:479Byte at maximum)			

Figure 3.32. Data request frame structure (PC to YRC1000micro).



Type	Byte 0	Byte 1	Byte 2	Byte 3
Identifier	Fixed character strings for identification (YERC)			
Data size	Header part size (fixed to 0x20)		Data part size (variable value)	
Reserve 1 / processing division	Reserve 1 (fixed to “3”)	Processing division	ACK	Request ID
Block No.	Allocate the block number from 0 to0x7fff_ffff Add 0x8000_0000 to the last block			
Reserve 2	Reserve 2 (fixed to “99999999”)			
Sub-header	Service (when replying)	Status: When normal operation:0x00 When abnor-mal opera-tion:0x1f other than 0x1f[1)]	Added status size	Padding
	Added status size		Padding	
Data division	Data division (variable:479Byte at maximum)			

Figure 3.33. Data receive frame structure (YRC1000micro to PC).

Table 3.8. Details of the settings for the header

Item		Data size	Settings
Identifier		4Byte	Fixed to "YERC"
Header part size		2Byte	Size of header part (fixed to 0x20)
Data part size		2Byte	Size of data part (variable)
Reserve 1		1Byte	Fixed to "3"
Processing division		1Byte	1: robot control 2: file control
ACK		1Byte	0: Request 1: Other than request
Request ID		1Byte	Identifying ID for command session (increment this ID every time the client side outputs a new command. In reply to this, server side answers the received value.)
Block No.		4Byte	Request: 0 Answer: add 0x8000_0000 to the last packet. Data transmission other than above: add 1 (max: 0x7fff_ffff)
Reserve 2		8Byte	Fixed to "99999999"
Sub-header (request)	Command No.	2Byte	Execute processing by this command. (conforms to "Class" of CIP communication protocol)
	Instance	2Byte	Define SECTION to execute a command. (conforms to "Instance" of CIP communication protocol)
	Attribute	1Byte	Define SUB SECTION for executing a command. Attribute: (conforms to "Attribute" of CIP communication protocol)
	Service (request)	1Byte	Define data accessing method.
Sub-header (answer)	Service (answer)	1Byte	Add 0x80 to service (request).
	Status	1Byte	0x00: normal reply 0x1f: abnormal reply (size of added status: 1 or 2) Other than 0x1f: abnormal reply (size of added status: 0) Refer to *chapter 3.4.1 "Status Code"*
	Added status size	1Byte	Size of added status (0: not specified / 1: 1 WORD data / 2: 2 WORD data)
	Added status	2Byte	Error code specified by added status size For details, refer to *chapter 3.4.2 "Added Status Code"*
Padding		Variable	Reserve area

According to the header data, a communication library is built to connect to the robot, control, and receive signal easily, which can be reused for other versions of the brand controllers. The library includes 3 parts: UDP connection class, YRC1000micro command generator class, and YRC1000micro communication control class.

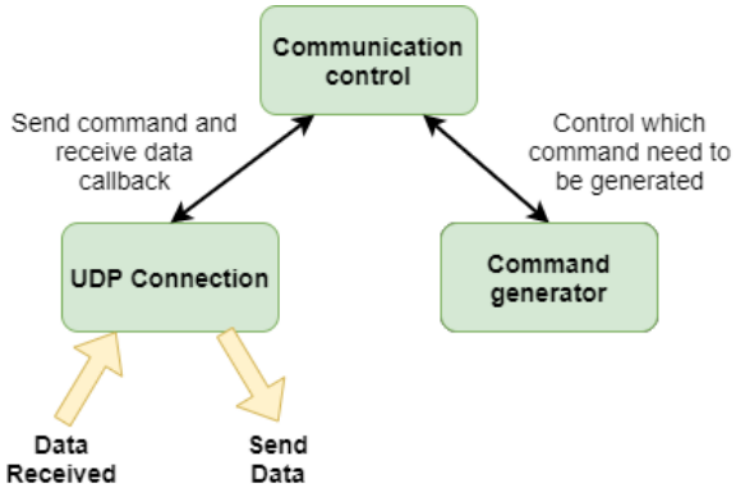


Figure 3.34. Overview of the communication and control library.

- UDP connection: build connection, manage data transmission between PC and controller including receiving command data from Communication control class and send data through generated port. This class utilizes QUdpSocket class provided by Qt to make UDP connection between PC and robot.
- Command generator: automatically called by default every time Communication control class operates, and generate command according to the class. There are different values of some specific variables in the header for different commands.
- Communication control: perform communication management including sending commands, receiving data, command-sending status and robot status from the controller.

Functions built in the YRC1000micro communication control class are presented in Table 3.9 below:

Table 3.9. Built-in Functions of the YRC1000micro class.

Functions	Description
motomanConnect	Connect computer to the robot
motomanDisconnect	Disconnect computer to the robot
motomanOnServo	Turn on the robot servo

motomanOffServo	Turn off the robot servo
motomanReadPosition	Get robot position
motomanReadPulse	Get robot pulse
motomanReadStatus	Get robot running status
motomanMoveCartesian	Move robot to the given Cartesian point
motomanSelectJob	Select robot job in the controller
motomanStartJob	Start the selected job in the controller
motomanTeachPosition	Write position variable to the controller
motomanWriteByte	Write byte variable to the controller
motomanTeachPluralPosition	Write multiple position variables to the controller
motomanReadIO	Read controller IO data
motomanReadControlMode	Read controller mode (teach/play/remote)

3.2.3. Gripper control

This section is divided into 2 parts: building the program on the microcontroller and the PC.

According to the manufacturer's datasheet, a pulse with frequency $f = 50Hz$, or $T = 2000\mu s$ must be fed into the MG996R servo to control the rotation angle. How many degrees the motor shaft rotates depends on the duty cycle of the pulse being fed into the motor.

In order to determine the proper working rotation angle of the motor physically, a voltage of 9VDC and current of 1.5A is supplied to the motor, and the STM32F407VGTx kit is used to generate the PWM signal. The pin connection is presented in table x below:

Table 3.10. Wiring Diagram.

STM32F407	MG996R	Vcc
GND	GND	GND

_	Vcc	$+9V$
PB8	PWM	_

After evaluating, the actual rotation is measured and recorded into table x, as shown below:

Table 3.11. Comparison between the manufacturer suggested and actual duty cycle.

Duty Cycle (Manufacturer)	**Duty Cycle (Actual)**	**Angle**
5%	3%	0
10%	8.75%	90
11.5%	10.5%	120
13%	12%	150
14%	12.88%	170

From Table 3.11, the relationship between the rotation angle and the duty cycle is linearized into the equation:

$$Duty\ Cycle\ (\%) = 11.25 \times Angle\ (degree) + 640$$

For grabbing motion of the servo, the duty cycle of the control pulse is modified with a fixed value of 990 us, whereas for the releasing motion, the duty cycle is fixed to 1350 us.

Interface between computer and microcontroller:

In this thesis, a CP2102 UART to USB adapter is used in order to establish connection and transfer data between the microcontroller and the main computer.

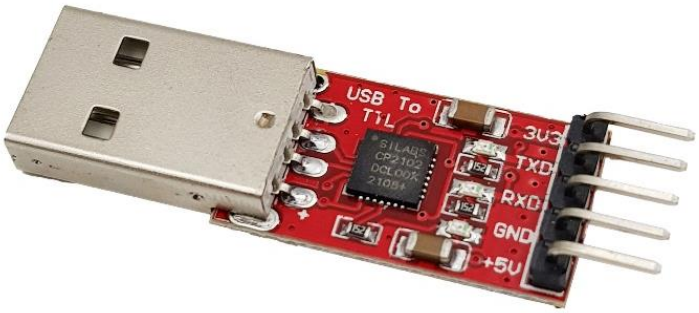


Figure 3.35. CP2102 UART to USB adapter.

Building the program on the microcontroller board

The following peripherals included in the board are used for the program:

- UART4: Configured to transmit at 115200 baud rate, 8-bit data, 1 stop bit and no parity. Port PA1 is configured to be the RX pin and PA0 is the TX pin. In addition, transmission through DMA is enabled for this UART peripheral.
- Timer 6: configured as a 16-bit timer, to export PWM pulses. Uses port PB8.

The program loaded to the microcontroller, on receiving signal from the computer, will collect the data in the UART DMA interrupt, then perform actions according to the defined convention. After finishing controlling the servo, the microcontroller then sends a series of data back to the computer to signal completion.

The data frame sent by the microcontroller to the computer is defined as below:

Table 3.12. Transmission frame structure from the microcontroller to the PC..

Byte	Content	Value
0 – 1	Header	36055
2	Command	Command received from the computer

3	Error	Error = 0: No error Error = 1: Invalid Header Error = 2: Invalid Command Error = 3: Invalid Value Error = 4: Invalid CRC
4- 5	CRC check	CRC = Header + Command + Error Returns error if CRC value doesn't

Building the program on the computer:

The computer program communications with the microcontroller, by sending commands (grab, release, etc.) and receiving feedback after controller is finished. Similarly, the computer will also send a series of bytes to the microcontroller via UART.

The data frame sent by the computer to the microcontroller is defined as below:

Table 3.13. Transmission frame structure from the PC to the microcontroller.

Byte	Content	Value
0 – 1	Header	36055
2	Command	Command = 1: Grab Command = 2: Release Command = 3: Manual Mode (Custom degree)
3	Value	Degree (in Manual Mode) Default is 1 for Grab and Release
4- 5	CRC check	CRC = Header + Command + Value Returns error if CRC value doesn't match

A failsafe mechanism is also added to the program, and is activated when the connection between the microcontroller and the PC. During such event, the program will show a pop-up warning, and stop the program on users' prompt, and at the same time bring the robot back to its initial position.

3.2.4. Hand-eye calibration

To convert the camera coordinates to the robot coordinates, it is necessary to determine the 4x4 homogeneous transformation matrix. The determination of this matrix also depends on how the camera is mounted relative to the robot. There are two ways to mount the camera: fixed at a point and mounted on the robot arm. For the case of this project, the camera is fixedly installed on the frame, so the calibration calculation is based on the unchanged position of the ArUco tag relative to the end-effect of the robot, as seen in Figure 3.36.

Figure 3.36. Hand-eye calibration demonstration.

Based on the constant position of the tag with respect to the end-effector of the robot ${}^{e}T_{t} = const$, transformation matrices ${}^{e}T_{b}, {}^{c}T_{t}$ are calculated – transformation matrix of the robot base frame with respect to the end-effector frame and tag frame with respect to the camera frame, respectively – in order to find the calculate transformation matrix ${}^{b}T_{c}$ – transformation matrix of the camera frame with respect to the robot base frame. For the best result and reduction of the

calibration error, 5000 samples are collected. Formulas related to the calibration process as below:

$$^{e}T_{t} = {}^{e}T_{b}{}^{(1)} \times {}^{b}T_{c} \times {}^{c}T_{t}{}^{(1)} = {}^{e}T_{b}{}^{(2)} \times {}^{b}T_{c} \times {}^{c}T_{t}{}^{(2)} = const$$

$$({}^{e}T_{b}{}^{(2)})^{-1} \times {}^{e}T_{b}{}^{(1)} \times {}^{b}T_{c} = {}^{b}T_{c} \times {}^{c}T_{t}{}^{(2)} \times ({}^{c}T_{t}{}^{(1)})^{-1}$$

$$A_i X = X B_i$$

With:

$$\begin{cases} A_i = ({}^{e}T_{b}{}^{(2)})^{-1} \times {}^{e}T_{b}{}^{(1)} \\ B_i = {}^{c}T_{t}{}^{(2)} \times ({}^{c}T_{t}{}^{(1)})^{-1} \\ X = {}^{b}T_{c} \end{cases}$$

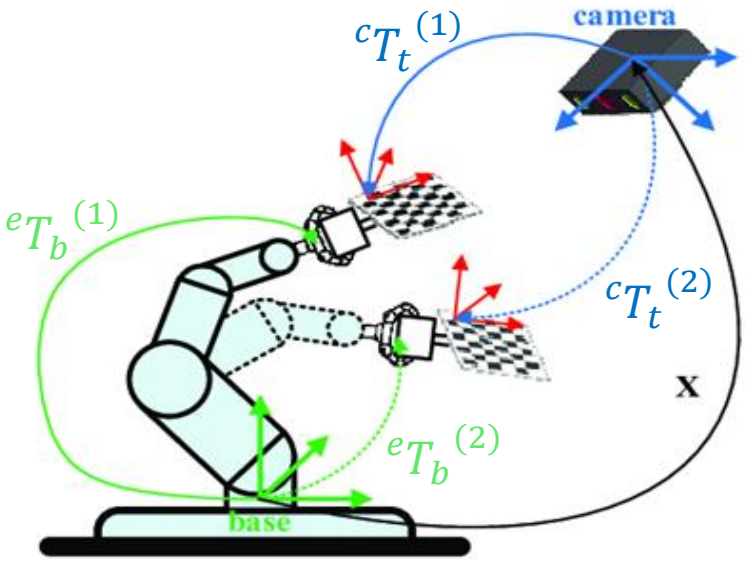


Figure 3.37. Hand-eye calibration transformation diagram.

OpenCV library is used to detect the ArUco tag and MATLAB platform is used to perform calculation because of the large number of samples to obtain highest accuracy as possible.

3.2.5. Grasping point transformation

From result of the hand-eye calibration $^{b}T_{c}$ and pose estimation $^{c}T_{o}$, the transformation matrix of the object with respect to object can be obtained as the below:

$$^{b}T_{o} = {}^{b}T_{c} \times {}^{c}T_{o} \tag{3.11}$$

Before grasping the object, the robot needs an approaching point at a distance d away from the grasping point:

$$ {}^{b}T_{approaching} = {}^{b}T_{o} \times \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & -d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.12} $$

Both approaching point and grasping point will be calculate inverse kinematics to make sure that they are inside the workspace of the robot.

Conventions for the axes of the object are as follows:

- ❑ X axis will be along the body of the bolt.
- ❑ Y axis will be symmetric axis of the bolt.
- ❑ Z axis will be approaching direction of the gripper to object.

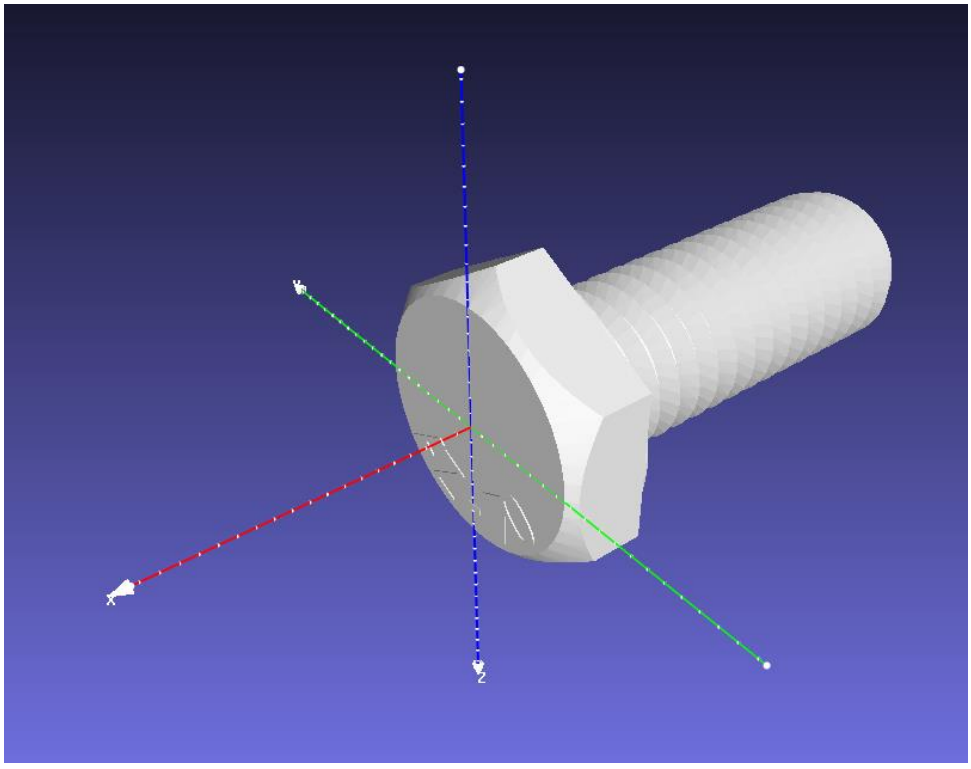


Figure 3.38. Coordinate frame of a M10x25 bolt.

3.3. System software

The system software will include the interface (GUI) and the main algorithm of the bin picking system running under the GUI. The interface is the closest thing to the user of the program, it is the first thing that users come into contact with the program so it

needs to be noticed in terms of aesthetics, as well as layout, group choose to use the following tools and functions (Figure 3.39) of Qt to design the interface.

3.3.1. Widgets

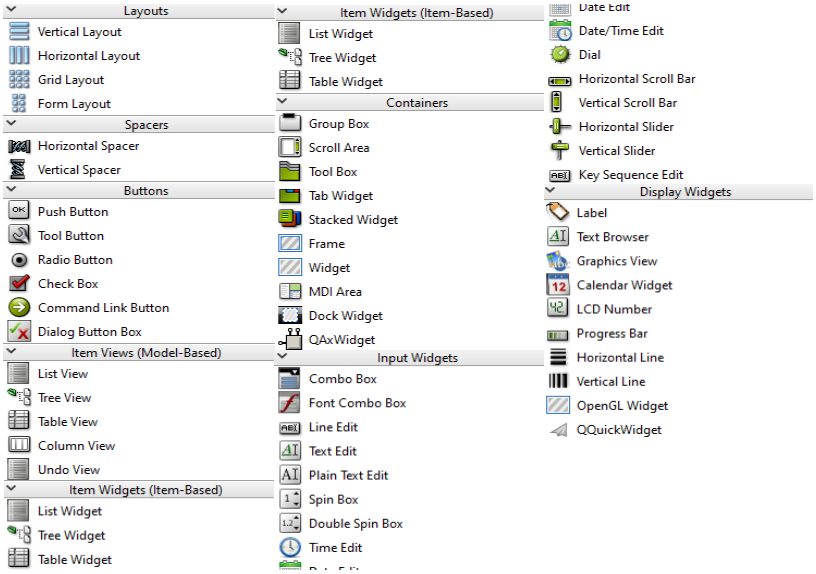


Figure 3.39. Widget menus in Qt Designer.

The main functions of the blocks that are used mostly:

- Label: In addition to being used for annotation and naming of objects (tabs, fields, boxes, etc.), frames are also created to display 2D images from the camera along with the object classification results of the deep learning model (YOLOv5).
- Push Button: used to design push button in the GUI.
- Line Edit: create box to display or input information.

3.3.2. Properties

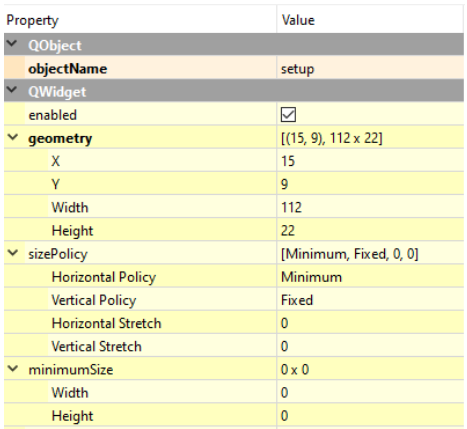


Figure 3.40. Widget properties in Qt Designer.

Each object on the Qt interface is equipped with a multi-functional attribute set, usually used by groups to change the size, status, and display content of objects. In particular, there is an “objectName” attribute used in programming related to that object and the “Palette” attribute allows the group to color-coordinate the interface with a color set developed by developers beforehand so that programmers can choose.

3.3.3. Main algorithms of the software

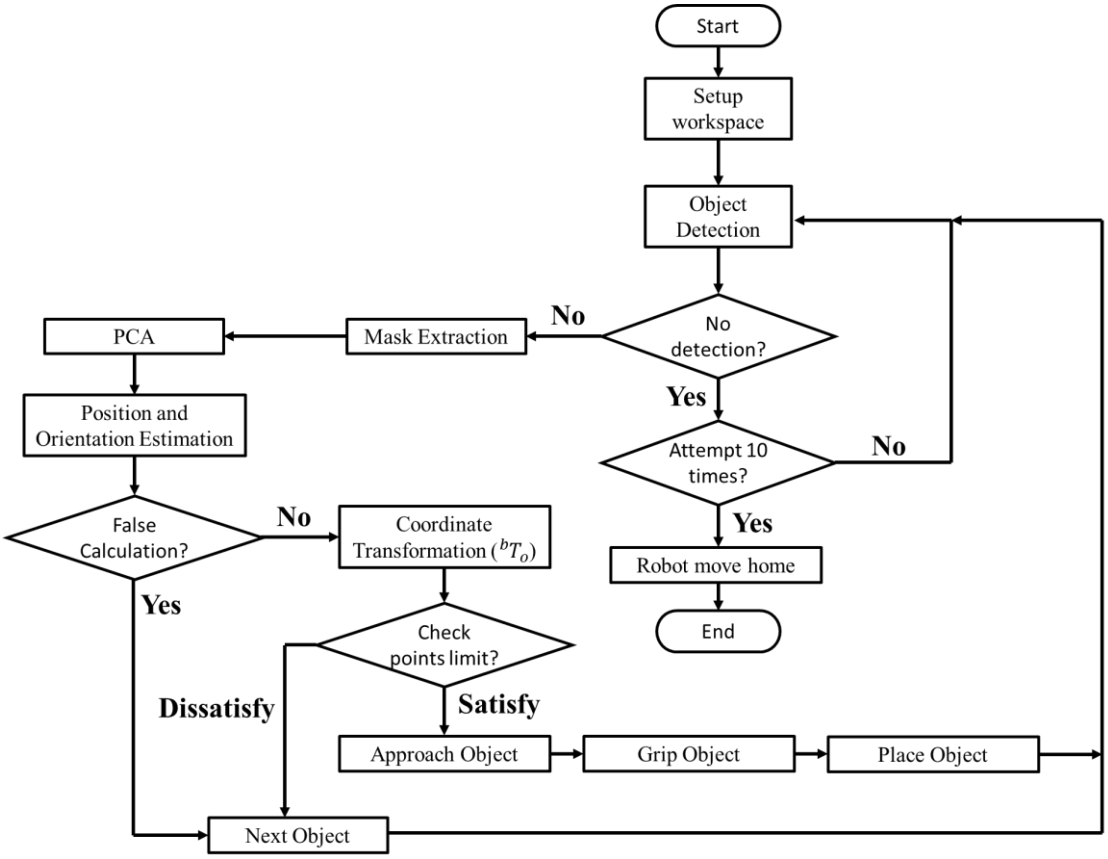


Figure 3.41. Main algorithm of bin picking system.

First of all, the user will use the built function button “Setup Workspace” in the GUI to select the workspace or the container of the objects. After that, the system will start running.

For the beginning stage of the system, the camera will send the RGB and depth frames to the model deep learning YOLOv5 to detect the object. If the object is

detected, the mask and information of detected object will be extracted to process PCA and 6D pose estimation.

During the process, if there is any false calculation, the system will move to detect next object to process it. If not, the calculated point including 6D pose data will be transformed from ${}^{c}T_{o}$ – the transformation matrix of the object with respect to the camera - to ${}^{b}T_{o}$ - the transformation matrix of the object with respect to the robot base. The final transformation matrix ${}^{b}T_{o}$ will be convert into Cartesian point as (x, y, z, Rx, Ry, and Rz). From the calculated grasping point, the approaching point is calculated also with a constant distance d away from it. Those points are then checked to make sure they are in the operation workspace of the robot. If the inspection is satisfied, the computer will send those points to the controller to move the robot to the grasping point to grip and place the object to the pre-defined area.

The system continue to detect the next object untill there is no object left to be gripped, which means the system has made 10 attemps to detect but there is no detection, the system will move the robot to home position and stop the system.

Chapter 4. THESIS RESULTS

4.1. Detection and segmentation results

4.1.1. Evaluation criteria

To evaluate the YOLOv5 deep learning model, the team will use metrics such as Precision, Recall, and mean Average Precision (mAP). These are commonly used evaluation metrics for object detection models. However, it is first necessary to understand the concept of Intersection over Union (IoU), which plays a crucial role in determining the aforementioned evaluation metrics.

4.1.1.1. Intersection over Union (IoU)

Intersection over Union (IoU) is a metric used to evaluate the accuracy of predicted bounding boxes in relation to ground truth bounding boxes. Predicted bounding boxes are generated by a model after its prediction process, while ground truth bounding boxes are manually created to accurately identify the location of objects in an image. IoU is calculated as the ratio of the area of overlap between the predicted and ground truth bounding boxes to the area of their union. The area of overlap represents the intersection of the two bounding boxes, while the area of union represents their combined area. This metric provides a measure of how well the model's predictions align with the true locations of objects in an image and is commonly used in object detection tasks.

4.1.1.2. Precision

Precision is a metric used to evaluate the accuracy of a model's object predictions. It is calculated as the ratio of correct object predictions made by the model to the total number of times the model made predictions for that object. In other words, precision measures how many of the model's predictions were actually correct out of all the predictions it made for a given object. A high precision value indicates that the model is accurately predicting the presence of the object in question.

$$Precision = \frac{True\_Positives}{True\_Positives + False\_Positives} \quad (4.1)$$

In which:

$True_Positive$s is the number of correct predictions by the model.

$True\_Positives + False\_Positives$ is the total number of predictions by the model.

4.1.1.3. Recall

Recall is a metric used to evaluate the accuracy of a model's object predictions. It is calculated as the ratio of correct object predictions made by the model to the total number of objects that need to be predicted. Alternatively, recall measures how many of the objects that should have been predicted by the model were actually predicted correctly. A high recall value indicates that the model is accurately identifying the presence of the object in question.

$$Recall = \frac{True\_Positives}{True\_Positives + False\_Negatives} \quad (4.2)$$

Where:

$True\_Positives + False\_Negatives$ represents the total number of objects that is supposed to be predicted by the model.

The evaluation of $True\_Positives$ is based on the Intersection over Union (IoU) metric with a predefined threshold (where $True\_Positives$ has IoU $\geq IoU_{threshold}$).

4.1.1.4. Mean Average Precision (mAP)

This is the result of calculating the average Precision with different IoU thresholds. There are two popular types of mean Average Precision (mAP): mAP@50 ($IoU_{threshold} \geq 0.5$) and mAP@50-95 (Precision with $0.5 \leq IoU_{threshold} \leq 0.95$, with a step size of 0.05). The mAP@50 measures the Precision at a single IoU threshold of 0.5, while mAP@50-95 measures the average Precision across multiple IoU thresholds ranging from 0.5 to 0.95 with a step size of 0.05.

4.1.2. Evaluation results and comments

```
segment/val: data=/content/yolov5/screw-detection-17/data.yaml, weights=['/content/drive/MyDrive/exp2/weights/best.onnx'], batch_size=32, imgsz=640
YOLOv5 🚀 v7.0-163-g016e046 Python-3.10.11 torch-1.11.0+cu102 CUDA:0 (Tesla T4, 15102MiB)

Loading /content/drive/MyDrive/exp2/weights/best.onnx for ONNX Runtime inference...
Forcing --batch-size 1 square inference (1,3,640,640) for non-PyTorch models
val: Scanning /content/yolov5/screw-detection-17/valid/labels.cache... 107 images, 0 backgrounds, 0 corrupt: 100% 107/107 [00:00<?, ?it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95)     Mask(P          R      mAP50  mAP50-95): 100% 107/107
                   all        107        402      0.966      0.982      0.988      0.822      0.963       0.98      0.987       0.69
Speed: 0.5ms pre-process, 24.5ms inference, 0.9ms NMS per image at shape (1, 3, 640, 640)
```

Figure 4.1. Evaluation process of YOLOv5 on Google Colab.

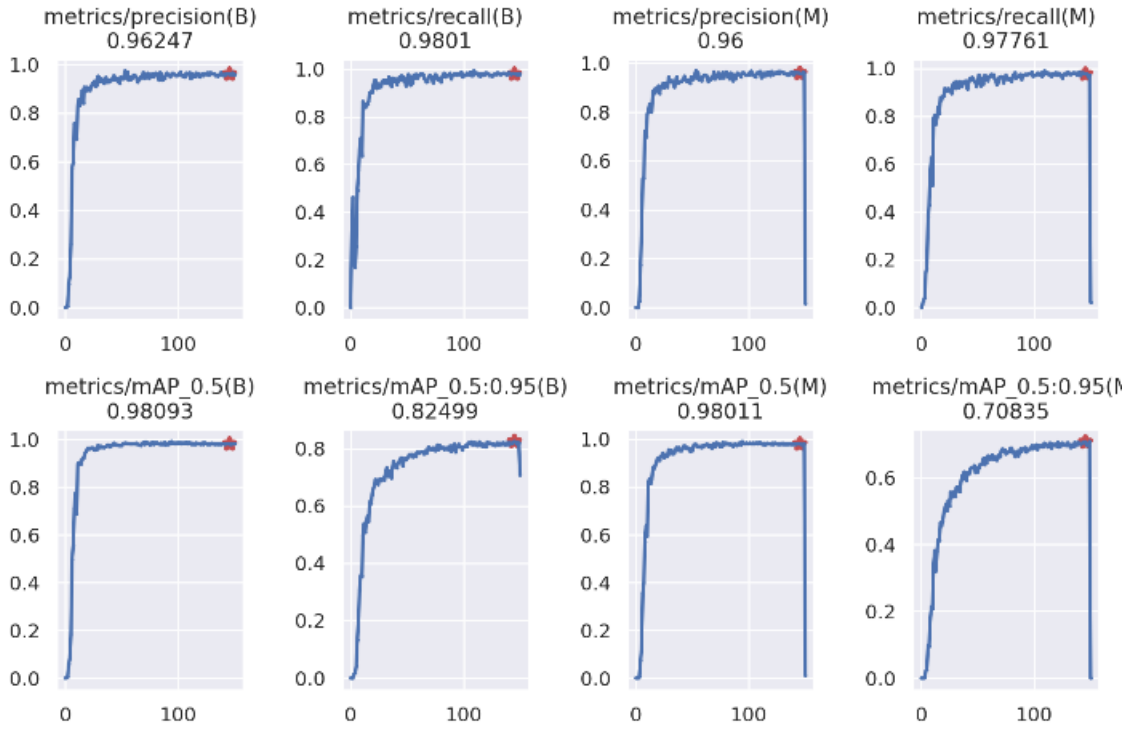


Figure 4.2. Plots of evaluation metrics of YOLOv5 during the training process.

Table 4.1. Evaluation results of different metrics of YOLOv5

Type	Precision (P)	Recall (R)	mAP @50	mAP @50-95
Bounding Box	0.966	0.982	0.988	0.822
Mask	0.963	0.98	0.987	0.69

An evaluation of the model was conducted using a set of 107 images. Overall, the results indicate that the model performs well in practice, as evidenced by high values for evaluation metrics such as Precision and Recall (evaluated at an

$IoU_{threshold}$ of 0.6), mAP@50, and mAP@50-95. However, for mask detection, the mAP@50-95 only reaches 0.69, indicating the performance is not up to standard and further improvements are necessary.

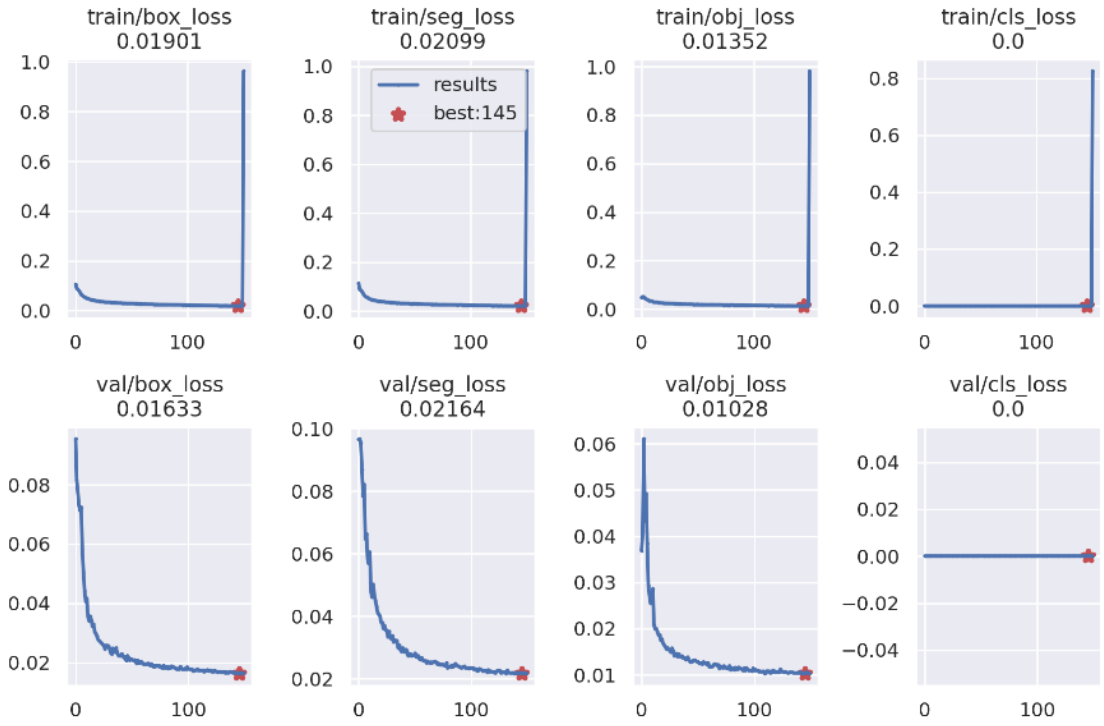


Figure 4.3. Plots of loss functions of YOLOv5 during the training process

An analysis of Figure 4.3 reveals that during the training process, there is a sharp decrease in the values of the loss functions in the initial iterations. As training progresses, these values gradually approach a certain limit and eventually plateau. This plateauing of the loss function values indicates that the training has reached a state of stability and that further training is unlikely to result in significant improvements in model performance.

Figure 4.4. Detection results of YOLOv5 on actual objects in different backgrounds

The object recognition model exhibits the ability to accurately identify objects that are rotated in various directions and are not obscured by other foreign objects. The model can also detect objects in various backgrounds. This performance aligns with the initial goals established by the team, which aimed to develop a model capable of recognizing objects in complex environments. Furthermore, as expected, the model is unable to recognize foreign objects or objects that are positioned below other object. However, the model is, by intention, trained to ignore certain pose of the object, specifically when the object is placed vertically.

4.2. Hand-eye calibration results

4.2.1. Evaluation method and ground truth data

For evaluating the hand-eye calibration, the team will base on Equation 4.3, for ${}^{c}T_{o}$ and ${}^{b}T_{o}$, the object in the formula will be replaced with the ArUco tag (t)

of the OpenCV library, so the formula for transformation matrix of the tag with respect to the base will be:

$$^{b}T_{t} = {}^{b}T_{c} \times {}^{c}T_{t} \tag{4.3}$$

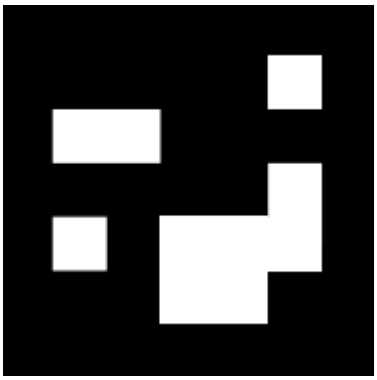


Figure 4.5. ArUco marker.

By using the OpenCV library and intrinsic parameters provided by the manufacturers, the RealSense D435 camera can detect and obtain the position and orientation of the ArUco tag precisely. The ArUco tag used is the 5x5_250_ID1 tag, which can be generated easily from the OpenCV library.

$^{b}T_{c}$ is the hand-eye calibration result and $^{c}T_{t}$ is the transformation of the tag with respect to the camera, so $^{b}T_{t}$ will be the final actual result of the tag with respect to the robot base. This transformation matrix will be then converted into the Cartesian value set $[x\ y\ z\ Rx\ Ry\ Rz]$ based on Roll-Pitch-Yaw angle using Equation 2.9.

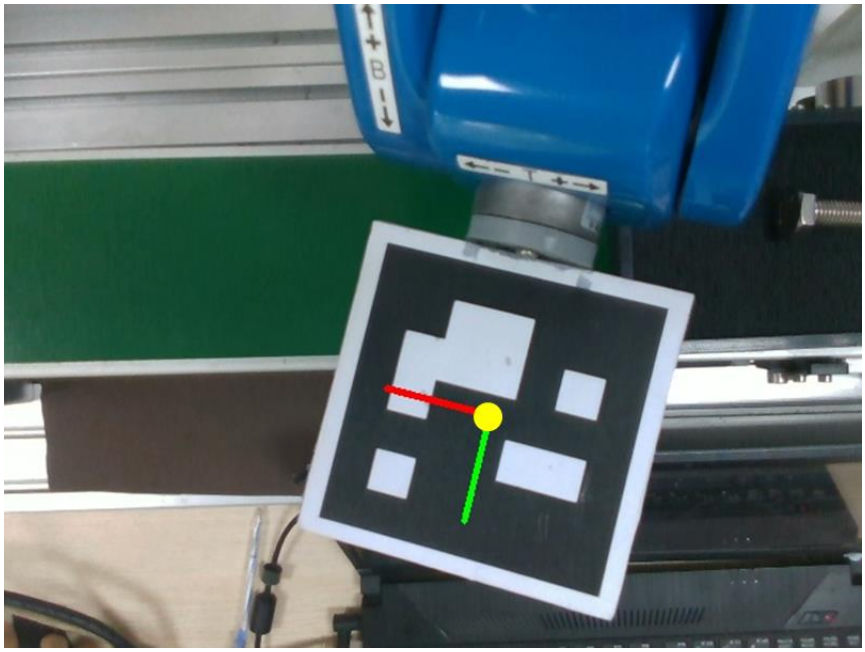


Figure 4.6. TCP of the robot on the center of the ArUco marker.

For the ground-truth ${}^{b}T_{o-groundtruth}$, the team will install the ArUco tag into the end-effector of the robot and set the tool or the tool center point (TCP) of the robot on the teach pendant to be the center of ArUco tag. And the Cartesian value set from the robot will be retrieved through the UDP communication with the robot controller and become the ground-truth data.

From the ground-truth data and the calculated result, the team will calculate the error by the difference between them.

$$Error = Ground\ truth - Result \tag{4.4}$$

4.2.2. Evaluation results and comment

The robot is controlled to move to 50 different positions and get the data at those positions.

Table 4.2. Error evaluation results of hand-eye calibration.

	Errors (mm-degree)					
	x	**y**	**z**	**Rx**	**Ry**	**Rz**
1	-0.19	-0.91	-3.71	-0.94	1.01	-0.28

2	-0.04	-0.72	-4.85	-0.39	-0.38	0.03
3	-0.92	-0.3	-4.51	-0.32	1.49	-0.02
4	-0.96	-0.08	-1.94	-0.12	0.33	-0.3
5	-0.53	0.06	1.44	0.11	0.91	-2.37
…	…	…	…	…	…	…
45	1.68	-0.31	-2.99	-0.61	1.27	-1.87
46	0.38	-2.16	-3.95	-2.74	1.9	-1.03
47	-0.62	-0.41	-3.81	-1.34	0.2	-0.61
48	-0.16	-0.92	-4.44	-1.41	0.01	-0.16
49	0.66	1.61	-4.92	-0.14	0.95	1.32
50	0.07	2.63	-3.51	-0.42	1.5	0.7
Mean	**0.62**	**-0.79**	**-1.98**	**-0.72**	**0.96**	**-1.21**

The mean errors of the hand-eye calibration are 0.62mm, −0.79mm, −1.98mm, −0.72°, 0.96°, and −1.21° for x, y, z, Rx, Ry, Rz, respectively.

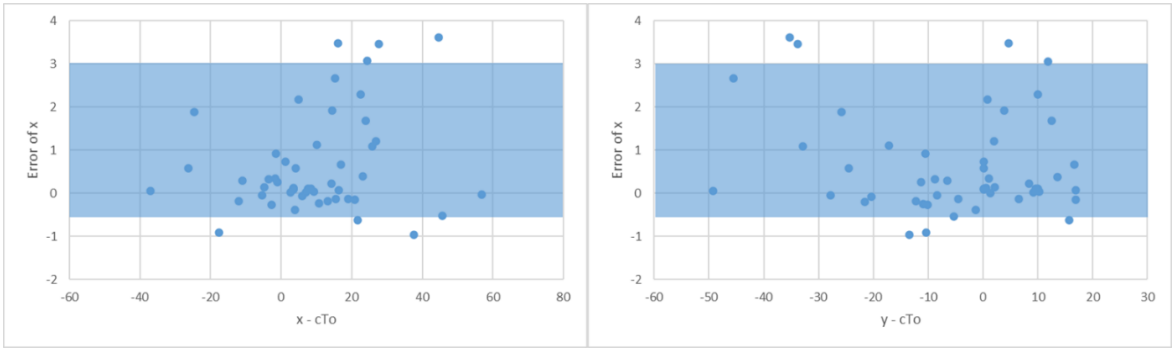


Figure 4.7. Scatter plots of the distribution of x error with respect to x and y coordinate.

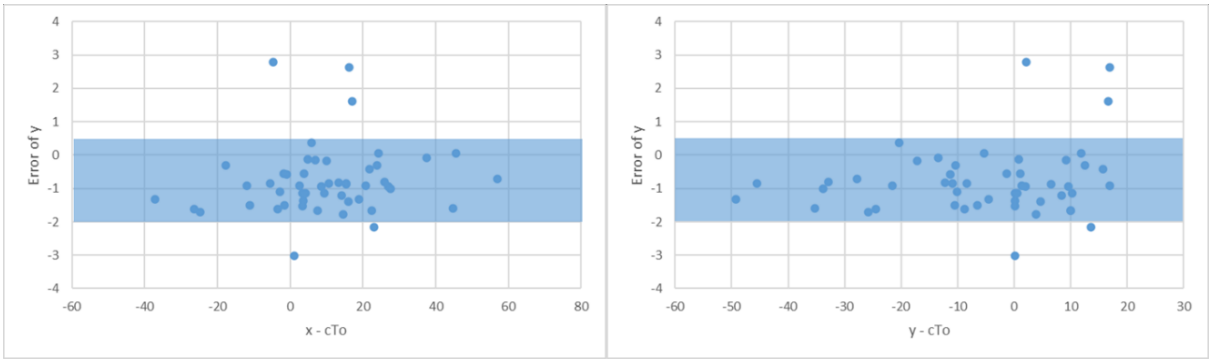


Figure 4.8. Scatter plots of the distribution of y error with respect to x and y coordinate.

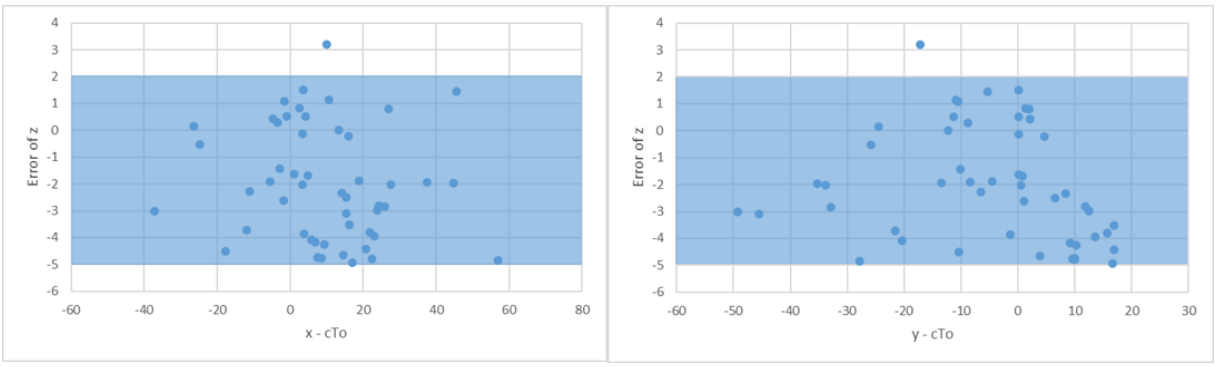


Figure 4.9. Scatter plots of the distribution of z error with respect to x and y coordinate.

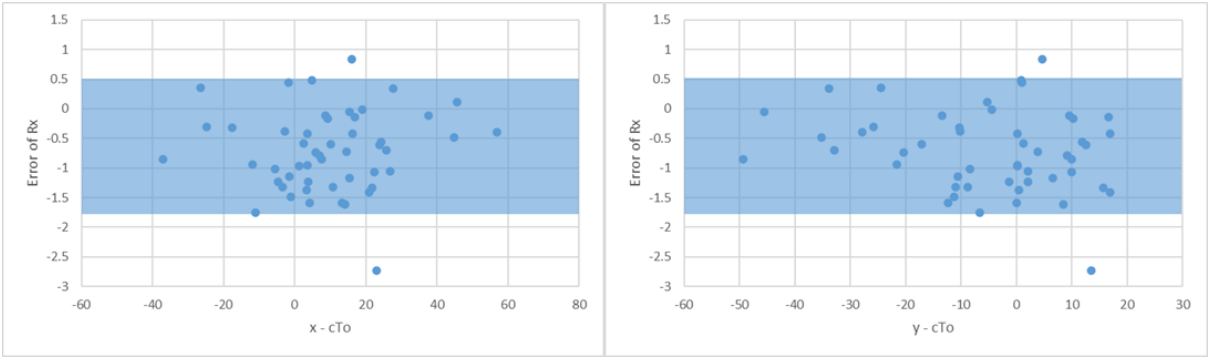


Figure 4.10. Scatter plots of the distribution of Rx error with respect to x and y coordinate.

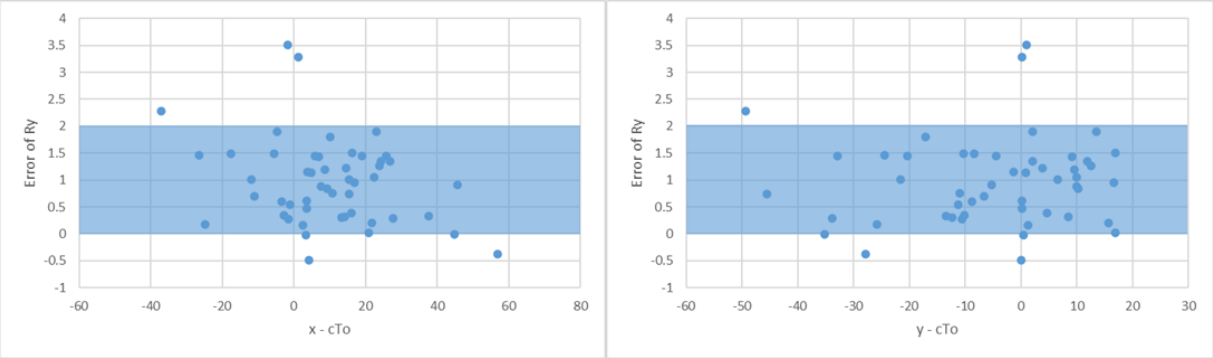


Figure 4.11. Scatter plots of the distribution of Ry error with respect to x and y coordinate.

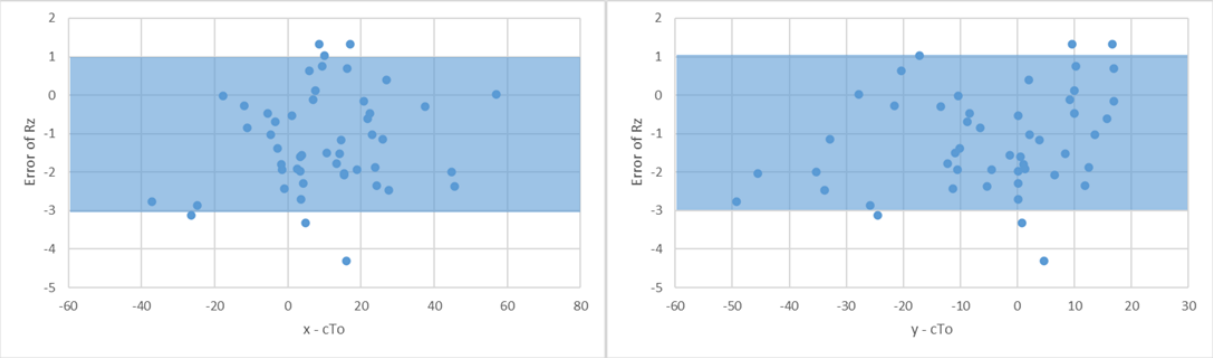


Figure 4.12. Scatter plots of the distribution of Rz error with respect to x and y coordinate.

From these plots shown above, it can be seen that most of the errors are in an acceptable range, particularly approximately $[-3\ \text{mm}, +3\ \text{mm}]$ for position and $[-2°, +2°]$ for rotational angles. However, the error of z is slightly higher than 2 other position values $[-5\ \text{mm}, +2\ \text{mm}]$, which can still be acceptable since the radius of the bolt's body is 5 mm.

4.3. Position and orientation estimation results

4.3.1. Evaluation methods and evaluation metric

4.3.1.1. Evaluation metric

After a few research, the team decided to use 5cm-5deg metric to be the evaluation metric.

5cm-5deg metric was proposed by Shotton and his partners [11]. Based on this metric, the estimation result is correct when the translation and rotation error is below $(5\ cm, 5°)$. However, in the Shotton's research, the camera workspace is

$2\,m^3$, while in the case of the team, the workspace has the size of $0.5m \times 0.5m \times 0.4m = 0.1m^3$. Therefore, the team propose that the estimation is correct when the translation and rotation error is below ($1\ cm, 5°$).

$$Accuracy\ (\%) = \frac{N_{correct\ estimation}}{N_{total\ samples}} \times 100 \quad (4.5)$$

In addition, the team also calculates the error of the estimation result.

$$Error = Ground\ truth\ - Estimation \quad (4.6)$$

4.3.1.2. Evaluation methods

Based on the formula ${}^bT_o = {}^bT_c \times {}^cT_o$, the grasping point or the final result of the estimation is ${}^bT_o$. Therefore, the team will evaluate both the estimation result of ${}^cT_o$ and ${}^bT_o$, which are the estimation of the object with respect to the camera and the robot base, respectively.

- Estimation of the object with respect to the camera:

Similar to the hand-eye calibration evaluation mentioned above, the team will continue to use the ArUco tag of OpenCV library for the evaluation. However, for this evaluation, the team will attach the bolt at a constant distance to the tag, so the formula for ${}^cT_{o\ ground-truth}$ is as below:

$${}^cT_{o\ ground-truth} = {}^cT_t \times {}^tT_o \quad (4.7)$$

Which

$${}^tT_o = \begin{bmatrix} -1 & 0 & 0 & o_x \\ 0 & 0 & -1 & o_y \\ 0 & -1 & 0 & o_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With o_x, o_y, o_z are the translation of the tag to the object.

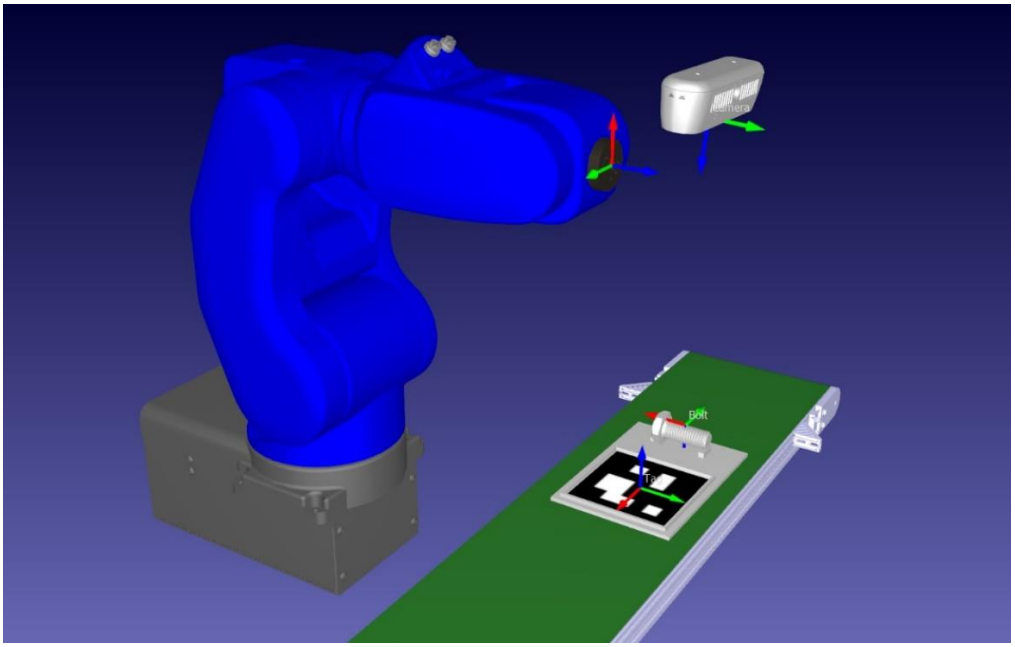


Figure 4.13. Overview of the actual setup of calibration hardware system.

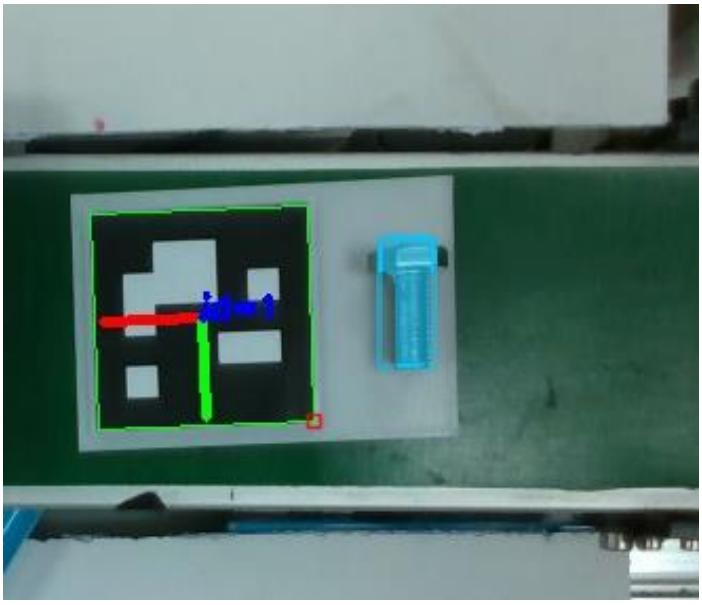


Figure 4.14. Detection result of ArUco marker with the evaluated object.

❑ Estimation of the object with respect to the robot base:

For this evaluation, the team will use the robot itself to be the ground truth data. The team designs a robot tool for the object to be attached to it. When the robot moves, the object will translate and rotate the same as robot in

position translation rotation angles (x, y, z, R_x, R_y, and R_z). The Cartesian robot position variable retrieved from the controller is the ground truth data.

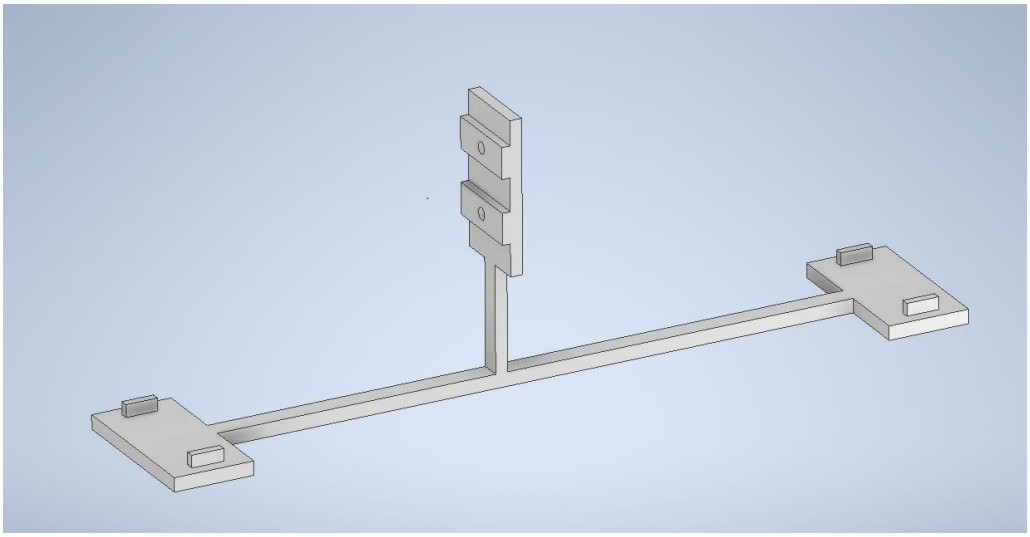


Figure 4.15. Custom tool for evaluation of object w.r.t the robot base.

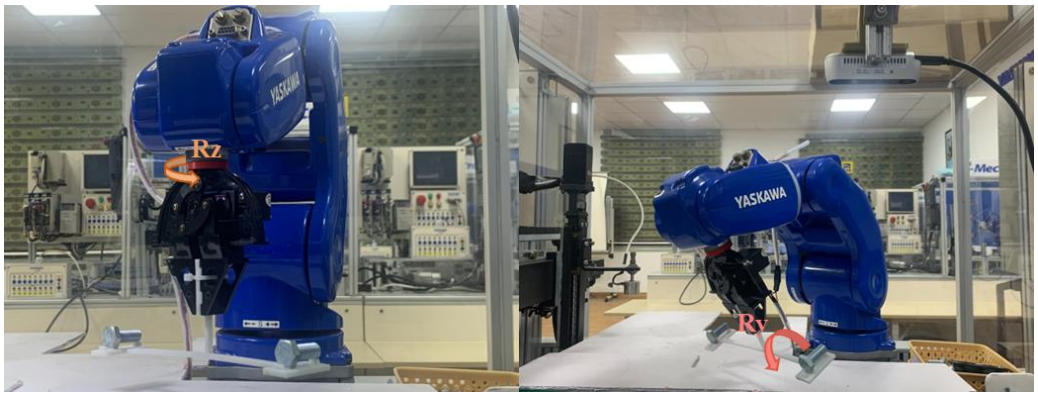


Figure 4.16. Demonstration of evaluation methods in different rotation angle.

4.3.2. Evaluation results and comments

- *Estimation of the object with respect to the camera:*

The team controls to move the robot to 50 different positions and get the data at those positions.

Table 4.3. Error evaluation results of the estimated object w.r.t. the camera

	Error (mm-degree)				
	x	**y**	**z**	**Ry**	**Rz**
1	1.09	-2.34	-1.55	2.15	0.96

2	2.21	-1.25	-1.51	-0.56	-4.92
3	-1.38	-3.83	0.87	2.24	-3.48
4	0.07	-2.46	-2.46	3.16	-1.11
5	-2.99	1.74	6.39	2.97	0.36
…	…	…	…	…	…
46	-0.79	0.12	-1.46	2.05	-2.91
47	1.22	-1.54	0.51	-0.42	-3.72
48	-0.85	0.29	-0.75	-0.17	-2.19
49	2.64	1.66	0.28	-1.28	-6.17
50	-1.56	0.99	2.76	-2.44	0.99
Mean	**-0.07**	**-0.24**	**0.49**	**0.19**	**-1.76**

Table 4.4. Accuracy evaluation results.

Object	No. Execution	Accuracy (%)	Average execution time (s)
Bolt M10x25 (mm)	50	90	0.35

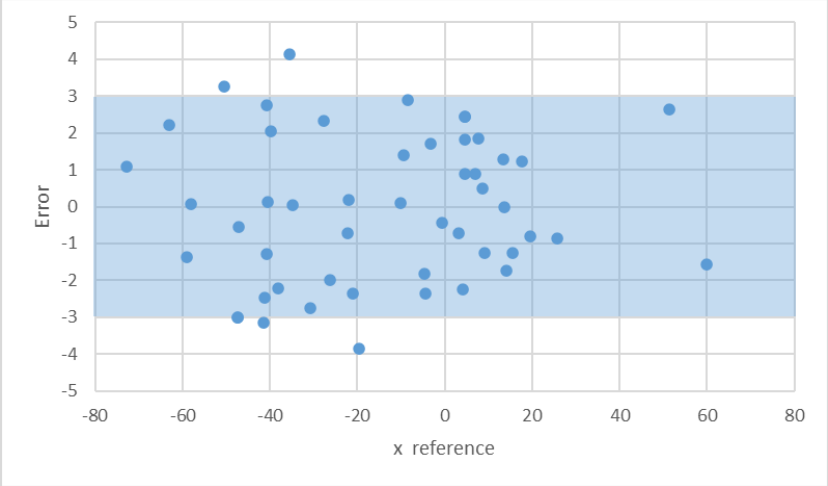


Figure 4.17. Error distribution of x at different levels.

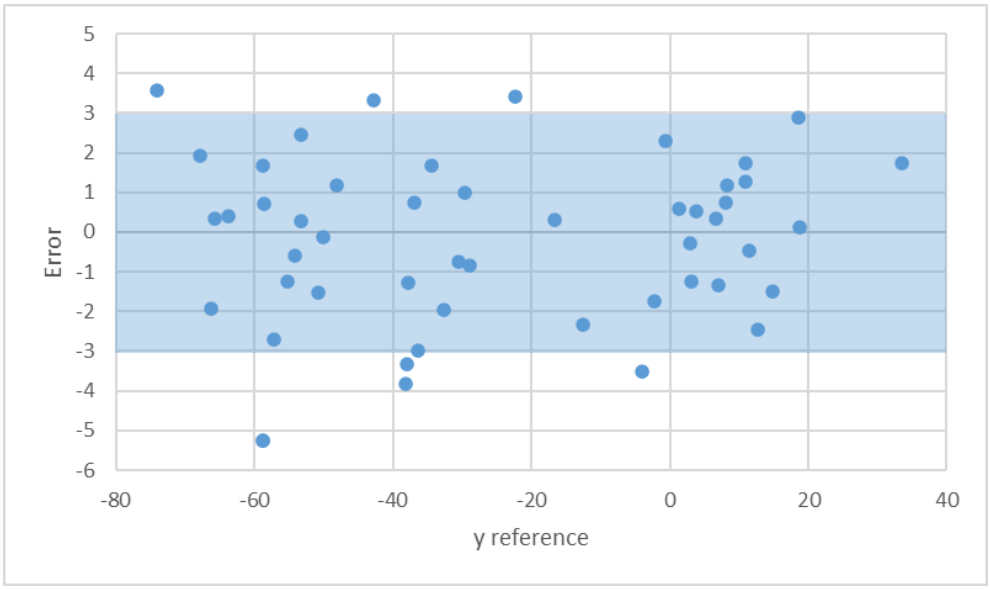


Figure 4.18. Error distribution of y at different levels.

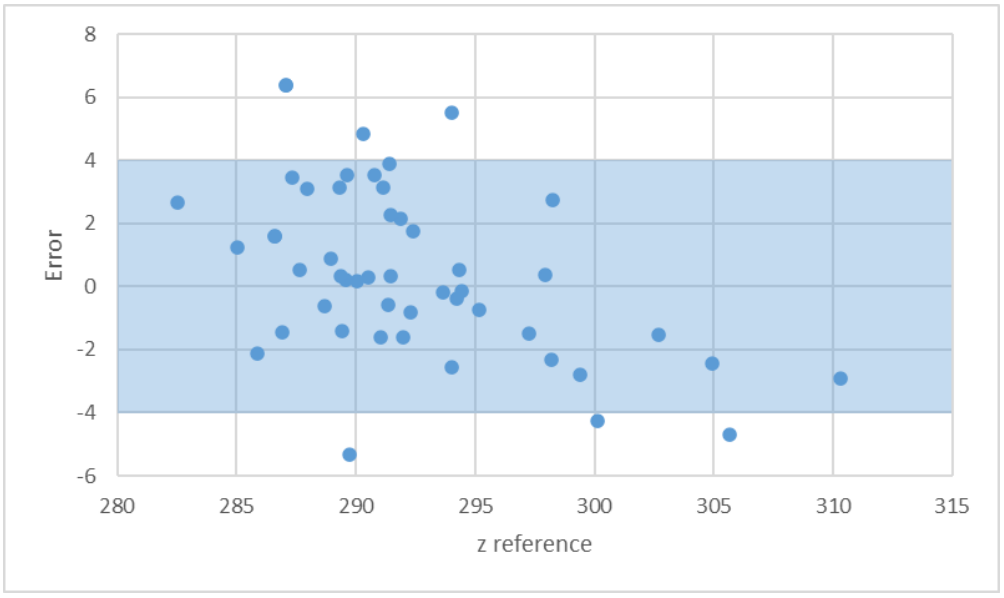


Figure 4.19. Error distribution of z at different levels.

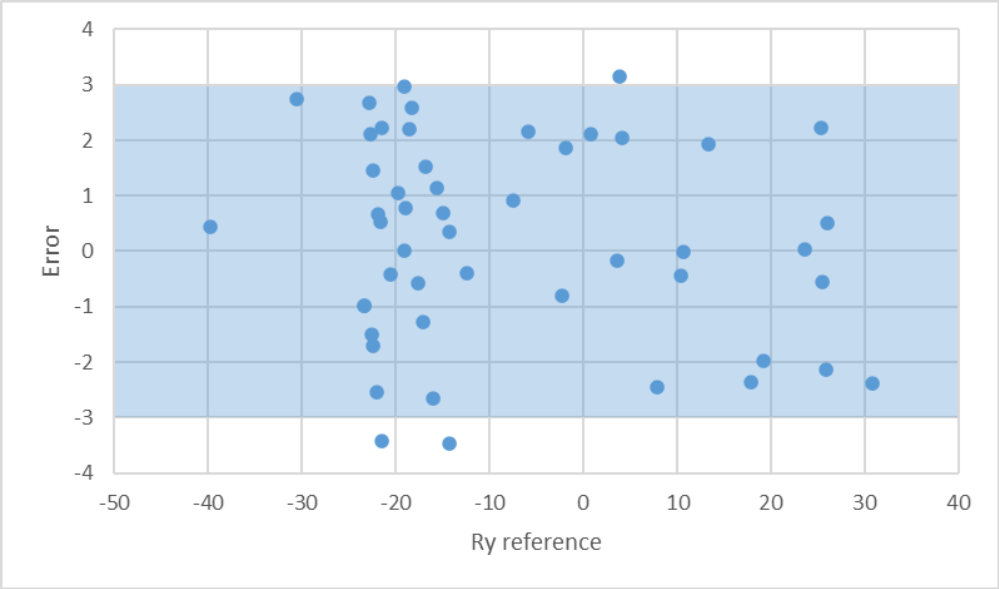


Figure 4.20. Error distribution of Ry at different levels.

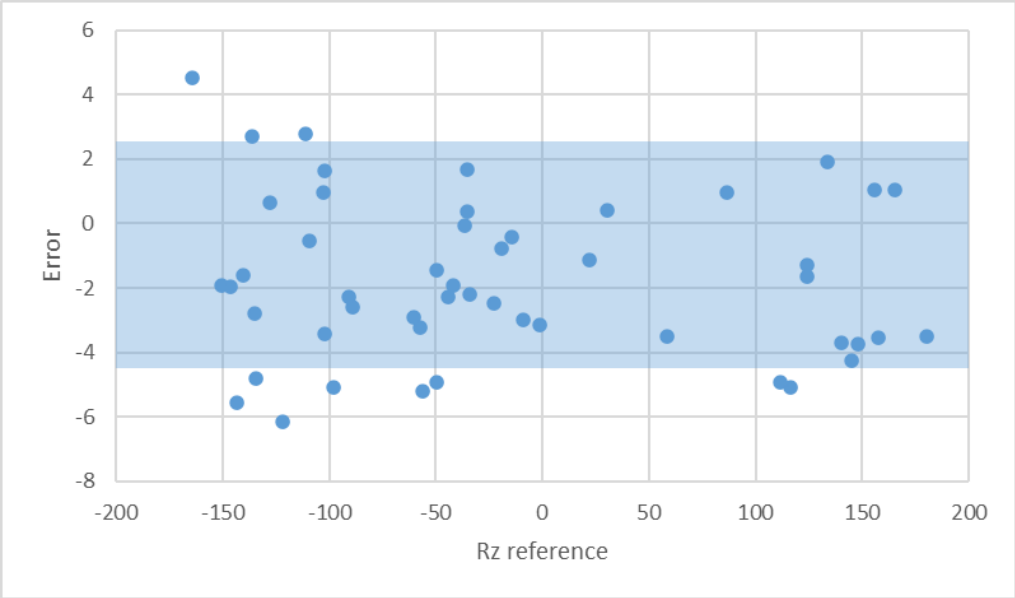


Figure 4.21. Error distribution of Rz at different levels.

The mean errors are -0.07mm, -0.24mm, 0.49mm, 0.19°, and -1.76° for x, y, z, R_y, and R_z respectively. The accuracy based on 5cm-5deg metric is pretty high (90%). Overall, the estimation errors of 5 values are in acceptable ranges around [-3 mm, +3 mm] for translation and [-3°, +3°] for rotation. However, the estimation of Ry and Rz depends strongly on the quality of the depth frame retrieved from the camera and the mask of the YOLOv5 detection outcome.

❑ *Estimation of the object with respect to the robot base:*

The team controls to move the robot to 50 different positions and get the data at those positions. The points will increase gradually along x in the range of [188 mm, 286 mm], y in the range of [-161 mm, 55 mm], Ry in the range of [-25°, 25°] and Rz in the range of [-54°, 48°].

Table 4.5. Error evaluation results of object estimation w.r.t. the robot base.

	Errors (mm-degree)				
	x	**y**	**z**	**Ry**	**Rz**
1	-1.77	-3.22	-6.43	-0.97	-2.26
2	-0.03	-2.56	-5.56	-0.89	-4.39
3	-1.44	-2.77	-5.62	-0.24	0.09
4	0.85	-2.79	-5.25	0.83	-0.13
5	1.24	-3.09	-5.54	-0.7	-1.86
…	…	…	…	…	…
46	-2.18	0.32	4.25	1.14	3.15
47	-2.46	-0.1	4.01	-0.68	1.36
48	-2.54	-0.02	3.75	-0.75	0.75
49	-2.59	-1.02	3.77	0.69	1.53
50	-3.14	0.62	4.67	1.02	1.56
Mean	**-2.3**	**-1.99**	**1.18**	**-0.25**	**0.34**

Table 4.6. Accuracy evaluation result.

Object	**No. Execution**	**Accuracy (%)**	**Average execution time (s)**
Bolt M10x25 (mm)	50	100	0.35

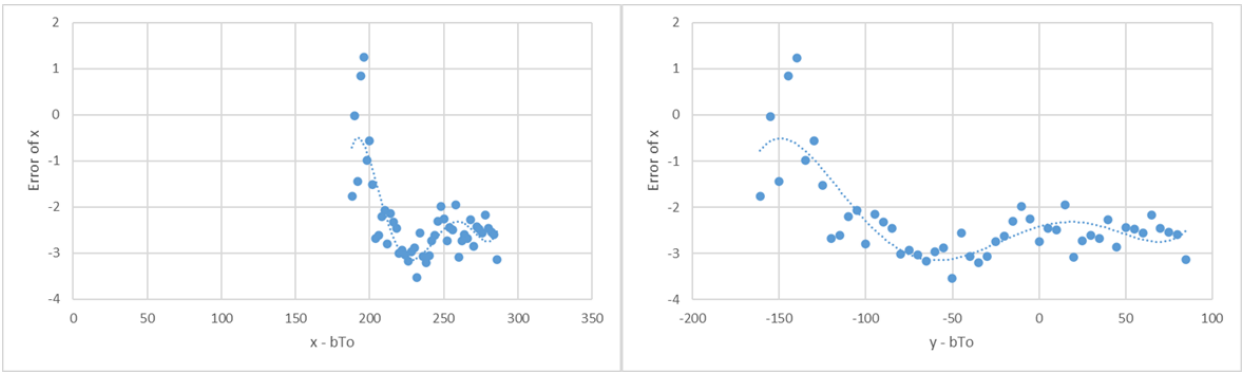


Figure 4.22. Distribution of x error with w.r.t the changes in x (left) and y (right) w.r.t robot base.

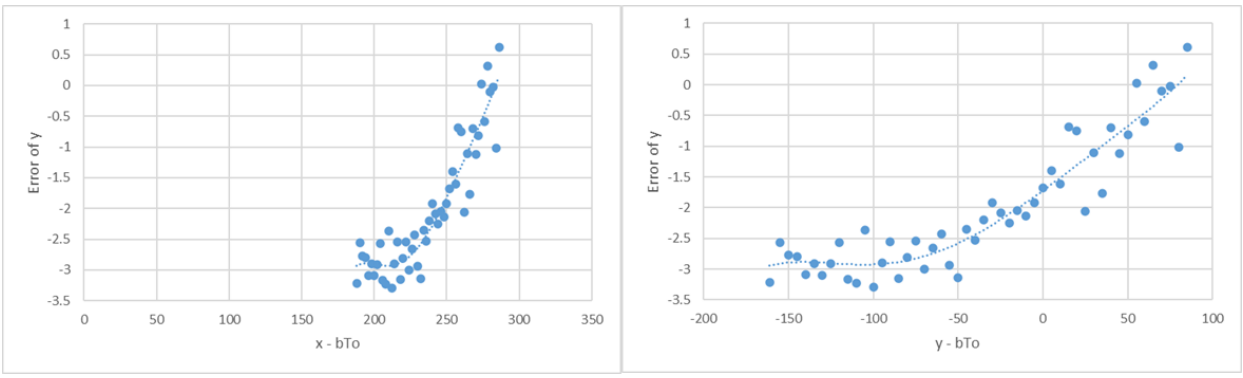


Figure 4.23. Distribution of y error with w.r.t the changes in x (left) and y (right) w.r.t robot base.

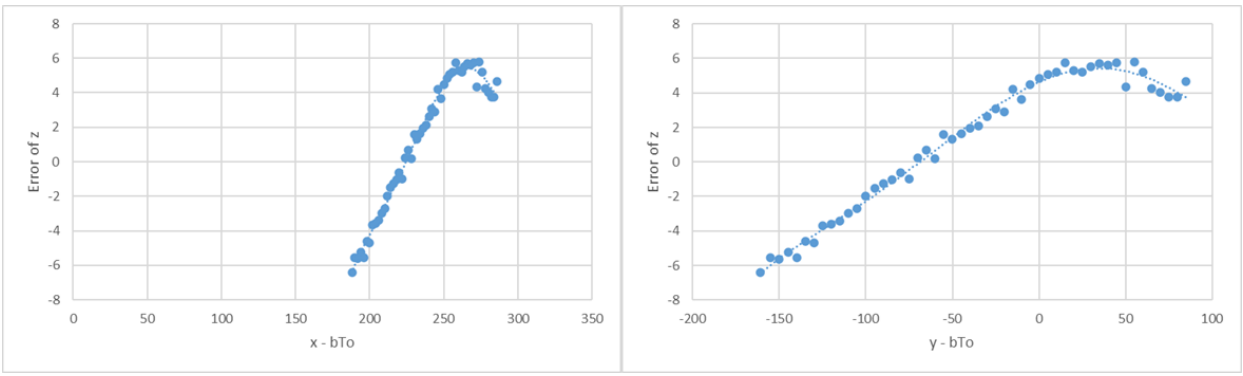


Figure 4.24. Distribution of z error with w.r.t the changes in x (left) and y (right) w.r.t robot base.

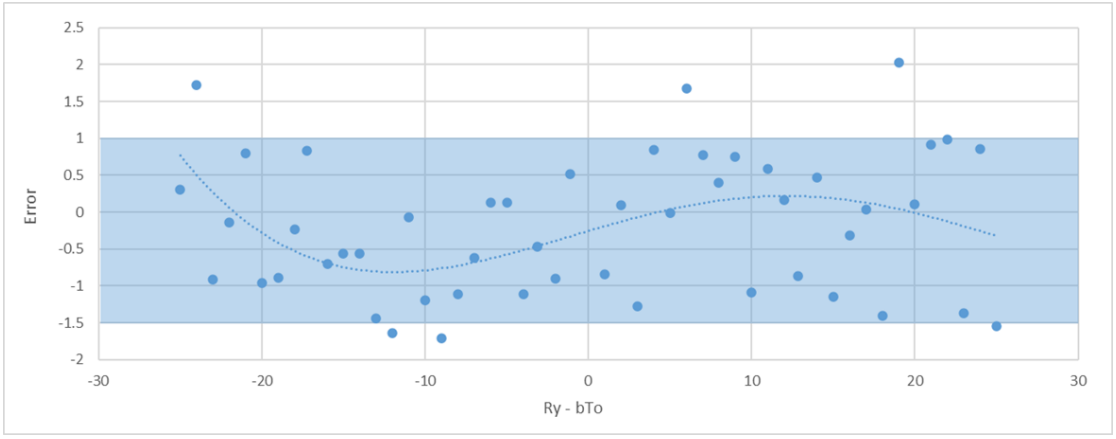


Figure 4.25. Distribution of Ry error with w.r.t the changes in Ry w.r.t robot base.

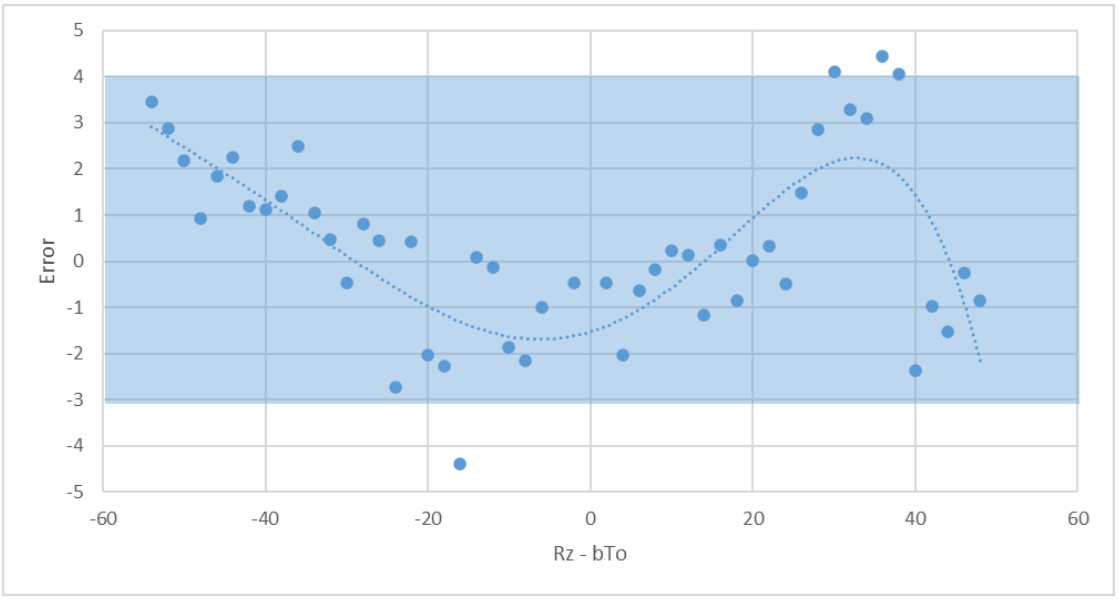


Figure 4.26. Distribution of Rz error with w.r.t the changes in Rz w.r.t robot base.

The mean errors are -2.3mm, -1.99mm, 1.18mm, -0.25°, and 0.34° for x, y, z, Ry, and Rz, respectively. The accuracy based on 5cm-5deg metric is high (100%). Overall, the combination of the estimation errors of object with respect to cam and the hand-eye calibration error will lead to the slightly higher error of the final result for grasping point, especially for z value, when the range of estimation error is around [-6mm, 6mm]. However, it can be seen from the figures above, the 5^{th} order polynomial regression trends of the 5 estimation errors that are sketched based on the distribution of the scatter points of error data are nearly fitting. They can be used to modify the final result for a better performance.

4.3.3. Evaluation of the object in other sizes

The team also performs evaluation of 100 samples for other sizes of the object to know if the algorithm can be applied to other size of the bolt. In 100 samples, 50 samples for the bolt M10x40 (mm) and 50 others for the bolt M12x40 (mm). These evaluation are made for the transformation of the object with respect to the camera. These evaluation results consist of both errors and accuracy as the same for the bolt M10x25 (mm).

Table 4.7. Error evaluation results of the bolt M10×40 (mm) estimation w.r.t. the robot base.

	Errors (mm-degree) of bolt M10x40 (mm)				
	x	**y**	**z**	**Ry**	**Rz**
1	-1.31	0.48	2.99	0.61	-1.8
2	-1.38	1.29	3.32	2.41	-0.58
3	-1.62	0.42	3.65	-0.31	-0.42
4	-2.04	0.52	2.87	-1.6	0.77
5	-1.43	0.27	2.96	3.34	0.42
…	…	…	…	…	…
46	-1.36	0.83	2.33	-0.05	-0.66
47	-1.01	0.73	3.37	-0.07	-0.63
48	-1.76	0.62	2.83	-2.8	-0.81
49	-1.11	0.18	2.91	0.59	-0.57
50	-1.56	0.26	2.84	-2.1	0.96
Mean	**-2.04**	**0.57**	**1.73**	**0.64**	**0.76**

Table 4.8. Accuracy evaluation result of the bolt M10×40 (mm).

Object	No. Execution	Accuracy (%)	Average execution time (s)
Bolt M10x40 (mm)	50	100	0.36

Table 4.9. Error evaluation results of the bolt M12×40 (mm) estimation w.r.t. the robot base.

	Errors (mm-degree)				
	x	y	z	Ry	Rz
1	2.26	2.24	-1.18	-2.87	-0.82
2	2.43	2.39	-2.64	-0.53	-0.45
3	1.37	1.67	-0.38	-1.94	-0.35
4	2.76	2.41	-0.38	-0.98	-0.09
5	1.51	2.78	-3.29	-0.05	0.67
…	…	…	…	…	…
46	0.69	0.97	-0.75	1.13	0.46
47	-0.69	0.48	-1.24	-2.74	-0.38
48	1.14	0.98	-0.78	-2.3	1.12
49	-0.24	0.89	-1.49	-0.02	-0.98
50	1.33	1.93	-0.51	1.14	0.21
Mean	**0.85**	**1.58**	**-2.45**	**0.65**	**0.05**

Table 4.10. Accuracy evaluation result of the bolt M12×40 (mm).

Object	No. Execution	Accuracy (%)	Average execution time (s)
Bolt M12x40 (mm)	50	100	0.36

From these tables shown above of 2 size of the bolt M10x40 (mm) and the bolt M12x40 (mm), the mean error and accuracy of both sizzes are still obtained at high scores with not much difference from those of the original research object – the bolt M10x25 (mm). However, the dataset for training set of the YOLOv5 is built for the bolt M10x25 (mm), so it will somehow affect directly the detection result of YOLOv5 for the object in other sizes.

4.4. Bin picking results

4.4.1. Criteria of a picked object

To evaluate the error of final result of grasping object, the team focuses on 2 criteria: orientation of the object and effectiveness of grasping the object. The objects are placed in the workplace with those criteria:

- ❑ Within the scope of the robot workspace.
- ❑ Within the camera's field of view.
- ❑ Not interference with the other objects when the robot grips.

4.4.2. Evaluation results

Table 4.11 Evaluation table of grasp possibilities.

Object	No. Executions	Success	Failure	Possibilities (%)
Bolt M10x25	100	94	6	94

The estimation time of YOLOv5 is approximately around 250-350 ms and for pose estimation is around 50 ms. Therefore, the total time of detection and pose estimation is around 300-400ms, which is acceptable calculation time for the current laptop configuration. This number can be reduced for a higher configuration.

Figure 4.27. Demonstration of the robot at approaching and grasping points.

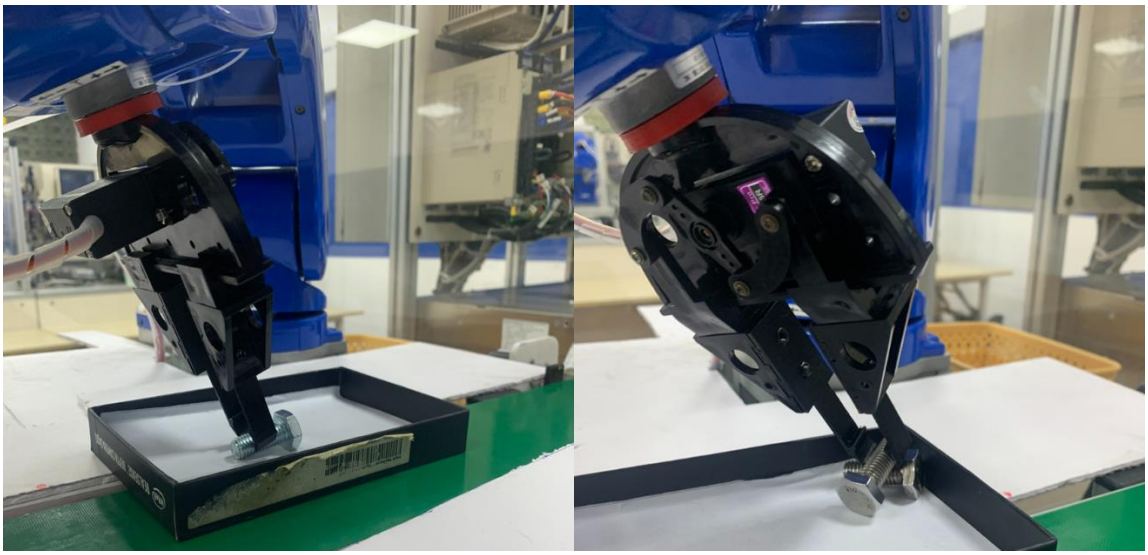


Figure 4.28. Robot reaching objects at the correct R_y angles.

Figure 4.29. Robot reaching objects close to the container walls.

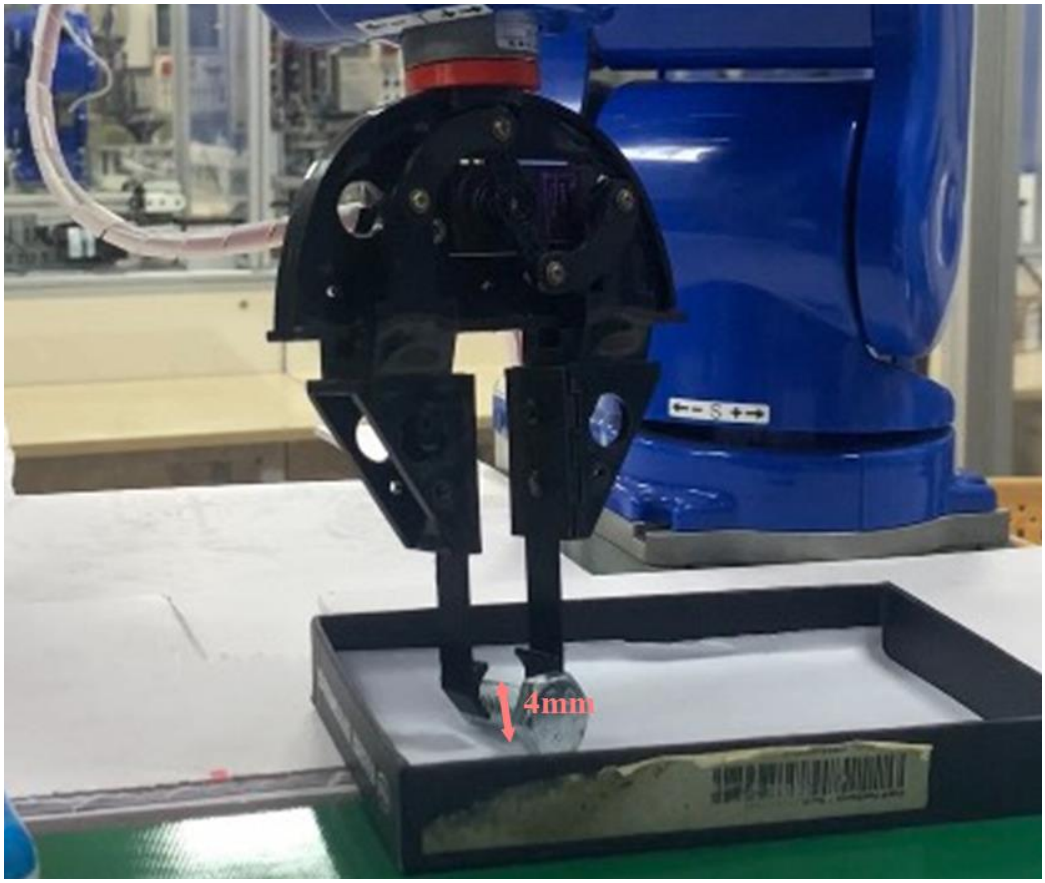


Figure 4.30. Robot not fully reaching the object.

The possibility of grasping the object is relatively high (94%), which has achieved the set goal of the thesis.

From Figure 4.16, it can be seen that the gripper reaches to the grasping point at the correct angles R_y. For Figure 4.29, the team has successfully processed the cases in which the objects are close to the wall of the container. In addition, the mechanical property of the gripper part designed in the Autodesk Inventor has made it easier to grasp the objects and hold them properly.

However, there are still some cases (Figure 4.30) that the estimation of z value of the object incorrectly which leads to failure in grasping the object.

4.5. Graphic user interface

The GUI consists of 3 parts: Robot Control Panel, Pick and place and, Model evaluation.

4.5.1. Robot Control Panel

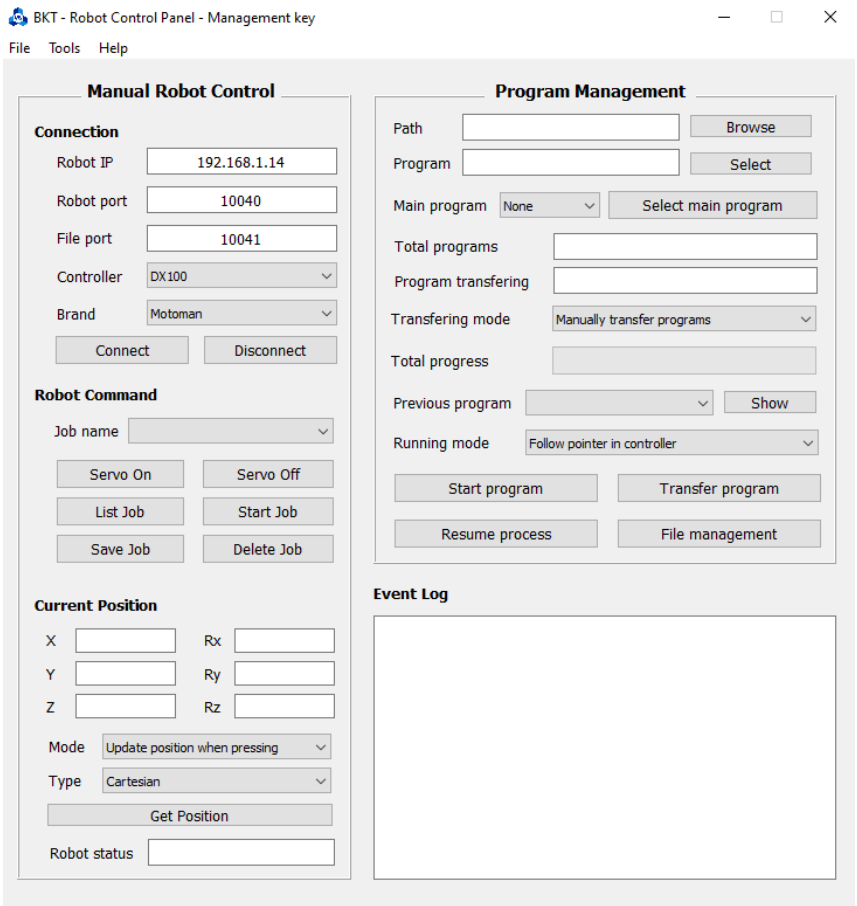


Figure 4.31. Control panel interface.

The Robot Control Panel interface contains almost of basic manual functions for robot control on the teach pendant, which includes:

- ❑ Turn on/off robot servo.
- ❑ Get robot data (running status, position, teach pendant mode).
- ❑ Data file control (load job, delete job, save job, select job, start job, list job).

4.5.2. Pick and place

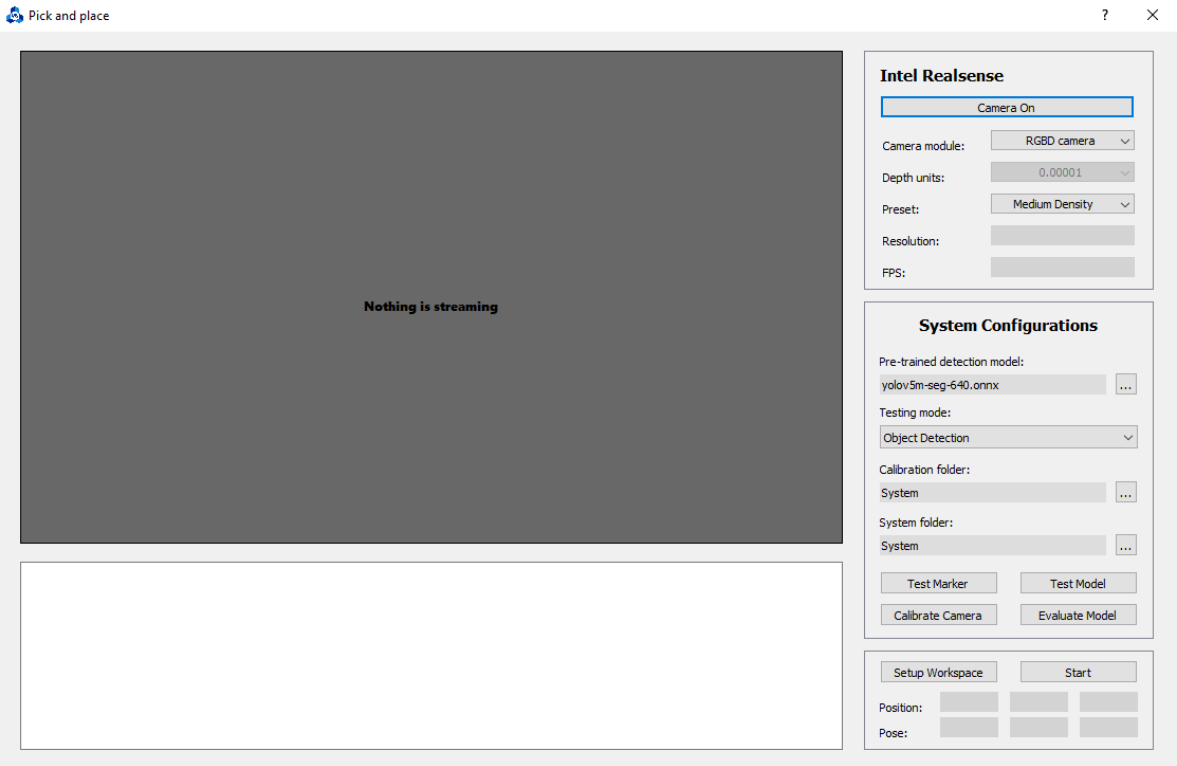


Figure 4.32. Pick-and-place control interface.

This interface consists most of the main functions of this thesis, which will allow the users to supervise and configure the camera settings, configuration data folder, acquire data from the camera, detection and pose estimation result of the objects (with the segmentation mask covering). Moreover, the interface has two important buttons: “Setup Workspace” and “Start”, which to setup the workspace of the container and trigger the system to run or stop, respectively.

4.5.3. Model evaluation

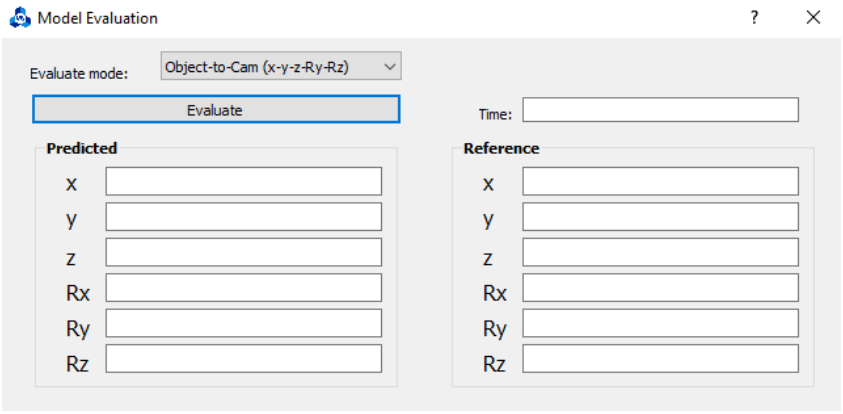


Figure 4.33. Evaluation dialog.

This interface helps the team to evaluate the whole system including the position and orientation estimation evaluation of the object to both camera and robot base. It will also display all the calculated result and the reference data retrieved from the controller.

Chapter 5. CONCLUSION

5.1. Results

In the course of conducting research on this topic, the team successfully addressed several issues and obtained the following outcomes:

The team employed a combination of deep learning techniques for object recognition and classification, alongside the PCA algorithm, and utilized Euclidean geometry to ascertain the position and orientation of objects. By comprehending the fundamental principles and characteristics of the camera, the team was able to exploit its features to enhance results in accordance with the research topic.

The team also gained an understanding of the YOLOv5 network's structure and made modifications to certain parameters to achieve optimal training results.

In terms of robotics control, the team implemented the UDP protocol to develop a library for connecting with the robot based on the manufacturer's guidelines. This enabled control over most functions via a laptop computer.

The gripper was designed with consideration for both the size of the object and the robot's payload capacity. It was controlled by a microcontroller that was programmed separately and synchronized signals with the computer program to facilitate the handling of the objects.

Ultimately, the team successfully developed a program to facilitate communication between users and computers for the purpose of controlling and monitoring the robot's execution process.

5.2. Future development

Regarding image processing, the team has aimed for solutions to improve the results as follows:

- ❑ Re-train the model to recognize objects being in orthogonal position relative to the ground, and adjust the grasping method accordingly.
- ❑ Make use a 3D camera with better specifications to obtain the point cloud of small-sized objects more effectively to further improve the pose estimation algorithm. The team suggests using the Intel RealSense D405 camera,

which is more suitable for capture depth image more accurately at medium distance.

- ❑ Research and utilize a more advanced machine learning model for object pose estimation rather than just object detection, so that the model becomes more flexible and the orientation of objects, in this case, bolts, of different lengths can be fully calculated and grasped.

Furthermore, for the pick and place methods, the gripper could benefit from a new redesign to further increase grasping probability.

Regarding user interface:

- ❑ Optimize code for further speed improvements.
- ❑ Optimize the program for stability and to operate on a variety of systems with lower specifications (e.g., embedded computer, HMI, etc.).

REFERENCES

[1] e. a. C. Martinez, "Setup of the Yaskawa SDA10F Robot for Industrial Applications, Using ROS-Industrial," *Advances in Automation and Robotics Research in Latin,* pp. 186-203, 2017.

[2] Interactive Design, "Automated test tube pick," [Online]. Available: https://interactivedesign.com/automatedtest-tube-pick/.

[3] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *IEEE Computer Conference on Computer Vision and Pattern Recognition*, 2016.

[4] Redmon, Joseph, and Ali Farhadi., "YOLO9000: better, faster and stronger," pp. 7263-7271, 2017.

[5] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[6] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao, *YOLOv4: Optimal Speed and Accuracy of Object Detection*, 2020.

[7] J. Glenn, *YOLOv5 by Ultralytics,* 2020.

[8] Muhammad Usman Khalid, Janik M. Hager, Werner Kraus, Marco F. Huber, Marc Toussaint, *Deep Workpiece Region Segmentation for Bin Picking,* 2019.

[9] T. P. Triển, H. Đ. Dũng, "ƯỚC LƯỢNG VỊ TRÍ VÀ HƯỚNG VẬT THỂ CHO ROBOT GẮP," Bộ môn Điều khiển Tự động, Khoa Điện - Điện tử, Trường Đại học Bách khoa - ĐHQG TPHCM, Thành phố Hồ Chí Minh, 2021.

[10] S. Liu, L. Qi, H. Qin, J. Shi and J. Jia, Path Aggregation Network for Instance Segmentation, 2018.

[11] J. Shotton, B. Glocker, C. Zach and S. Izadi, A. Criminisi and A. Fitzgibbon, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images.," in *2013 IEEE Conference on Computer Vision and Pattern Recognition.*, Portland, OR, USA, 2013.

[12] Yaskawa Electric Corporation, Motomini Instruction, 2018.

[13] Yaskawa Electric Corporation, YRC1000micro OPTIONS INSTRUCTIONS FOR ETHERNET FUNCTION, 2018.

[14] "Introduction to Principal Component Analysis (PCA)," [Online]. Available: https://docs.opencv.org/4.x/d1/dee/tutorial_introduction_to_pca.html.

[15] N. Mai, "Tổng hợp kiến thức từ YOLOv1 đến YOLOv5 (Phần 3)," [Online]. Available: https://viblo.asia/p/tong-hop-kien-thuc-tu-yolov1-den-yolov5-phan-3-63vKjgJ6Z2R.

[16] I. R. Technology, Intel RealSense D400 Series Product Family, 2020.

[17] C. Jee, "Robot-assisted high-precision surgery has passed its first test in humans," 11 February 2020. [Online]. Available: https://www.technologyreview.com/2020/02/11/844866/robot-assisted-high-precision-surgery-has-passed-its-first-test-in-humans/.

[18] Minghao Gou, Haolin Pan, Hao-Shu Fang, Ziyuan Liu, Cewu Lu, Ping Tan, *Unseen Object 6D Pose Estimation: A Benchmark and Baselines*, 2022.