

# Final Year Project (2021 – 2022)

## ELEC Final Report

# Smart Tank: Visual SLAM Based Competitive Robot with Object Detection for STEM Education

**Project ID:** SSJ03a-21  
**Supervisor:** Prof. Shaojie SHEN  
**Author (Student ID):** TAM Ho Kuen (), DENG Xingfa ()  
**Date:** 15 September 2021

### Main Objective

This project proposes a project that combines the latest STEM knowledge from diverse disciplines, including computer science, mathematics, and electronic engineering. The project develops an intelligent tank by applying computer vision, SLAM, embedded systems, basic electronics, and 3D printing. These exclusive illusion materials can support the teacher to master the methodology quickly and teach the youth the principle by offering hands-on experience on our proposed tank. It aims to provide a prior experience to the youth of STEM products that may ignite their passion for STEM subjects, even devoting themselves to future STEM careers.

### Objective Statements

1. To design and prototype a tank turret that can shoot the target with different types of Nerf darts by designing a shooting module and developing an ammunition selecting algorithm.
2. To develop the embedded system of the Smart Tank that can be manually controlled to shoot, move, and rotate the turret horizontally and vertically.
3. To program the microcomputer to enable object detection in the Smart Tank.
4. To implement robot navigation in a new environment with SLAM algorithms.

## **Contents**

<b><u>ABSTRACT</u></b>	v
<b><u>Section 1 INTRODUCTION</u></b>	P1
1.1 Background and Engineering Problem	P1
1.2 Objectives	P1
1.3 Literature Review of Existing Solutions	P2
<b><u>Section 2 METHODOLOGY</u></b>	P3
2.1 Overview of Smart Tank	P3
2.1.1 Product Description	P3
2.1.2 System Block Diagram for Smart Tank	P3
2.1.3 Components List	P4
2.1.4 ECE Knowledge	P4
2.2 Objective Statement Execution - Plan for Producing Smart Tank	P5
2.2.1 Prototyping a Well-functioning Tank Turret	P5
2.2.2 Prototyping the Smart Tank	P9
2.2.3 Targets Identification and Tracking	P12
2.2.4 Robot Navigation in a New Environment	P19
2.3 Smart Tank Evaluation & Discussion	P22
<b><u>Section 3 CONCLUSION</u></b>	P23
<b>REFERENCES</b>	P24
<b>APPENDICES</b>	P25
Appendix A – Final Project Plan	P25
Appendix B – Budget	P28
Appendix C – Meeting Minutes	P29
Appendix D – Group Members’ Contributions	P32
Appendix E – Deviation(s) from the proposal and supporting reason(s)	P34
Appendix F – Monthly Reports	P35
Appendix G –Different designs of the shoot and loading module for 2.2.1 Task 1	P40
Appendix H – Different designs of the turret for 2.2.1 Task 2	P41
Appendix I – Source code for 2.2.2 Task 2	P42
Appendix J – Source code for 2.2.3 Task 3 and 4	P48
Appendix K – Source code for 2.2.4 Task 3	P54

## **List of Illustrations**

### **List of Figures**

Figure 1 – Schematic of the whole Smart Tank system	P3
Figure 2 – Detailed dimensions of a standard Nerf dart	P5
Figure 3a – Standard Nerf darts	P5
Figure 3b – Nerf AccuStrike darts	P5
Figure 3c – Nerf suction darts	P5
Figure 4a – Folded single-layer carousel design	P6
Figure 4b – Separated two-layer carousel design	P6
Figure 4c – Revolving cylinder design	P6
Figure 5a – An overview of the preliminary prototype	P7
Figure 5b – The front of the prototype	P7
Figure 5c – The middle of the prototype	P7
Figure 5d – The lower part of the prototype	P7
Figure 5e – The joystick for controlling the horizontal and vertical rotation	P7
Figure 5f – The LCD touch screen with user interface	P7
Figure 6 – The schematic of the preliminary prototype system	P7
Figure 7 – 3D graph of X-Loader v5	P8
Figure 8a – 3D graph of X-Turret Gen3	P9
Figure 8b – X-Turret Gen3 with X-Loader v5	P9
Figure 9a – Mecanum wheeled chassis	P11
Figure 9b – Tracked chassis	P11
Figure 10 – Overview of the Smart Tank	P12
Figure 11a – Yolo.py before modification	P14
Figure 11b – Yolo.py after modification	P15
Figure 12 – Results of implementing YOLO V3	P16
Figure 13a – Dataset of training targets	P17
Figure 13b – Labeled results of dataset	P17
Figure 14a – Results of implementing YOLO V3 to detect Drone	P18
Figure 14b – Results of implementing YOLO V3 to detect Tank	P18
Figure 15a – Micro-computer of MANIFOLD 2-G	P19
Figure 15b – Other components of wireless monitor, Intel Realsense d435i and prototype of the Tank	P20
Figure 16 – The result of Hector mapping	P21
Figure 17 – The result of running VINS-Mono in a designed route	P21

**List of Tables**

Table 1. List of Components and Specifications	P4
Table 2. Smart Tank Schedule	P27
Table 3. Budget	P28

**ABSTRACT**

To motivate more youth to pursue a STEM career, this project aimed to design and build a robotic tank for STEM education purposes that allows secondary and college students to gain hands-on experience in robotic design. We present the Smart Tank, a visual SLAM based competitive robot with object detection for STEM Education. In the development of the Smart Tank, we applied the knowledge of embedded system design and 3D printing technology to prototype the ammo selection system and body of the Tank. In order to combine the latest technology of computer science applications, we developed the function of object detection and simultaneous localization and mapping (SLAM) on the Smart Tank. Therefore, the project could be customized as STEM education materials for youth.

## SECTION 1—INTRODUCTION

### 1.1 Background and Engineering Problem

STEM is referring to a curriculum that aims at nurturing students in four specific disciplines – science, technology, engineering, and mathematics. Engaging students in STEM disciplines helps build up the “four C’s” of the 21-century essential skills: critical thinking, creativity, collaboration, and communication [1]. The full STEM talents are expected to bring a new perspective on solving the existing or future problems of business or society.

However, the undesirable promotion of STEM education has been identified as an obstacle to the development of technology and innovation in Hong Kong. In Hong Kong, science subjects are optional electives for senior secondary students. They can opt to declare one, two or three science subjects as the electives. According to a survey in 2019/20, merely 49% of Hong Kong Diploma of Secondary Education Examination (HKDSE) candidates have studied at least one science subject [2]. Moreover merely 35% of UGC-funded undergraduates chose STEM-related programmes in 2019/20 [3].

As a result, the number of students with a STEM background is too small. The meaning behind the statistic is that there is no sufficient STEM education support in the Hong Kong education system. According to a survey by the Federation of Education Workers, 63.6% of teacher respondents have no confidence in teaching STEM subjects. As well as 81.6% and 70.5% of teacher respondents reflect there is not sufficient hardware equipment and support for teaching materials [4].

To fulfill the niche shortage of STEM education support, a system that combines the latest STEM knowledge from diverse disciplines, including computer science, mathematics, and electronic engineering. A smart tank that applies computer vision, robotics, and embedded systems which is for youth education purposes would support the teacher to master the methodology quickly and teach youth the principle by offering hands-on experience. Providing preliminary experience to the youth with STEM products may ignite their passion for STEM subjects, even, devoting themselves to future STEM careers.

### 1.2 Objectives

The project aims to design and build a robotic tank for STEM education purposes that allows secondary and college students to gain hands-on experience in robotic design by learning and applying theoretical knowledge from computer science, mathematics, and electronic engineering, and support teachers with little or no experience in teaching STEM.

#### 1.2.1 Objective Statements

Objective 1: “To design and prototype a tank turret that can shoot the target with different types of Nerf darts by designing a shooting module and developing an ammunition selecting algorithm.”,

Objective 2: “To develop the embedded system of the Smart Tank that can be manually controlled to shoot, move, and rotate the turret horizontally and vertically”,

Objective 3: “To program the microcomputer to enable object detection to the Smart Tank”,

Objective 4: “To implement robot navigation in a new environment with SLAM algorithms”.

### 1.3 Literature Review of Existing Solutions

There are three common methods that are promoting STEM education to the young generation: STEM education schools, STEM education toys and STEM competitions. We are writing to analyze the strengths and shortcomings of the mentioned mediums.

#### A. STEM Education Schools

STEM education schools are leading the trend of promoting STEM education. The schools are offering plenty of hands-on courses related to IoT, programming, robotics, etc. And the target customers are ageing in a wide range from 3 to 14 years old. More importantly, they are offering the service of on-site courses for professional STEM teachers to teach students in person [5]. However, STEM education schools also have some shortcomings. The majority of the education STEM schools are private organizations. There is no assurance of the education quality. As well as the costs of the curriculum are not transparent. Parents or schools have to pick the STEM education school in a market that lacks public information. The last and biggest shortcoming is that most courses from STEM education schools are self-financed. Then, financial affordability is a critical factor preventing some students from taking STEM courses.

#### B. STEM Education Product

STEM education toys are designed to let kids have fun and learn the knowledge at the same time by gaining hands-on experience in building the toys. Students can have a preliminary understanding of the latest technology like face recognition toys [6], and robotic cars [7] by playing with the toys. Since toys can always draw the attention of kids, the STEM interest of kids might be ignited. Then, more children might opt to study the science course in the future. However, STEM education toys are not a comprehensive product. The toys are always related to one or two fields of technology. For instance, some object detected camera is related to computer vision and robot master is majorly related to robotics. Moreover, some STEM education products are for toy use purposes. There is not enough instruction support to help teachers demonstrate the theory behind the product. Therefore, there is no comprehensive product as teaching material in the school.

#### C. STEM Competition

STEM competition is collecting all the STEM talents or enthusiasts to unleash their creativity. In Hong Kong, several organizations are arranging STEM competitions, including the Hong Kong Federation of Youth Groups [8], Hong Kong New Emerging Technology Education Association, etc. Most of the competition allows students to join the competition in groups. They can utilize their knowledge, creativity, innovation, and collaboration to solve problems in the real world. The competition can offer a terrific opportunity for youth to experience the potential STEM career works that might be the seed to lead them to pursue a STEM career in the future. Although there are a lot of competitions in Hong Kong, only a small group of students from school can participate in the event. Thus, the impact is still limited.

All three methods can promote STEM education to the students. However, every method has some limitations. There is no method that can promote STEM education to every student and deliver the latest comprehensive technologies to them.

Our project proposes a comprehensive STEM education project that can be used STEM education materials by Primary or Secondary teachers. The name of the project is "Smart Tank." Smart Tank is a STEM education product that combines all the latest technologies, including computer vision, SLAM, embedded systems, basic electronics, and 3D printing. Although the project is integrated with advanced knowledge from different fields, a solid academic background is not required for building the project. The Smart Tank can be customized in diverse designs. Schools can arrange the STEM design competition using Smart Tank as a

medium that allows every student to utilize their creativity and innovation. Then, every student can have a solid STEM experience in their junior years.

## SECTION 2—METHODOLOGY

### 2.1 Overview of Smart Tank

#### 2.1.1 Product Description

We create a robot tank that can localize and map the new environment simultaneously for moving to a designated location, as well as identify and track the targets and shoot it with different shells according to the objects it identifies. Smart Tank is an integration of hardware and software. By default, it is equipped with a sophisticated ammunition loading system, a stereo camera with an innate initial measurement unit (IMU) and the 4-Mecanum-wheel chassis. These components allow Smart Tank to sense the environment and detect the targets with the advanced sensors by applying different SLAM algorithms and computer vision algorithms for object detection. With the pioneering ammunition selection hardware design and algorithm, Smart Tank is ready to shoot different types of shells, which makes the Tank operation realistic. More potential and innovative rules for competitions could be developed by the users.

Unlike the traditional STEM education products that only perform single-purpose tasks, Smart Tank is a modular design and able to be disassembled. Users can replace the default parts including the turret, sensors, and wheels of the Tank according to their objectives and the missions of the STEM competition. Moreover, we encourage robot enthusiasts to develop and share their unique ideas with the Smart Tank community. Therefore, Smart Tank is reprogrammable and some of its components can be replaced.

#### 2.1.2 System Block Diagram for Smart Tank

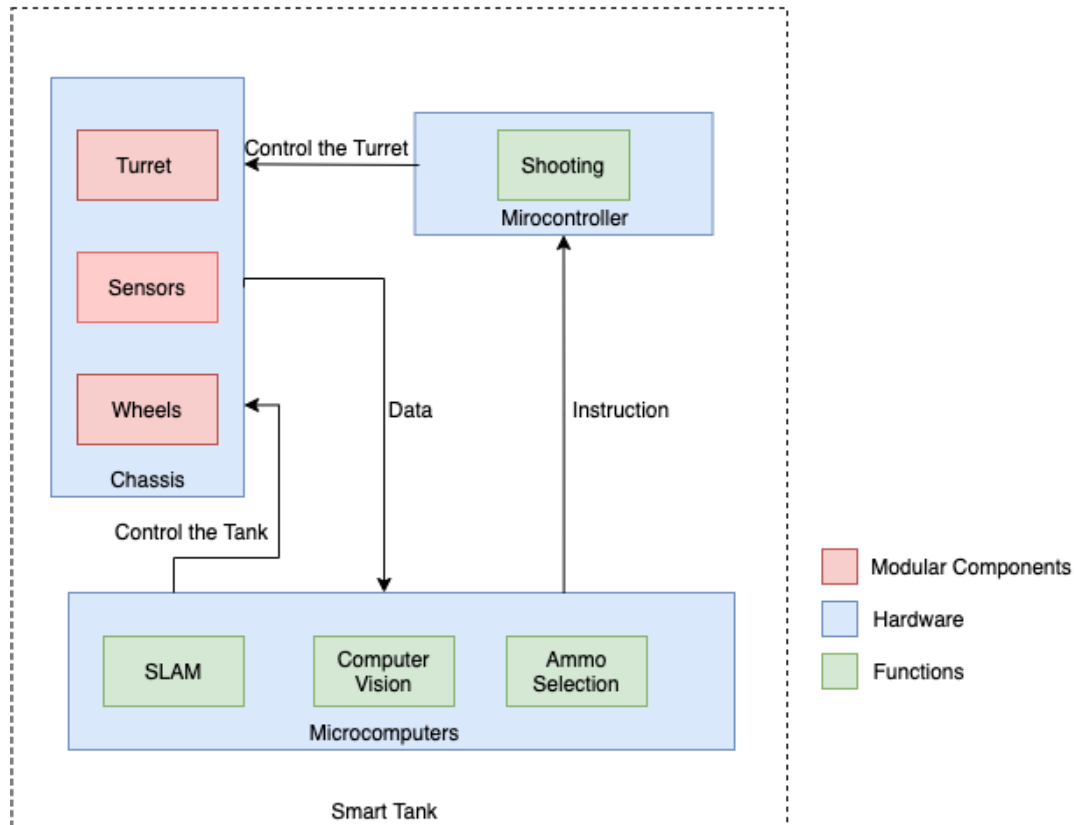


Figure 1 – Schematic of the whole Smart Tank system



### 2.1.3 Components List

Table 1. List of Components and Specifications

Items	Specifications/Model
Camera with IMU	Intel RealSense D435i
Servo motors	DG-995 20kg, DS3120
Microcontroller	DJI RoboMaster Development Board Type A, Arduino UNO
Microcomputer	DJI Manifold 2-G (128GB)
Turret of the tank	3D printed
Chassis of the tank	HKUST RI RoboMaster Chassis
DC motors	RS370-KT/4035
Li-ion batteries	DJI TB48D

### 2.1.4 ECE Knowledge

The Department of Electronic and Computer Engineering (ECE) carefully designed the undergraduate electronic engineering program and aims to provide all-rounded exposure to advanced and popular topics such as robotics, computer programming, digital communication, embedded system, and integrated circuit design in the 4-year curriculum. After seeking the various knowledge from the courses, we would like to utilize the technologies and problem-solving mindset which we learned and developed for this ECE FYP. We will explain the contributions of specific ECE courses to our FYP in the following.

To begin with ELEC 3210 Machine Learning and Information Processing for Robotics. This course introduces fundamental knowledge about the information processing techniques for robotics. It covers simultaneous localization and mapping (SLAM), Bayes theorem, hidden Markov model, Gaussian process, classification, support-vector machine (SVM) and kernel methods for regression. Moreover, we learnt the working principles of robotic sensors including Lidar, RGB-D camera as well as their specific model and real applications. We tried sensor fusion and ran different SLAM algorithms in robot operating systems (ROS). Therefore, this course contributes a lot to our FYP by enabling us to research and develop a robot for real application.

Besides, ELEC 3300 Introduction to Embedded Systems also provides some practical techniques for our ECE FYP. This course covers the integration of machine-level software and hardware in ARM-core microcontroller-based systems to help us interface and program microcontroller systems. Moreover, it provides several laboratory experiments to enhance our understanding of the techniques and technologies learnt via practices. Therefore, this course teaches us not only the technical skills about a project that integrates software and hardware, but also the practical problem-solving mindset evolved from the laboratory experiments.

Finally, ELEC 4240 Deep Learning in Computer Vision is responsible for one part of our FYP. This course covers convolutional neural networks (CNN), recurrent neural networks (RNN) and end-to-end optimization for training deep networks in computer vision. We learnt how to implement, train, and test deep neural networks on cutting-edge computer vision research.

## 2.2. Objective Statement Execution - Plan for Producing Smart Tank

### 2.2.1 Prototyping a well-functioning tank turret

This objective is to design and prototype a tank turret that can shoot and rotate. To achieve this, the turret should have a shooting and loading module, and a turret case containing this module. Since the turret is tailor-made, both the shooting and loading module and the turret case are supposed to be made by 3D printing.

**Task 1:** Designing and prototyping a shooting and loading module, and developing an ammunition selection algorithm

**Member in charge:** TAM Ho Kuen

**Task description:**

Researching the typical design of modern tanks' shooting modules. Then, we determine which design is most valuable for us to imitate and the future development. Hence, we design and prototype a shooting and loading module for the Tank. A shooting system that can shoot 3 meters away from targets with a 20cm dispersion, and a sophisticated loading system that allows the Tank to select a designated type of ammunition and then send it to the shooting position. Moreover, the ammunition capacity should be 6, and 3 types of ammunition should be equally stored in the Tank.

**Work Done:**

The Tank is supposed to shoot Nerf darts as shown in figure 2. We received comments from professional Nerf players, they suggested that a good Nerf player would use different Nerf darts situationally. For instance, they will use inexpensive standard Nerf darts as shown in figure 3a for most situations, Nerf AccuStrike darts as shown in figure 3b for long-range targets, and Nerf suction darts as shown in figure 3c for shooting at the smooth surfaces. Therefore, shooting different types of Nerf darts became one of the requirements of our project.

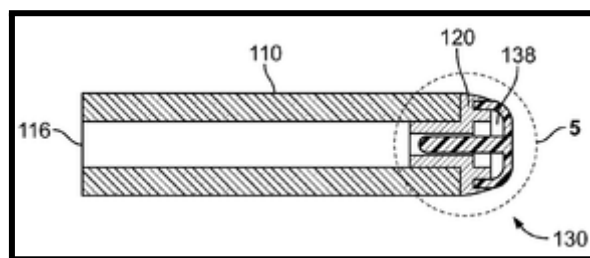


Figure 2 – Detailed dimensions of a standard Nerf dart

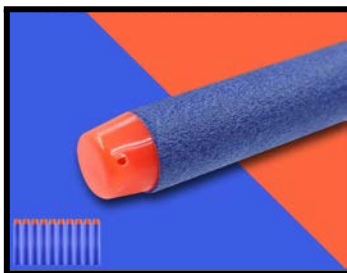


Figure 3a

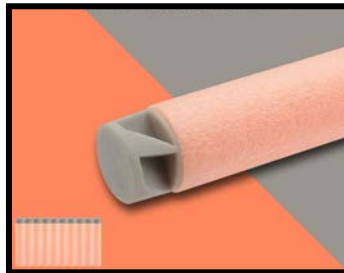


Figure 3b

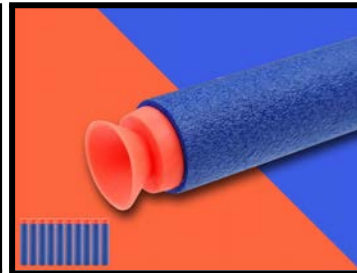


Figure 3c

Figure 3a – Standard Nerf darts

Figure 3b – Nerf AccuStrike darts

Figure 3c – Nerf suction darts

We have done a comprehensive study on the loading system of modern tanks, including Russian T80's folded single-layer carousel design as shown in figure 4a, Russian T90's Separated two-layer carousel design in figure 4b and French Leclerc's revolving cylinder design in figure 4c. Since Nerf darts have the dimensions of 12.7 and 72 mm with 12.7 mm diameter, the 4a and 4b designs will make the shooting module very large and occupy a lot of space in the turret. After balancing the complexity and budget we have, we designed to imitate France's Leclerc tank's autoloader which is a revolver-like design.



Figure 4a

Figure 4b

Figure 4c

Figure 4a – Folded single-layer carousel design

Figure 4b – Separated two-layer carousel design

Figure 4c – Revolving cylinder design

We have designed 5 versions of the shooting and loading system, while they have major differences. As the second version is a milestone of our project, we would introduce it as below.

After deciding to adopt the revolving cylinder design, we moved to the blueprint process and started making a preliminary prototype of the shooting module for the proof of concept. Since 3D printing is very expensive, we used cardboard and metal plates to make our first prototype in figure 5a.

The system flow of the shooting module's preliminary prototype is shown in figure 6. We used a 360-degree servo motor FS90R to rotate the designated slot to the shooting position as shown in figure 5c, then the dart was pushed to the shooting position by an MG90S servo motor as shown in figure 5c, followed by the acceleration by two DC motors with friction wheels as shown in figure 5b. Moreover, we used two MG90S servo motors for the horizontal and vertical rotation as shown in figure 5d, which were controlled by a joystick as shown in figure 5e. Moreover, we enabled the user to choose types of darts and aim the target with the LCD touch screen as shown in figure 5f.

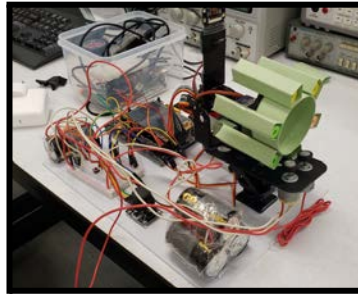


Figure 5a

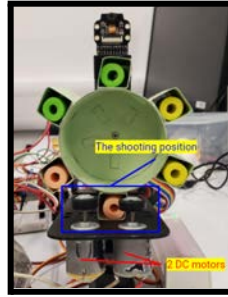


Figure 5b

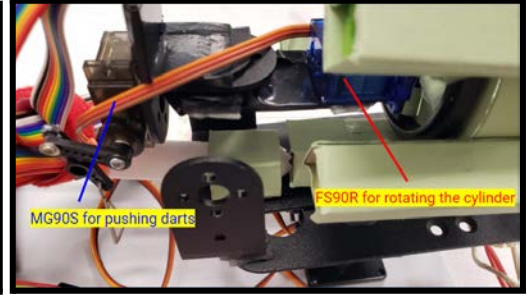


Figure 5c

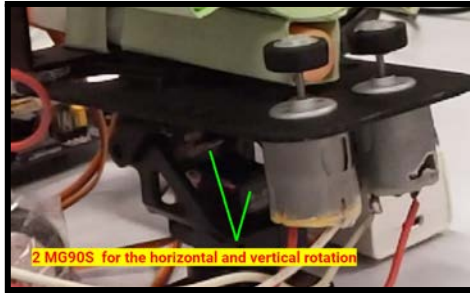


Figure 5d

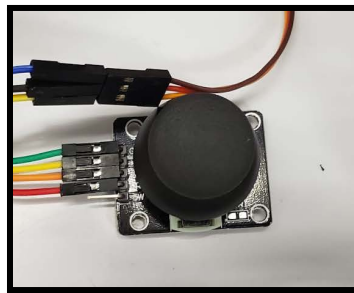


Figure 5e



Figure 5f

Figure 5a – An overview of the preliminary prototype

Figure 5b – The front of the prototype

Figure 5c – The middle of the prototype

Figure 5d – The lower part of the prototype

Figure 5e – The joystick for controlling the horizontal and vertical rotation

Figure 5f – The LCD touch screen with user interface

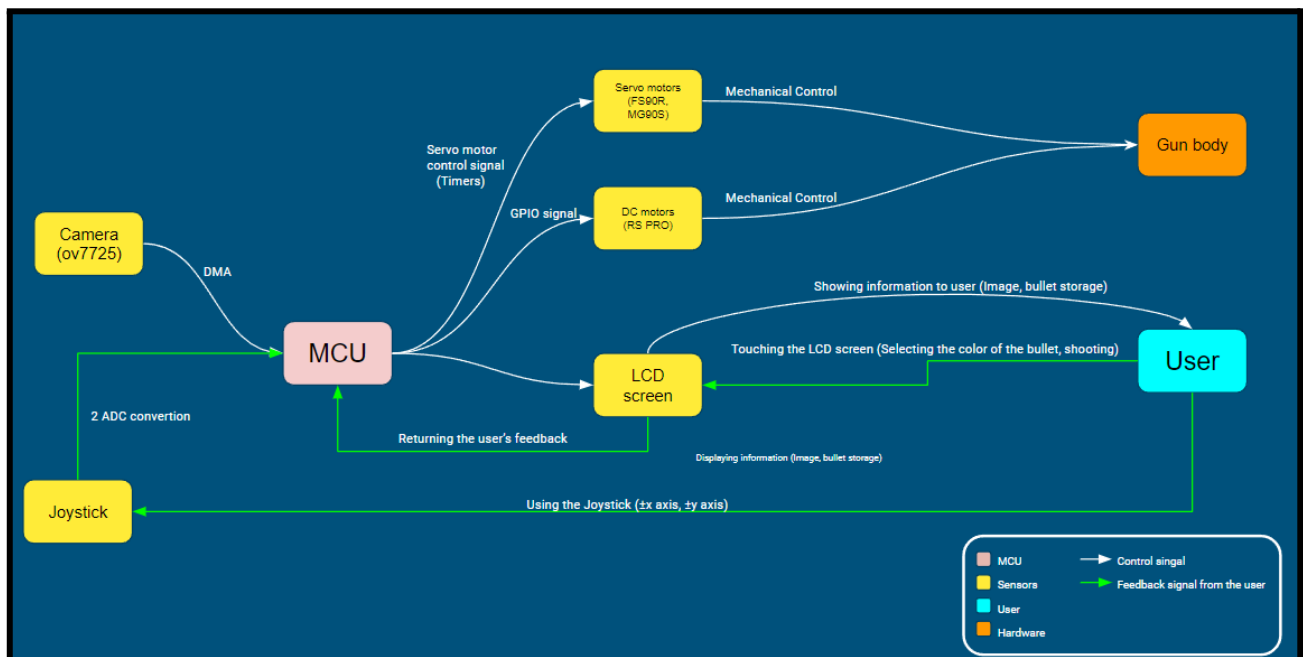


Figure 6 – The schematic of the preliminary prototype system

The preliminary prototype was successful, and able to perform most of the functions we proposed. Moreover, we recorded valuable feedback and data from testing it for further improvement. For example, the MG90S servo motors did not have enough power to lift the shooting and loading module. The FS90R did not meet the accuracy requirement, where the slot was frequently not collinear to the shooting position, resulting in a dart jammed in the chamber.

In the next phase, we planned to use 3D printing to make a more detailed and compact shooting and loading module. From the previous result, we firstly replaced the MS90S with DG-995 30kg which can offer higher torque force, as well as replaced the FS90R with DS3120 which can offer more accurate rotation. Besides, we upgraded DC motors to ultra-high-speed RS370-KT/4035 for enhancing the effective range. After that, we redesigned the whole module on SolidWorks. Moreover, we use a push-pull electromagnet to push the bullet to the shooting position.

The module has been corrected many times according to suggestions from PhD students and experimental results. The latest design, X-Loader v5, is shown as shown in figure 7a, while the X-Loader v5 with electronics is shown as shown in figure 8b. X-Loader v5 is completely modularized since it can be disassembled and replaced with a longer barrel or other upgrades.

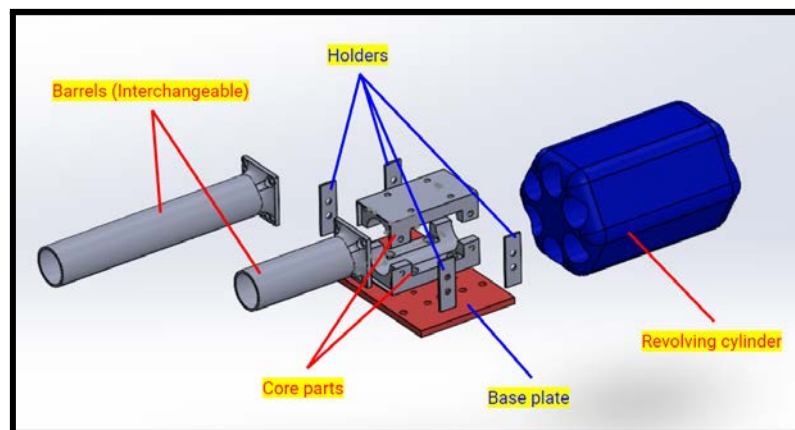


Figure 7 – 3D graph of X-Loader v5

Based on our testing, the finalized shooting and loading module has an excellent shooting accuracy with the dispersion less than 17 cm diameter when the Tank shoots at a target 3 meters away. The mechanical reliability is also satisfactory where the jamming rarely occurs.

Besides, we developed the ammunition selection based on the finite state algorithm. We assigned a state to Nerf darts in the revolving cylinder respectively. Since the ammunition storage contains different types of darts, the rotation order can switch between clockwise and anticlockwise for achieving the shortest path. Moreover, for some scenarios that the designated shells are all used, the loader switches to load another type of shell automatically. Based on our experiments, the ammunition selection algorithm is proved computationally inexpensive and robust.



**Task 2: Designing and prototyping a turret case****Member in charge:** TAM Ho Kuen**Task description:**

After accomplishing the implementation of the shooting and loading module, we have to design the turret, and install X-Loader v5 in it. The printing cost of the turret case should not exceed HKD 700.

**Work Done:**

The design process of a turret case is similar to designing the shooting and loading module, we firstly constructed the 3D model of the turret on SolidWorks. The difference is that we cannot print the turret case for testing after drafting a 3D model on SolidWorks, because the size of the turret exceeds the build volume of the 3D printer in the laboratory. We need to purchase printing services, and thereby the printing cost is a very important factor in designing the turret. As the printing price evaluates the complexity, the type of material and the consumption of materials for the product, we need to simplify the design and reduce the size as much as we can.

We now have three generations of the turret. The first design is too complex and costly, while the second one is too radical in cutting down the cost, resulting in poor compatibility with other components. The latest X-Turret Gen3 as shown in figure 8a strikes a balance between the cost and the system integrity. X-Turret Gen3 has consisted of different materials, ranging from resin and carbon fiber, according to the resultant force on a specific part when a bullet is shot. After double-checking with PhD students and correcting some minor mistakes, we sent the 3D graph to the 3D printing services provider to print it out.

Besides, we installed X-Loader v5 in X-Turret Gen3. Thanks to the precise measurement, the careful 3D design and the superior 3D printing quality, X-Turret Gen 3 is very compact, where no space is wasted and parts are closely stuck with each other. Therefore, the printing cost is successfully minimized to HKD 625, as well as no additional purchase is needed.

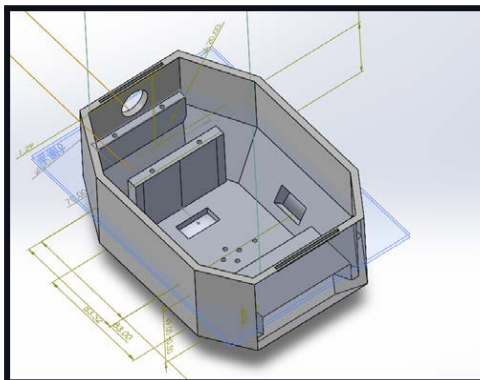


Figure 8a

Figure 8a – 3D graph of X-Turret Gen3

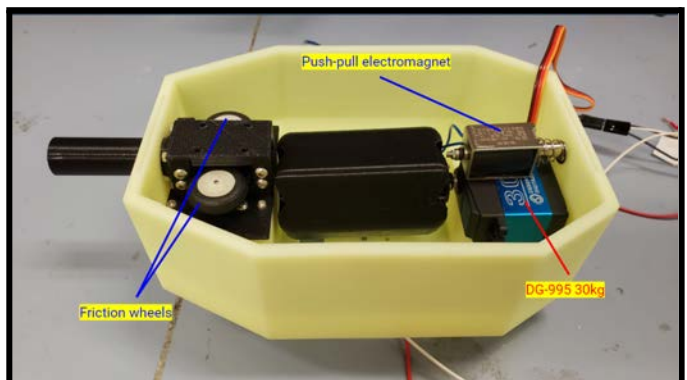


Figure 8b

Figure 8b – X-Turret Gen3 with X-Loader v5

**2.2.2 Prototyping the Smart Tank**

The objective is to make the Smart Tank by assembling different components. For instance, a chassis, a turret, microcontrollers, a microcomputer and sensors. The Tank and its turret should be controlled to move and rotate by a corresponding controller respectively.

**Task 1:** Researching on and choosing the hardware for making the Tank

**Member in charge:** TAM Ho Kue

**Task description:**

Mecanum wheeled chassis and the tracked chassis are the most common chassis for a competitive robot car. Researching on the advantages and disadvantages of these two is required, and we evaluate and determine which one is the most suitable for our project. Moreover, we need to identify any other useful components for our project. For instance, microcontrollers and controllers.

**Work Done:**

In order to facilitate our research, we borrowed Mecanum wheeled chassis from HKUST Robotics Institute as shown in figure 9a and bought the tracked chassis as shown in figure 9b.

Based on our studies, we discover there are only five states for a tracked chassis.

1. Two tracks move forwards or backwards with the same speed.
2. Two tracks move forwards or backwards at different speeds.
3. One track moves forward and the other is locked.
4. A track moves forwards and another one moves backwards with the same speed.
5. A track moves forwards and another one moves backwards at different speeds.

The corresponding movement of the vehicle is as follows.

1. Move forwards or backwards.
2. Rotate around the central point of one axle.
3. Rotate around the central point of one axle.
4. Perform instant center of rotation.
5. Rotate around the central point of one axle.

Moreover, a tracked chassis has an excellent performance in sloped, smooth and uneven environments.

Compared to tracked chassis, Mecanum wheeled chassis has more states which can move the same as a tracked chassis, but it can also move sideways and move diagonally which are very crucial in a robot competition. However, Mecanum wheeled chassis is more expensive and performs badly on the frictionless or uneven ground [9].

Our target environment is flat and requires intensive responses, as well as the chassis borrowed from HKUST Robotics Institute has high acceleration and maximum speed. Therefore, Mecanum wheeled chassis is regarded as the most suitable for our project.

After that, we chose the DJI RoboMaster development board type A, which is STM32 ARM-core microcontroller-based, and an Arduino board to develop the embedded system. Moreover, we chose DJI Manifold 2-G (128GB) as our microcomputer and Intel Realsense D435i for image processing, as well as two controllers for the manual control of the tank's movement and its turret's rotation respectively.

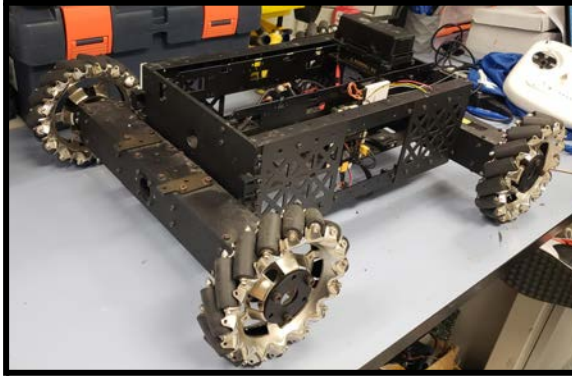


Figure 9a

Figure 9a – Mecanum wheeled chassis

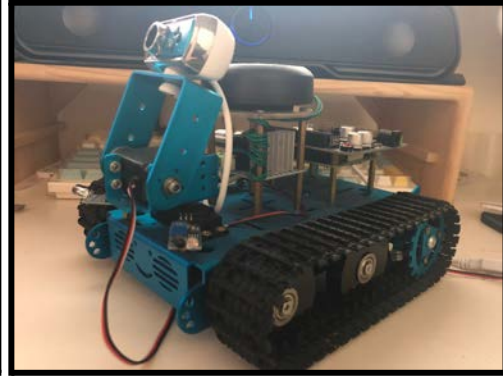


Figure 9b

Figure 9b – Tracked chassis

**Task 2:** Embedding all the components and sensors to make the Smart Tank prototype

**Member in charge:** All

**Task description:**

Since we got all necessary components for making the Tank, the next step is assembling them and developing the embedded system. The Tank and its turret should be controlled to move and rotate by a corresponding controller respectively.

**Work Done:**

The embedded system of the Tank can be separated into two parts, which are the turret and the chassis.

For the turret, since we have installed speed up motors, servo motors and a push-pull electromagnetic in the turret for doing some full-scale shooting and loading experiments in objective 1, we need to calibrate servo motors and enable manual control with a controller.

We assigned a number to slots of the revolving cylinder. In order to rotate the slot to be collinear with the barrel, we assigned different pulse-width modulation (PWM) signals to the servo motor. Moreover, we connect the Bluetooth receiver to the controller. Users can select one of three types of Nerf darts by pressing three different buttons on the controller, activating speed up motors and triggering the push-pull electromagnetic. For the rotation, we enable it by sending PWM signals to two servo motors for horizontal and vertical rotation respectively. Users can press arrow shape buttons for the corresponding direction of the rotation.

For the chassis, since Mecanum wheels have complex kinematics, we spent a lot of time understanding the mathematics and the equation behind it. The Tank successfully moved forwards and backwards, as well as moved sideways. However, it was nearly impossible for us to unleash the potential of the Mecanum wheels, as a lot of calibrations and calculations were required. Fortunately, an MPhil student reminded us the chassis we borrowed was from DJI and was used in a RoboMaster competition. Therefore, we found the package regarding the control of the chassis. Users can control the chassis with two joysticks on a DJI controller. It can move as we suggested on objective 2.2.2 Task 1. Therefore, the manual control system of the Tank is developed.

Lastly, we connected Arduino and DJI RoboMaster development board type A to Manifold 2G. The Smart Tank is ready to do image processing and run SLAM algorithms.



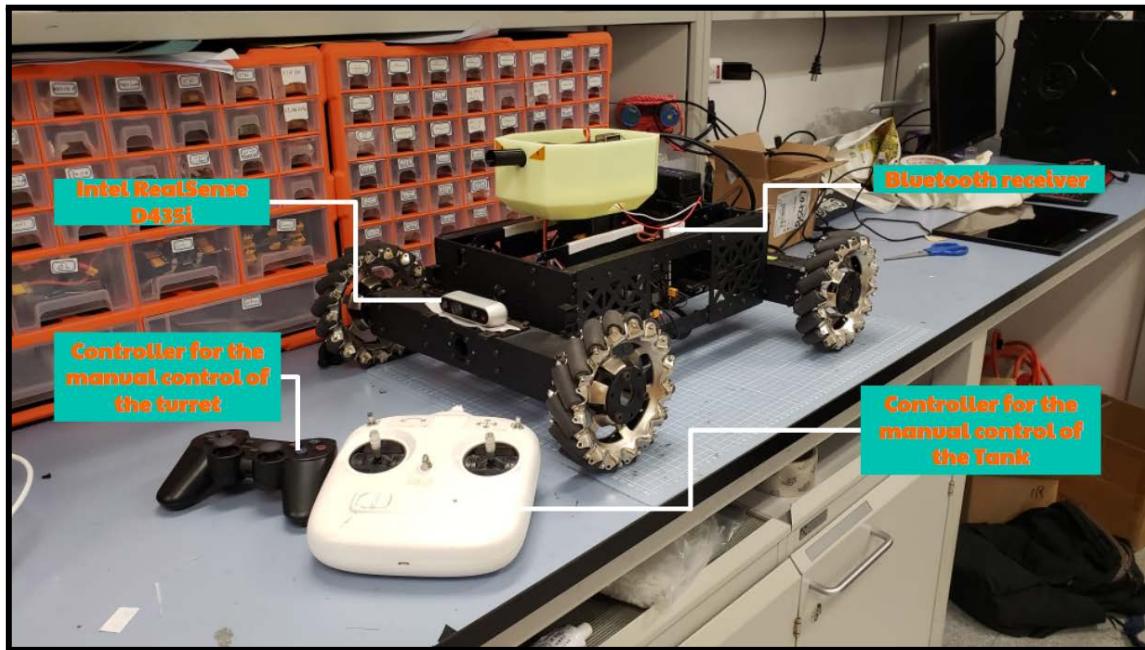


Figure 10 – Overview of the Smart Tank

### 2.2.3 Targets Identification and Tracking

This objective is to enable the Tank to identify and track the target.

**Task 1:** Setting up the environment of hardware (Intel RealSense D435i) and software(Ubuntu)

**Member in charge:** DENG Xingfa

**Task description:**

Setting up the environment on the Virtual Machine of Ubuntu 18.04 by installing the dependencies software.

**Work Done:**

For setting up the environment for object detection, we create a virtual machine of Ubuntu 18.04 on the hard drive. We install the compulsory dependency packages on Ubuntu.

The dependencies include:

- ROS Melodic
- Realsense SDK
- ROS Wrapper
- Python 3.6
- Tensorflow 1.14.0
- Keras 2.2.5
- H5py 2.10
- Matplotlib
- Opencv-python
- Yolo v3 package

After all the packages were correctly installed and the functions are successfully working by test, the software and hardware environment has been set up completely.

**Task 2:** Implementing the YOLO V3 [10] real-time object detection algorithm on the Smart Tank**Member in charge:** DENG Xingfa**Task description:**

Using the algorithm of YOLO V3 to implement the real-time object detection algorithm to achieve preliminary recognition of random objects.

**Work Done:**

Since the Intel RealSense D435i was the camera in our project, we should select the object detection algorithm that was compatible with the camera. Therefore, I selected the YOLO V3 as the object detection algorithm for our project. The YOLO V3 is compatible with Intel RealSense D435i and offers a real-time object detection function. Moreover, it was a user-friendly algorithm for data training of specific targets identification. After the software and hardware environment were set up completely, I modified the code inside the yolo.py file to use the data from Intel Realsense to implement the Yolo V3 object detection.

```

text_origin = np.array([left, top + 1])

# My kingdom for a good redistributable image drawing library.
for i in range(thickness):
    draw.rectangle(
        [left + i, top + i, right - i, bottom - i],
        outline=self.colors[c])
draw.rectangle(
    [tuple(text_origin), tuple(text_origin + label_size)],
    fill=self.colors[c])
draw.text(text_origin, label, fill=(0, 0, 0), font=font)
del draw

end = timer()
print(end - start)
return image

def close_session(self):
    self.sess.close()

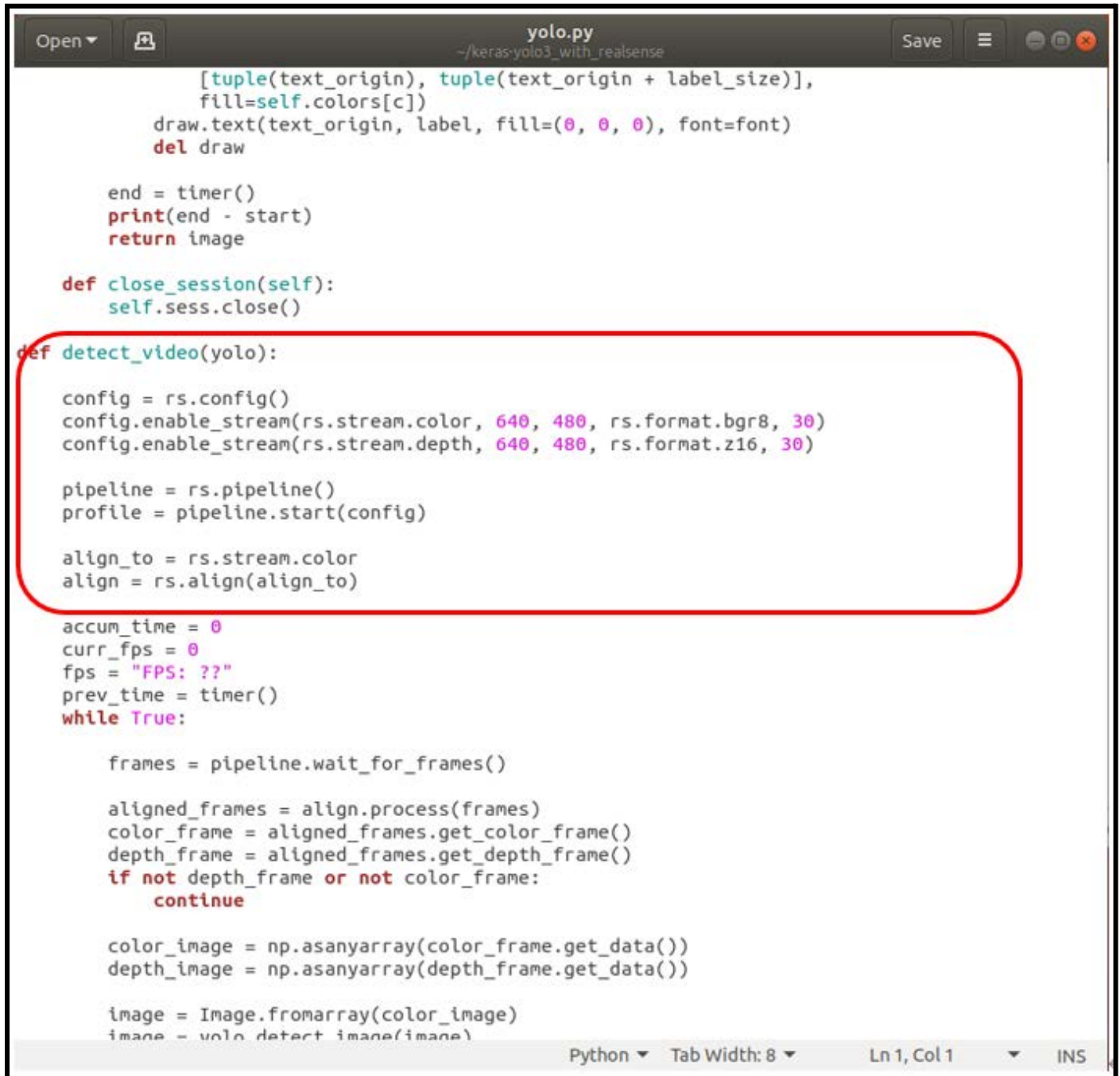
def detect_video(yolo, video_path, output_path=""):
    import cv2
    vid = cv2.VideoCapture(video_path)
    if not vid.isOpened():
        raise IOError("Couldn't open webcam or video")
    video_FourCC = int(vid.get(cv2.CAP_PROP_FOURCC))
    video_fps = vid.get(cv2.CAP_PROP_FPS)
    video_size = (int(vid.get(cv2.CAP_PROP_FRAME_WIDTH)),
                  int(vid.get(cv2.CAP_PROP_FRAME_HEIGHT)))
    isOutput = True if output_path != "" else False
    if isOutput:
        print("!!! TYPE:", type(output_path), type(video_FourCC), type(video_fps),
              type(video_size))
        out = cv2.VideoWriter(output_path, video_FourCC, video_fps, video_size)

    accum_time = 0
    curr_fps = 0
    fps = "FPS: ???"
    prev_time = timer()
    while True:
        return_value, frame = vid.read()
        image = Image.fromarray(frame)

```

Python Tab Width: 8 Ln 186, Col 1 INS

Figure 11a – Yolo.py before modification



```

yolo.py
~/keras-yolo3_with_realsense

    [tuple(text_origin), tuple(text_origin + label_size)],
    fill=self.colors[c])
draw.text(text_origin, label, fill=(0, 0, 0), font=font)
del draw

end = timer()
print(end - start)
return image

def close_session(self):
    self.sess.close()

def detect_video(yolo):
    config = rs.config()
    config.enable_stream(rs.stream.color, 640, 480, rs.format.bgr8, 30)
    config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)

    pipeline = rs.pipeline()
    profile = pipeline.start(config)

    align_to = rs.stream.color
    align = rs.align(align_to)

    accum_time = 0
    curr_fps = 0
    fps = "FPS: ??"
    prev_time = timer()
    while True:

        frames = pipeline.wait_for_frames()

        aligned_frames = align.process(frames)
        color_frame = aligned_frames.get_color_frame()
        depth_frame = aligned_frames.get_depth_frame()
        if not depth_frame or not color_frame:
            continue

        color_image = np.asanyarray(color_frame.get_data())
        depth_image = np.asanyarray(depth_frame.get_data())

        image = Image.fromarray(color_image)
        image = yolo.detect_image(image)

```

Python Tab Width: 8 Ln 1, Col 1 INS

Figure 11b – Yolo.py after modification

After the code modification, the image size and format are defined and the YOLO V3 will retrieve the data from the Intel Realsense D435i to implement the real-time object recognition.

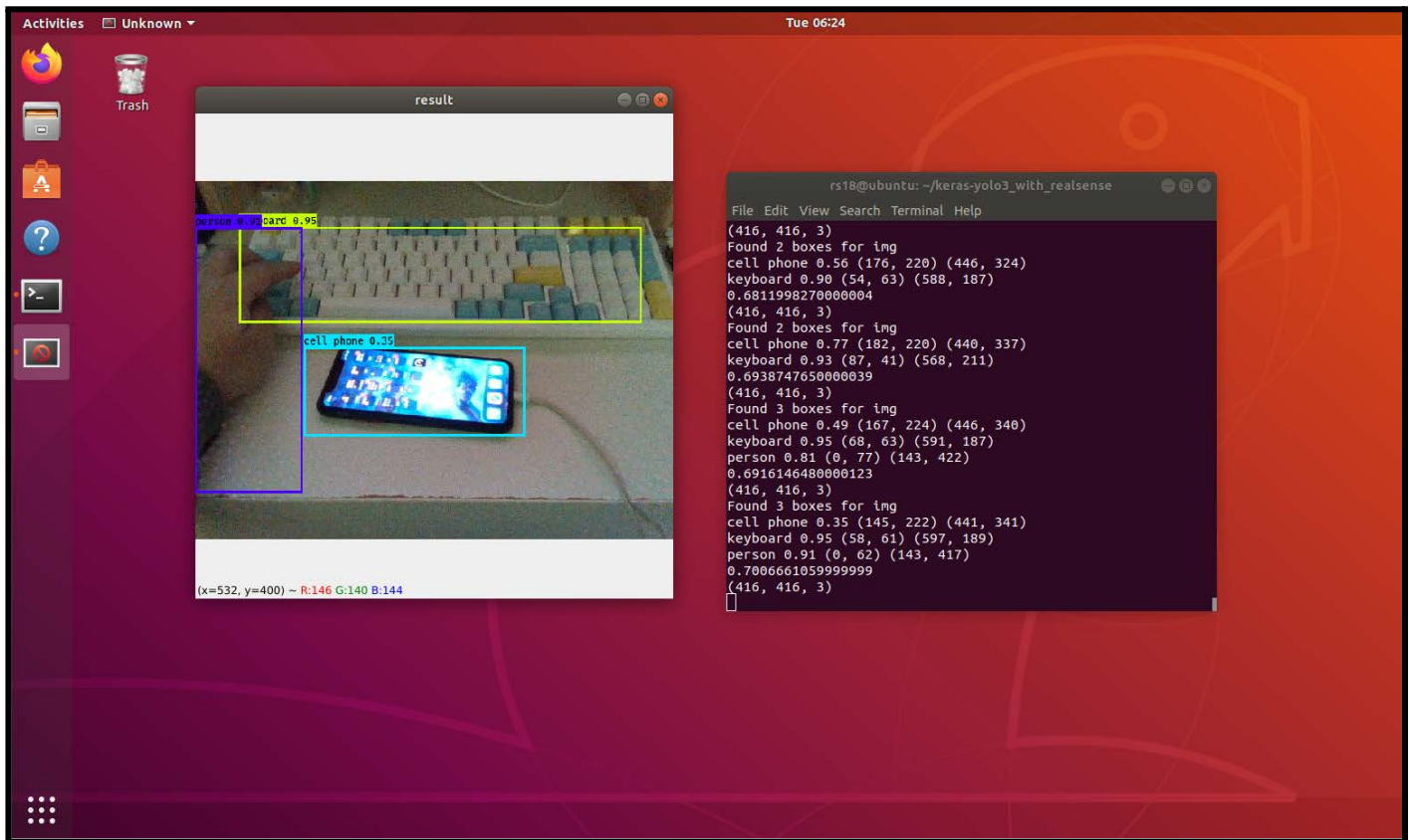


Figure 12 – Results of implementing YOLO V3

By implementing the YOLO V3, The camera of Realsense collected the data of the frames it captures and transmits the data to the computer. The Yolo.py will receive the data and compare the data with the object detection weight. Then, it will display the real-time detection results on the screen. In figure 12, we can see the result that the YOLO V3 identifies the keyboard, person and cell phone successfully.

### Task 3: Training the dataset for special object recognition of drone and tank

**Member in charge:** DENG Xingfa

#### Task description:

Collecting the png format pictures of drone and tank, using Labellmg to label all the pictures and train the data with weights. Therefore, I can use YOLO V3 to identify specific targets.

#### Work Done:

Since we are intended to detect the specific objects (Drone and Tank), I have to train the data to generate the weight for YOLO V3 to identify the Drone and Tank. Therefore, I collect 50 photos of Drone and 50 photos of Tank in png format (480 pixels \* 640 pixels) for the dataset. And I use the Labellmg to label all the pictures to identify the position and stance of the target in the picture. The dataset of training target and labelled results are shown in figure 13a and figure 13b.



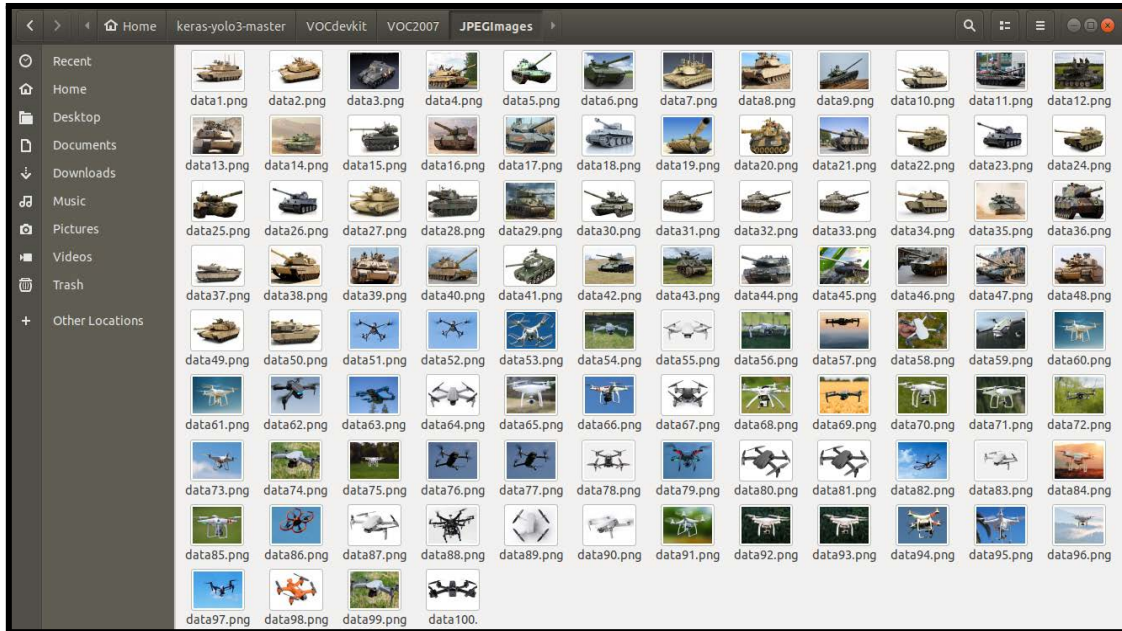


Figure 13a – Dataset of training targets

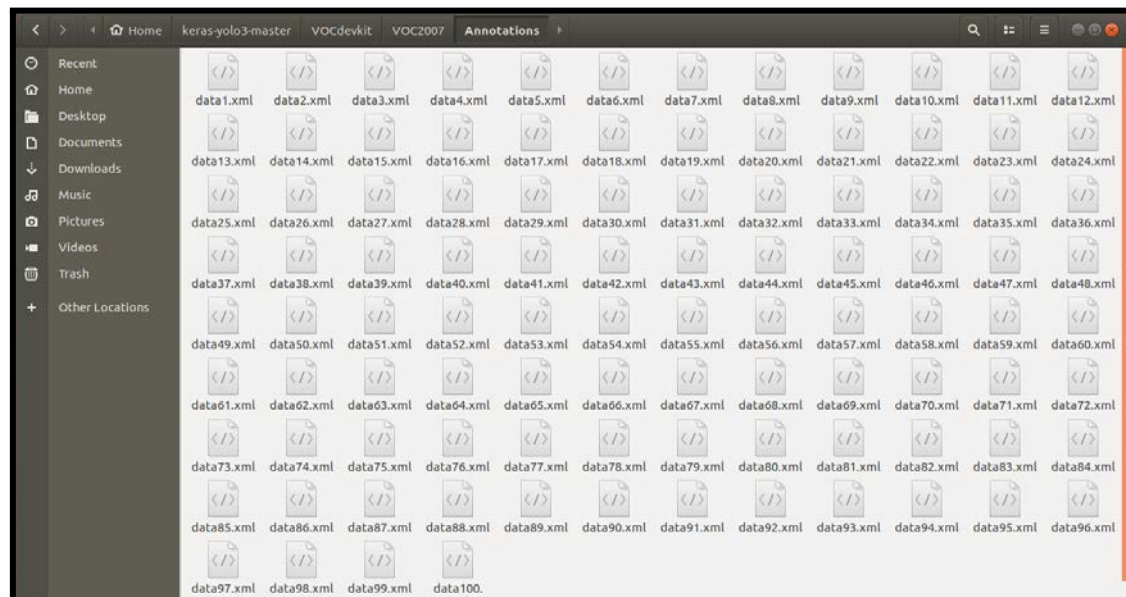


Figure 13b – Labeled results of dataset

After I labelled all the pictures, I utilized 80% of data for training, 10% of data for testing and 10% of data for valuation. Then, I trained the data with weight and select the generated weight with the least loss for the YOLO V3.

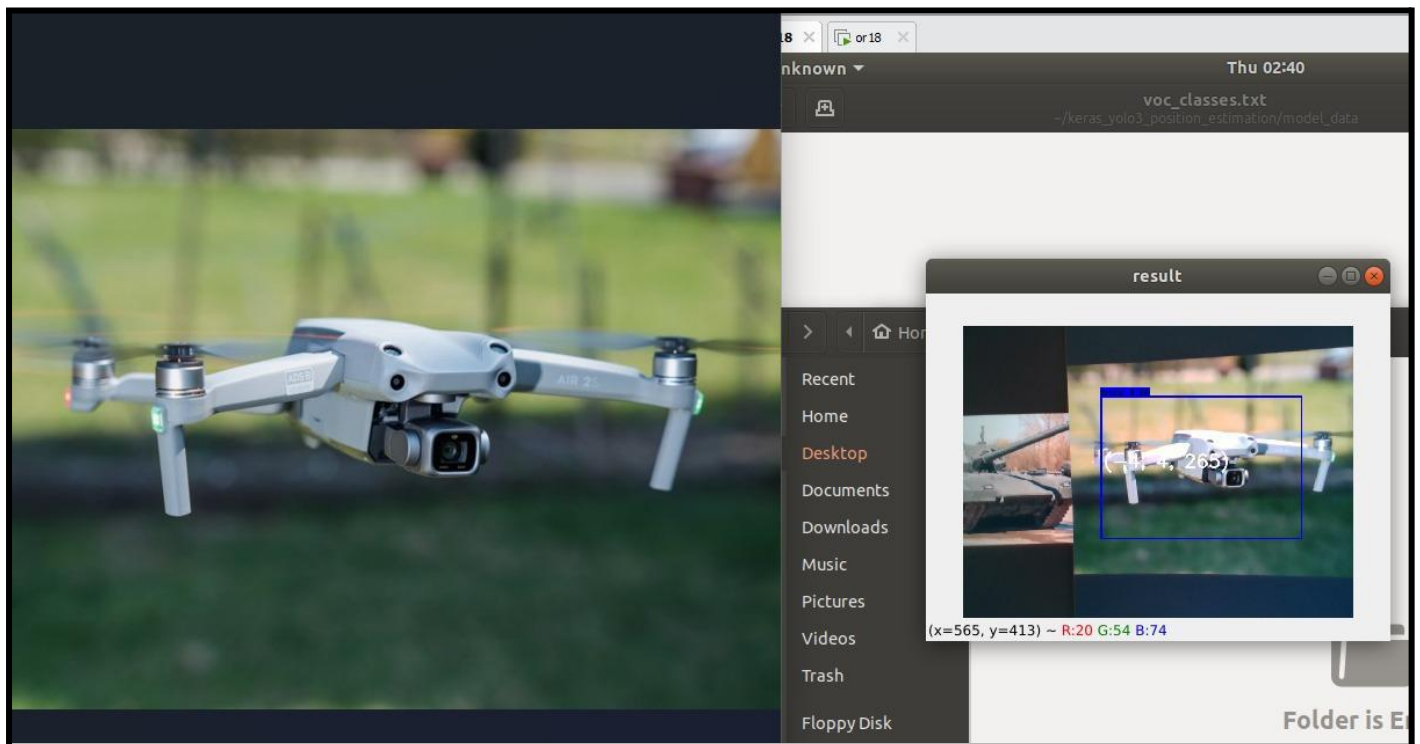


Figure 14a – Results of implementing YOLO V3 to detect Drone

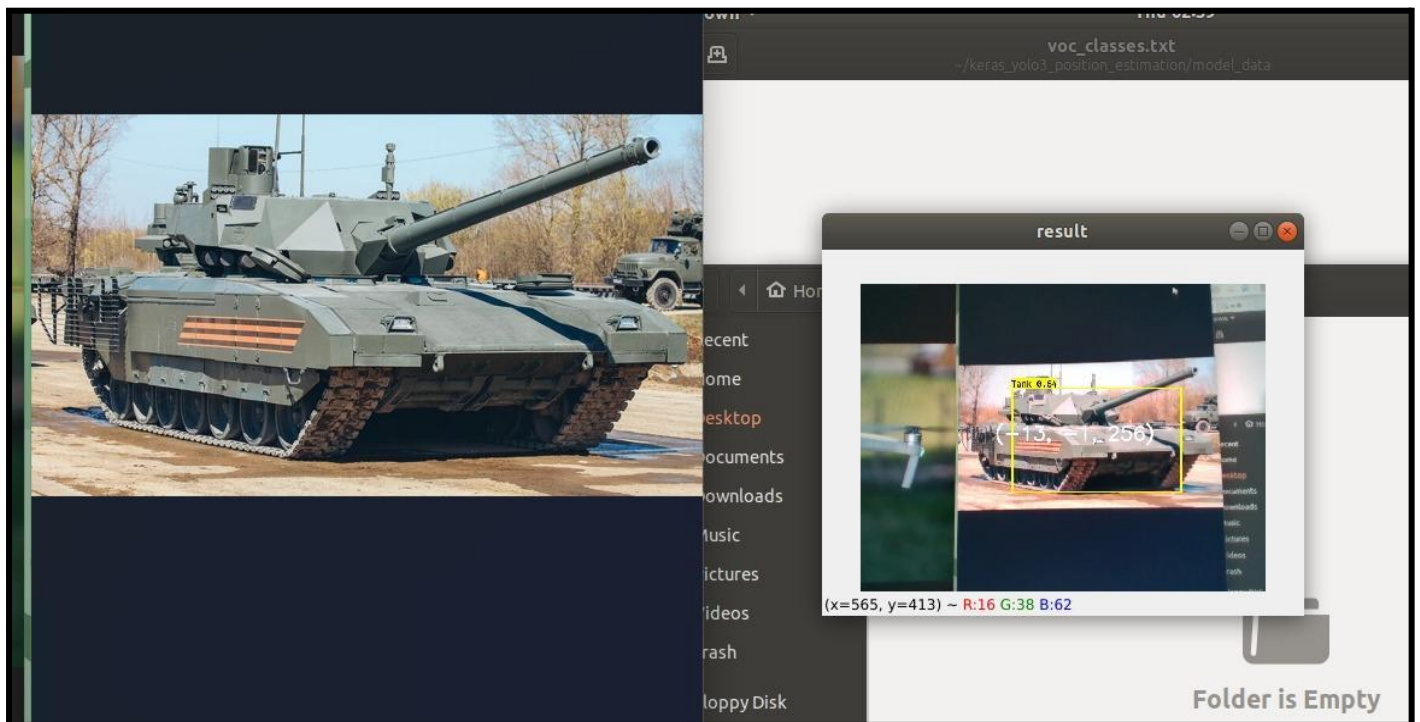


Figure 14b – Results of implementing YOLO V3 to detect Tank

After implementing the YOLO V3 with trained weight, the YOLO V3 identified the targets of Drone and Tank successfully. Moreover, the YOLO V3 could achieve real-time identification.

Then, when it was implemented on the Smart Tank. It could provide real-time identification for particular targets.

**Task 4:** Implementing the object position estimation algorithm on Smart Tank

**Member in charge:** DENG Xingfa

**Task description:**

After the object is detected, the algorithm will retrieve the centre position of the target, get the position estimation coordinates and display them on the screen.

**Work Done:**

The target coordinates could be obtained since the YOLO V3 could identify the target. Then, we could retrieve the pixel depth value with the target coordinates from the depth camera image which was captured by Realsense. Then, we could obtain the X, Y and Z coordinates results by calculation. Then, we displayed the coordinates result in the YOLO V3 in the real-time mode. The results could be seen in figures 14a and 14b. The YOLO V3 with position estimation algorithm could indicate the X, Y and Z coordinates when the target was identified. The above method was learned from a youtube channel [11] which offered the tutorial on Position Estimation on Intel Realsense D435i.

#### 2.2.4 Robot Navigation in a New Environment

This objective is to enable the Tank to move to a designated location in a new environment to localize the Tank and map the surrounding route or environment.

**Task 1:** Setting up the hardware environment

**Member in charge:** DENG Xingfa

**Task description:**

We will Install sensors (e.g. camera and Lidar) on the chassis. Then, we connect the sensors to the microcontroller and test the inputs from the sensors.

**Work Done:**

With the support of the Robotic Institute, we borrowed the Micro-computer of MANIFOLD 2-G and a wireless monitor for the Tank to implement SLAM navigation. In order to reduce the cost of our project, the Lidar was not used in our project and the only sensor for the Tank was the Intel Realsense D435i. The components for SLAM were shown in figure 15a and figure 15b. All the components were assembled correctly and tested functionally.



Figure 15a – Micro-computer of MANIFOLD 2-G



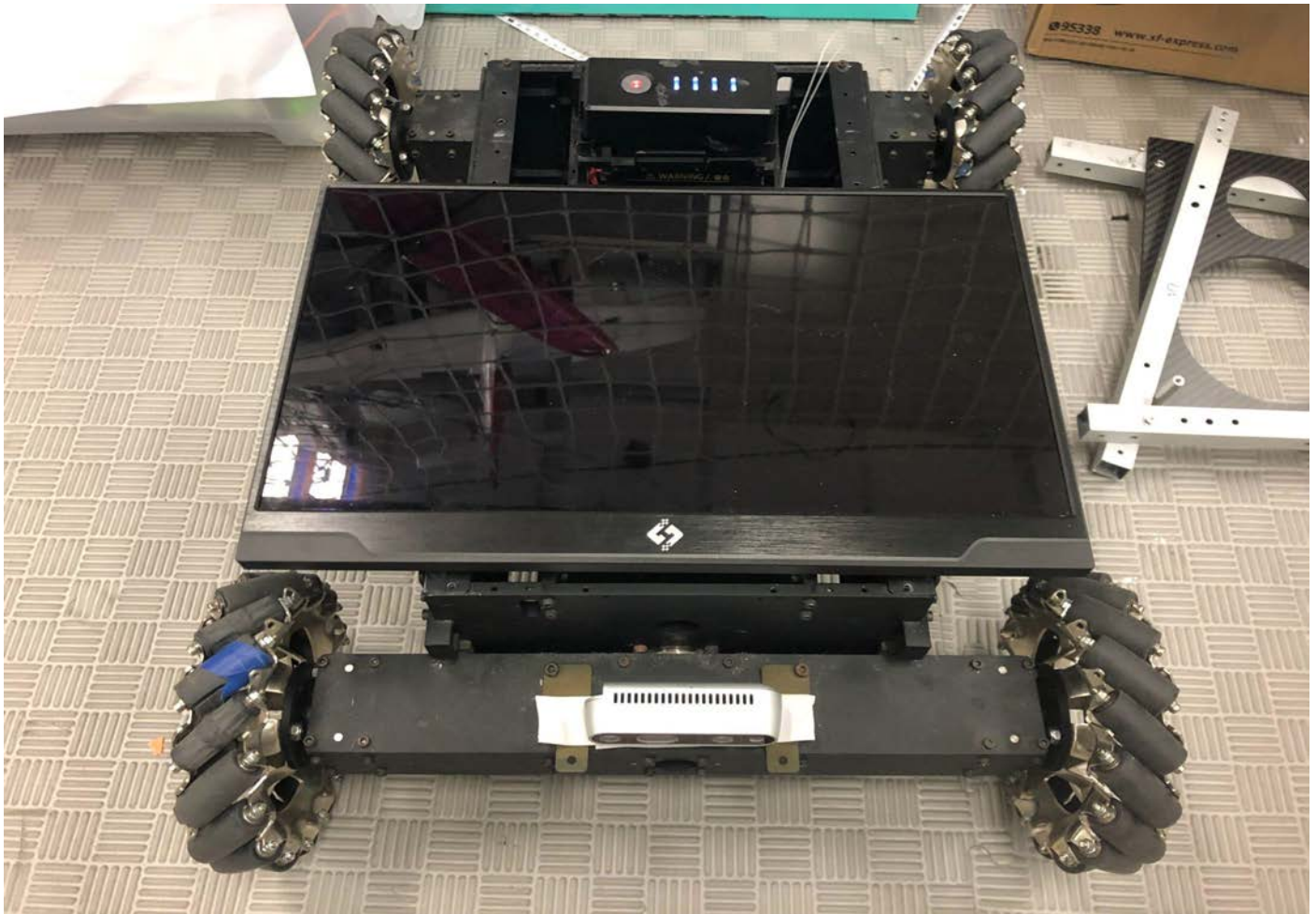


Figure 15b - Other components of wireless monitor, Intel Realsense d435i and prototype of the Tank

**Task 2:** Setting up the simulation environment for ROS in the Linux system on the Manifold 2G

**Member in charge:** DENG Xingfa

**Task description:**

Setting up the environment on the Manifold 2G for the SLAM development and testing all the functionality of all the software that required.

**Work Done:**

For setting up the environment for SLAM development, all the dependencies required were installed.

The dependencies include:

- ROS Kinetic
- Realsense SDK
- ROS Wrapper
- Python 3.6
- OpenCV 3.3.1
- Eigen 3.3.3
- VINS-Mono package [12]

After all the packages are correctly installed and the functions are successfully working by test, the software environment was set up completely.

**Task 3: Implementing SLAM algorithms****Member in charge:** DENG Xingfa**Task description:**

Compiling the SLAM algorithm VINS-Mono on ROS and measuring camera-odometry calibration, camera internal parameters and camera external parameters to construct the camera model for the Smart Tank. Finally, running the SLAM algorithm on the Tank.

**Work Done:**

In the course of ELEC3210 (Machine Learning and Information Processing for Robotics), we selected the Hector SLAM algorithm for the robot navigation simulation project and successfully obtained the mapping result on the simulated environment. The result was shown in figure 16. However, Hector mapping is one node for LiDAR-based SLAM. Since the Lidar sensor was not available for our project, we used the VINS-Mono SLAM algorithm which could use the Intel Realsense D435i as the sensor in our project.

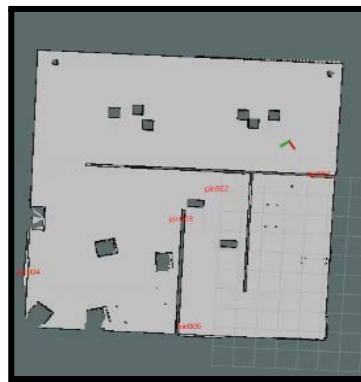


Figure 16 – The result of Hector mapping

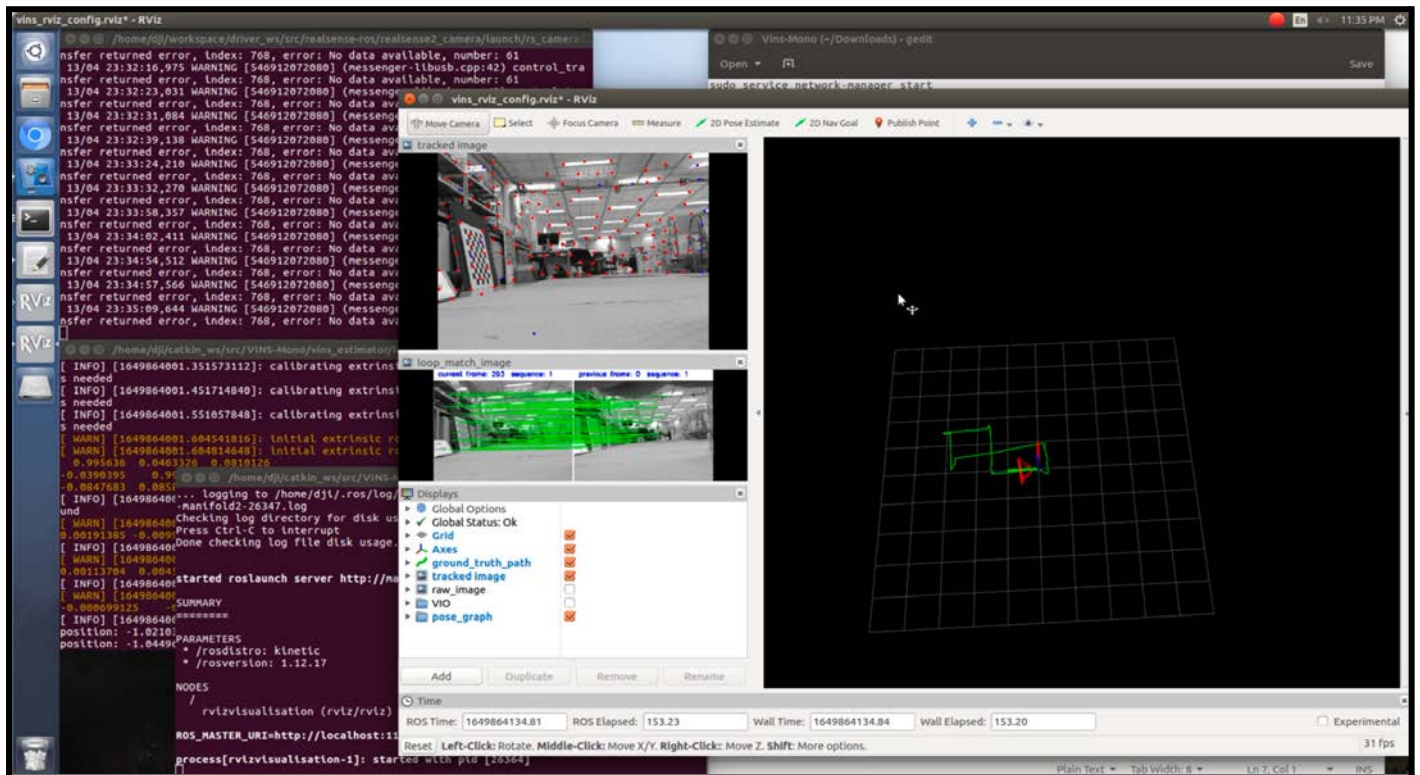


Figure 17 – The result of running VINS-Mono in a designed route

After testing the hardware and software environment, we calibrated the camera parameters and ran the VINS-Mono. After controlling the Tank to drive over a designed route, we received an accurate mapping result of the route in 3D. The result was shown in figure 17.

### 2.3 Smart Tank Evaluation & Discussion

Our main objective was to create a competitive robot tank that utilizes advanced robotics-related technologies including 3D printing, embedded system, computer vision and SLAM for STEM education. This allows users to learn from hands-on experience. The tank is supposed to perform shooting, moving, being controlled with controllers, identifying objects and running SLAM algorithms.

The first thing we evaluated was X-Loader v5 and X-Turret Gen3. According to the result in Subsection 2.2.1 Task 1, the shooting accuracy of X-Loader v5 is a 17cm dispersion where the Tank is 3 meters away from targets. This shooting performance is even greater than a 20cm dispersion which is a requirement for PhD students and our estimation. Moreover, X-Turret Gen3 strikes a balance between the cost and the system integrity. Unlike the HKD1400 printing cost of the X-Turret Gen1, X-Turret Gen3 costs HKD 625 which is less than half of Gen1. Since X-Turret Gen3 perfectly matches X-Loader v5, a few changes in the shooting and loading module may not be compatible with the turret. However, X-Loader v5 has convinced us and got approval from us for its extraordinary shooting performance. Therefore, this issue is minor.

The second we evaluated was the Smart Tank prototype. The Tank can move as we suggested in Subsection 2.2.2 Task 1. It has a rapid response speed and is confident to encounter any enemy on the battlefield. Although the Mecanum wheeled chassis is the most suitable for our project, it was borrowed from HKUST Robotics Institute. We cannot modify its superstructure by removing its metal case or welding our car shell on the chassis. It results in no space for putting our connection wires and even sticking the microcontrollers on the metal case. Therefore, the Tank has a less attractive appearance. We are planning to dress it with cardboard.

The third thing we evaluated was computer vision and SLAM. For the computer vision, since we intended to identify the specific targets (Drone and Tank), we trained the data by ourselves. In the end, the Smart Tank identifies the targets of the Drone and Tank successfully by running the YOLO V3 of trained weights. However, since we only used 80 pictures for data training. The data source is not enough to get high accuracy of results. Therefore, the accuracy of Object detection was not high enough with the limited data set. For the SLAM, we used the VINS-Mono as the SLAM algorithm of our project and used the Manifold 2-G as the Microcomputer of our project. Since VINS-Mono is perfectly compatible with Manifold 2-G and Intel Realsense D435i, Therefore the accuracy and stability of running VINS-Mono on the Smart Tank in a designed route were high enough to gain the clear mapping.

All in all, the development achieves our main objectives. We made a competitive robot tank that can shoot different types of Nerf darts, have a manual control system, identify different objects and run visual SLAM algorithms. The Tank is not yet perfect, there is room to improve. We are confident that it would be a good STEM education material for teachers to teach youth by offering hands-on robotics experience during the STEM education classes. The Smart Tank may ignite their passion for STEM subjects, even devoting themselves to future STEM careers.

## SECTION 3—CONCLUSION

This project aimed to design and build a robotic tank for STEM education purposes that allows secondary and college students to gain hands-on experience in robotic design. In the development of the Smart Tank, we customized the design of the ammo selection system and the body of the Tank. By applying the knowledge of 3D printing, we printed the shell and the revolving cylinder and other parts of the hardware design. Then, we selected the DJI board A as the microcontroller for our embedded system development. After we prepared all the electronic components for our Tank including the robotic car frame, wheels, servos and wireless controllers. We completed the embedded system development. Then we used the Intel Realsense D435i as the camera to implement Object detection and SLAM algorithms. First, we used the Manifold 2G as the microcomputer of our project. By setting up the software and hardware development, we successfully run the algorithm of the YOLO V3(Object detection algorithm) and VINS-Mono (SLAM algorithm) on the microcomputer. In the end, the Smart tank could select the specific bullet and rotate the turret horizontally and vertically using the ammo selection system. Also, the Tank could run the algorithm of YOLO V3 to identify the target and run the algorithm of VINS-Mono to implement SLAM in a designed route.

However, this is not a complete product yet. There are still a lot of flaws that should be fixed and room to improve.

For the software development, we would like to implement object detection and SLAM in the Manifold 2G (Ubuntu16.04) in the beginning. However, the Manifold 2G is an ARM architecture CPU and a lot of packages or dependencies only support the X86 architecture CPU. Due to the limited time, we decided to implement object detection on the virtual machine (Ubuntu 18.04). With more time for development, we would try to research the solution for implementing the YOLO V3 in the Manifold 2G and combining two techniques to implement in the Manifold 2G for a better user experience in the future.

For the hardware development, we would like to complete all the embedded system designs on the microcontroller of DJI board A in the beginning. However, the fifth wave pandemic limited our hard development time in HKUST for preventing the virus from spreading and increasing the delivery time. Then, we could not get our debugger of DJI board A on time. In order to complete our project, we use the Arduino board as the microcontroller of our project. And we completed all the embedded system development on the Arduino Uno r3. With more time for development, we would try to complete all the embedded system designs on the DJI board A to control the ammo selection system and the body of the Tank.

## REFERENCES

- [1] B. Stauffer, "What Are the 4 C's of 21st Century Skills?", Applied education systems, May 2020, Available: <https://www.aeseducation.com/blog/four-cs-21st-century-skills>
- [2] Research Office Legislative Council Secretariat, "Research Brief: Nurturing of local talent June 2020", P.10, June 2020, [online]. Available: <https://www.legco.gov.hk/research-publications/english/1920rb03-nurturing-of-local-talent-20200601-e.pdf>
- [3] HKSmart City Blueprint, "Hong Kong Smart City Blueprint 2.0", P.20, December 2020, [online]. Available: [https://www.smartcity.gov.hk/modules/custom/custom\\_global\\_js\\_css/assets/files/HKSmartCityBlueprint\(ENG\)v2.pdf](https://www.smartcity.gov.hk/modules/custom/custom_global_js_css/assets/files/HKSmartCityBlueprint(ENG)v2.pdf)
- [4] Research Office Legislative Council Secretariat, "Research Brief: Nurturing of local talent June 2020", P.7, June 2020, [online]. Available: <https://www.legco.gov.hk/research-publications/english/1920rb03-nurturing-of-local-talent-20200601-e.pdf>
- [5] STEMex Learning Centre, "STEMex Learning Centre," [Online]. Available: <https://hk.stemex.org/>
- [6] M. Donlon, "Tech company develops facial recognition camera for the STEM classroom", GlobalSpec, Feb 2020, Available: <https://www.cnet.com/news/smart-bulbs-vs-smart-switches-the-pros-and-cons-of-connected-lighting/>
- [7] R. Barone, "Robotics in education—Advantages, benefits & importance for kids", iD Tech, Apr 2021, Available: <https://www.idtech.com/blog/educational-benefits-robotics>
- [8] The Hong Kong Federation of Youth Groups, "The Hong Kong Student Science Project Competition (HKSSPC) 2021", [Online]. Available: <https://hksspc.hkfyg.org.hk/en/competition-detail/hksspc2021/>
- [9] E. Wu, "Best Guide for Building A Robot Car", Seeedstudio, 2019, Available: <https://www.seeedstudio.com/blog/2019/07/27/best-guide-for-building-a-robot-car/>
- [10] J. Redmon, "YOLO: Real-Time Object Detection", pjreddie.com, Available: <https://pjreddie.com/darknet/yolo/>
- [11] robot mania. "Position estimation of an object with YOLO using RealSense" YouTube, Aug 2013, Available: <https://www.youtube.com/watch?v=--81OoXMvIw>
- [12] T. Qin, S. Cao, S. Shen, P. Li. "YHKUST-Aerial-Robotics/VINS-Mono", 2019, Available: <https://github.com/HKUST-Aerial-Robotics/VINS-Mono>

## APPENDICES

### Appendix A – Final Project Schedule

Table 2. Smart Tank Schedule

Objective Statements	Task	Group Member in charge	WK1 1 Sep	WK2 6 Sep	WK3 13 Sep	WK4 20 Sep	WK5 27 Sep	WK6 4 Oct	WK7 11 Oct	WK8 18 Oct	WK9 25 Oct	WK10 1 Nov	WK11 8 Nov	WK12 15 Nov	Wk 13 22 Nov	Wk 14 29 Nov
Prototyping a well-functioning tank turret																
	Designing and prototyping a shooting and loading module, and developing an ammunition selection algorithm	TAM Ho Kuen														
	Designing and prototyping a turret case	TAM Ho Kuen														
Prototyping the Smart Tank																
	Researching on and choosing the hardware for making the Tank	TAM Ho Kuen														
	Embedding all the components and sensors to make the Smart Tank prototype	TAM Ho Kuen,														
Targets Identification and Tracking																
	Setting up the environment of hardware (Intel RealSense D435i) and software(Ubuntu)	DENG Xingfa														
	Implementing the YOLO V3 real-time object detection algorithm on the Smart Tank	DENG Xingfa														

	Training the dataset for special object recognition of drone and tank	DENG Xingfa														
Robot Navigation in a New Environment																
	Setting up the hardware environment	DENG Xingfa														
	Setting up the simulation environment for ROS in the Linux system on the Manifold 2G	DENG Xingfa														
	Implementing SLAM algorithms	DENG Xingfa														

Table 2. Smart Tank Schedule (Continued)

Objective Statements	Task	Group Member in charge	Wk 15 7 Feb	WK16 14 Feb	WK17 21 Feb	WK18 28 Feb	WK19 7 Mar	WK20 14 Mar	WK21 21 Mar	WK22 28 Mar	WK23 4 Apr	WK24 11 Apr	WK25 18 Apr	WK26 25 Apr	WK27 2 May
Prototyping a well-functioning tank turret															
	Designing and prototyping a shooting and loading module, and developing an ammunition selection algorithm	TAM Ho Kuen													
	Designing and prototyping a turret case	TAM Ho Kuen													
Prototyping the Smart Tank															
	Researching on and choosing the hardware for making the Tank	TAM Ho Kuen													
	Embedding all the components and sensors to make the Smart Tank prototype	TAM Ho Kuen,													
Targets Identification and Tracking															

	Setting up the environment of hardware (Intel RealSense D435i) and software(Ubuntu)	DENG Xingfa													
	Implementing the YOLO V3 real-time object detection algorithm on the Smart Tank	DENG Xingfa													
	Training the dataset for special object recognition of drone and tank	DENG Xingfa													
<b>Robot Navigation in a New Environment</b>															
	Setting up the hardware environment	DENG Xingfa													
	Setting up the simulation environment for ROS in the Linux system on the Manifold 2G	DENG Xingfa													
	Implementing SLAM algorithms	DENG Xingfa													



## Appendix B – Budget

Table 3. Budget

Items	Cost
Camera with IMU	Borrowed from HKUST RI
Servo motors	\$450
Microcontroller	Borrowed from HKUST RI
Microcomputer	Borrowed from HKUST RI \$30
Turret of the tank	\$625
Chassis of the tank	Borrowed from HKUST RI
DC motors	\$30
Li-ion batteries	Borrowed from HKUST RI
<b>TOTAL</b>	<b>\$1105</b>

**Appendix C—Meeting Minutes**

on 30 April (Fri.)

Meeting 1:

Date: 30/04/2021

Time: 10:30am

Location: Zoom

Attendees: TAM Ho Kuen, DENG Xingfa

Absent: None

Minutes taken by: TAM Ho Kuen

- To present the preliminary proposal to Prof. Shen.
- To ask for Prof. Shen's approval on admitting our proposal.
- To discuss the technical details of our project.
- To discuss the support for implementing our project.
- To evaluate the difficulties we are likely to encounter.

Table 1. Action Items for Next Meeting

Action Item to be completed	By when	By whom
Correcting the preliminary proposal and presenting its novelty	May. 20 <sup>th</sup>	TAM Ho Kuen
Building connections with Prof. Shen's PhD students for further consultation	May. 20 <sup>th</sup>	All
Visiting HKUST Robotics Institute and find a workplace for our project	May. 20 <sup>th</sup>	DENG Xingfa

Next Meeting: Sep 11, 15:15, HKUST Robotics Institute

Meeting 2:

Date: 11/09/2021

Time: 15:15pm

Location: HKUST, Robotics Institute

Attendees: TAM Ho Kuen, DENG Xingfa

Absent: None

Minutes were taken by: TAM Ho Kuen

- To finalize the design of the Smart Tank.
- To distinguish the work of each other.
- To make an appointment with Mr. Shaozu Cao to gain further advice on the project.

Table 1 Action Items from Previous Meeting

Action Item to be completed	By when	By whom	Status
STEM education Research	Sept. 10 <sup>th</sup>	All	Completed
Proposal Report introduction	Sept. 10 <sup>th</sup>	DENG Xingfa	In progress (80% complete)
Proposal Report methodology	Sept. 10 <sup>th</sup>	TAM Ho Kuen	In progress (70% complete)

Table 2. Action Items for Next Meeting

Action Item to be completed	By when	By whom
Literature review	Sept. 12 <sup>th</sup>	DENG Xingfa
Proposal Report methodology	Sept. 12 <sup>th</sup>	TAM Ho Kuen
Proposal Report objectives 1,2	Sept. 13 <sup>th</sup>	DENG Xingfa
Proposal Report objectives 3,4	Sept. 13 <sup>th</sup>	TAM Ho Kuen
Finalize the proposal report	Sept. 14 <sup>th</sup>	All

Next Meeting: Jan 06,16:30, HKUST Robotics Institute

### Meeting 3:

Date: 06/01/2022

Time: 16:30 pm

Location: HKUST Robotics Institute

Attendees: TAM Ho Kuen, DENG Xingfa

Absent: None

Minutes taken by: DENG Xingfa

- DENG Xingfa is working on testing the algorithm VINS-Mono.

- TAM Ho Kuen is working on the simulation with Gazebo.
- Mr. Cao suggested we can use DJI Manifold 2-G (128GB) for running the VINS-Mono.
- Mr. Jiang will provide technical support in VINS-Mono if we encounter any problems.

Table 1. Action Items from Previous Meeting

Action Item to be completed	By when	By whom	Status
Building the prototype of Ammo Selection System	Jan. 4 <sup>th</sup>	DENG Xingfa	Completed
Simulation of the Hector mapping Algorithm in the ROS	Jan. 4 <sup>th</sup>	TAM Ho Kuen	Completed
Finalize the coming working plan with Mr. Jiang	Jan. 6 <sup>th</sup>	All	Completed

Table 2. Action Items for Next Meeting

Action Item to be completed	By when	By whom
Assemble the robotic car	Feb.7th	DENG Xingfa
Writing the code to control the car	Feb.7th	DENG Xingfa
Installing the Lidar on robotic car	Feb.14th	TAM Ho Kuen
Simulating the SLAM in a real environment	Feb.14th	All

Next Meeting: Feb 20,16:30, HKUST Robotics Institute

## Appendix D—Group Members' Contributions

Group member: TAM Ho Kuen

In this project, I was the leader of the hardware team, who is in charge of the hardware design of the Tank. Therefore, I was responsible for objectives 1 and 2. Moreover, I participated in the software team led by DENG Xingfa as well. My contributions to the FYP are listed in perspectives of objectives as follows.

For objective 1, my duty was doing the 3D design of the shooting and loading module, prototyping it with a 3D printer, and examining its shooting performance and system integrity. In the design phase, I kept close contact with a PhD student who provided consultation on our project. He is an experienced hardware designer. Therefore, I consulted his comments during my design process. Based on his sharing and comments, I learned the importance of modular design, and the use of different 3D printing materials. Every time I finished a design, I 3D printed out and tested its performance, and then I submitted the report to him for optimization. For the shooting and loading module, I have done 5 designs that have major differences from each other, and numerous minor corrections or updates. Additionally, I worked with DENG Xingfa to develop the ammunition selection algorithm, my duty was drafting the algorithm and testing the algorithm.

For objective 2, my duty was doing the 3D design of the turret, as well as choosing the chassis, microcontroller and other hardware. The design process was similar to what I mentioned above. The difference is that I needed to care about the price, as we purchased the printing services. The first design is too complex and costly, while the second one is too radical in cutting down the cost, resulting in compatibility with other components. The third design strikes a balance between the cost and the performance, I sent the design to the 3D printing services provider and printed it. Moreover, I conducted research on and tested different chassis, microcontrollers and sensors. I consulted PhD students' comments and chose the most suitable components for our project. After that, I worked with DENG Xingfa to develop the embedded system.

For objective 3, my duty was to set up the hardware for computer vision. I firstly consulted a PhD student, whose research interest is algorithm and path planning, regarding the performance and characteristics of different cameras. Eventually, I picked the Intel RealSense D435i which is a stereo camera with an inbuilt inertial measurement unit (IMU). After that, I helped DENG Xingfa to find pictures for the training as well.

For objective 4, my duty was to help DENG Xingfa implement robot navigation. At first, we wanted to implement it with an ordinary approach by using Lidar, because there is more research and algorithms for Lidar based SLAM. However, it can be further optimized by using the Intel RealSense D435i to run visual SLAM. Hence, we can reduce one sensor to do the sensor fusion. After that, I helped DENG Xingfa to test the results.

All in all, as a leader in the hardware team and a participant in the software team, I tried my best to contribute to the FYP. Making a robot tank was one of my goals during my university study. I am proud of the Tank fully designed and made by us two. In addition, I appreciate DENG Xingfa for working with me to implement this crazy idea. Lastly, I would like to thank HKUST RI, Prof. SHEN and his students.

Group member: DENG Xingfa

In this project, I was the leader of the software team, who is mainly in charge of the software development of the Tank. Therefore, I was responsible for objectives 3 and 4 in the project. Moreover, I offered software support in the hardware design. My contributions to the FYP are listed in perspectives of objectives as follows.

For objective 1, my duty was researching the prototype design, giving comments on the feasibility of the design and assisting in the embedded system development of the ammo selection system. After TAM Ho Kuen finalized the design and purchased all the components, I cooperated with TAM Ho Kuen with the embedded system design. My duty was focusing on coding to control the servos for the ammo selection system and the ps2 controller for remote wireless controlling the turret.

For objective 2, my duty was to give the comments on the prototype of Smart Tank and test the prototype. After TAM Ho Kune completed the design of the Tank and finished prototyping the Tank. I was in charge of testing the stability of the prototype. And I test the prototype with the SLAM algorithm and Realsense D435i. After the testing, I thought the design fit the needs of our project.

For objective 3, my duty was to select the object detection algorithm and train the data for achieving the specific targets real-time recognition. After TAM Ho Kuen consulted with a PhD student and selected the camera of Intel Realsense D435i for our project, I researched the object detection algorithm for the camera and selected the algorithm of YOLO V3 for our project. Afterwards, I studied the topic of Ubuntu, ROS, OpenCV, python, labellmg. Then, I set up the software and hardware environment and ran the package of YOLO V3 by using Intel Realsense D435i and successfully identified the targets. Then, I decided to train the data for specific targets (Drone and Tank) recognition. After I collected and labeled all the data, I trained the data with weights. Then I used the trained weight with the least valuation loss to run the algorithm of YOLO V3. Eventually, the algorithm was able to identify the Drone and Tank on a real-time basis.

For objective 4, my duty was to select the SLAM algorithm and implement the SLAM for our project. After receiving the suggestion from TAM Ho Kuen and a PhD student. I decided to use the algorithm of VINS-Mono instead of the Hector mapping for our project since the Lidar is not available in our project and Intel Realsense D435i was the only sensor available in our project. Then, I studied the manipulation of the Micro-Computer of Manifold 2G and started running the VINS-Mono on the RVIZ by using the simulation package. Then I installed the dependencies of Intel RealSense D435i and calibrated the parameters of the camera. With the assistance of the PhD student and TAM Ho Kuen. I successfully ran the VINS-Mono on the prototype of the Tank in the designed route and gained an accurate 3D mapping.

As a leader in the software team and a member of this FYP, I tried to perform my work at the highest level and be a good teammate to make the collaboration smoother. I wished my FYP could be a seed to encourage more young talent to pursue a STEM career in the future. In addition, I appreciated that TAM Ho Kune was willing to assist me in various aspects of FYP and I would like to thank the assistance of HKUST Robotic Institute, Prof. SHEN and his students.

**Appendix E—Deviation(s) from the proposal and progress report and supporting reason(s)**

For the software part, we would like to implement object detection and SLAM in the Manifold 2G (Ubuntu16.04) in the beginning. However, the Manifold 2G is an ARM architecture CPU and a lot of packages or dependencies only support the X86 architecture CPU. Due to the limited time, we decided to implement object detection on the virtual machine (Ubuntu 18.04) and we would try to combine two techniques to implement in the Manifold 2G in the future.

For the hardware part, we would like to complete all the embedded system design on the microcontroller of DJI board A in the beginning. However due to the fifth wave pandemic which limited our hard development time in HKUST for preventing the virus spreading and increased the delivery time. Then, we could not get our debugger of DJI board A on time. In order to complete our project, we use the Arduino board as the microcontroller of our project. And we completed all the embedded system development on the Arduino uno r3. With more time for development, we would try to complete all the embedded system design on the DJI board A to control the ammo selection system and the body of the Tank.

## Appendix F—Monthly Reports

## Monthly Report for ECE FYP/FYT

<b>Project Code:</b>	<b>SSJ03a-21</b>	<b>Supervisor(s):</b>	Prof. SHEN Shaojie
<b>Project Title:</b>	<b>Self-proposed Project</b>		
<b>Group Member(s):</b>	1)TAM Ho Kuen                                  2)DENG Xingfa		
<b>Reporting Period:</b>	Report #1        ■              11 Oct (Fall) Report #2                                  □ Nov (Fall) Report #3                                  □ Feb     (Spring)  (please attach Reports #1-2 to the Progress Report to be submitted in Jan) (please attach Reports #3 to the Final Report to be submitted in Apr)		
<b>Progress Report:</b>	<p>Our group believes the systematic approach is the best way to work on a sophisticated project with careful design. There are several internal deadlines for the corresponding objectives and the reviews by another groupmate for the validation. There is no compromise in our project.</p> <p>Up to 26th October 2021, we finished the two stage 1 objectives which are market research and an in-depth consultation with experts. They are going to be addressed in the following.</p> <p>To begin with the market research, we have done it in both quantitative and qualitative approaches. An engineering project is supposed to be designed for tackling real life problems and fulfill the niche in the world. Since our project is for STEM education, we interviewed the parents whose child/children aged 18 or below as well as received above 30 responses from questionnaires. The data and the comments from parents revealed that our project could benefit and appeal to children. Our project could impact the STEM market.</p> <p>After that, we had an in-depth consultation with the PhD student Mr. CAO. The implementation, which brings abstract ideas to practical use, is not easy. We discussed many details including the hardware design, sensors, algorithms and validations. We received insightful advice from him and planned timelines for our project with the aid of him.</p> <p>All in all, we finished stage 1 and are able to step further into stage 2. We think it is reasonable to spend two months on stage 1 which is approximately 15% of our project.</p>		
<b>Future Plan:</b>	Stage 2 <ul style="list-style-type: none"> <li>• To finish building the prototype of the turret.</li> <li>• To run the simulated robot car on the ROS.</li> <li>• To purchase the required components for the project.</li> <li>• To do the research on computer vision.</li> </ul>		



Group  
Representative's  
Signature:




(Version 2020-10)

Project Code:	SSJ03a-2 1	Supervisor(s):	Prof. SHEN Shaojie
Project Title:	Self-proposed Project		
Group Member(s):	1)TAM Ho Kuen                      2)DENG Xingfa		
Reporting Period:	Report #1 <input type="checkbox"/> Oct (Fall) Report #2 <input checked="" type="checkbox"/> Nov (Fall) Report #3 <input type="checkbox"/> Feb (Spring) (please attach Reports #1-2 to the Progress Report to be submitted in Jan) (please attach Reports #3 to the Final Report to be submitted in Apr)		
Progress Report:  <ul style="list-style-type: none"> <li>List the work completed in this reporting period.             <ul style="list-style-type: none"> <li>Identify the major difficulties encountered.</li> </ul> </li> <li>Comment on the overall progress.</li> </ul>	<p>In our previous report, we mentioned our main focus in this month is to build the prototype of the turret and do the research on computer vision. The progress is below.</p> <p>For the turret, we have tried various plans and constructed the prototype. Under the consideration of hardware complexity and stability, we adopted the revolving cylinder design. For the proof of concept (POC), we used cardboard to build the simplified version for now. It will be further upgraded to better materials and eventually will be 3D printed. For the control part, we used STM32 to test our hardware because STM32 has better capability on debugging, but we will transfer the platform to Arduino to combine with our Tank chassis. Up till now, we are still working on optimizing the hardware design in terms of mechanical parts and the control.</p> <p>After that, we have finished the research on computer vision and gained basic understanding in this aspect. We have read some books and videos including Zhou Zhihua's Machine Learning , Rafael C. Gonzalez's Digital Image Processing and Stanford University online course Convolutional Neural Networks for Visual Recognition. Now we have fundamental knowledge on computer vision but lack hands-on experience. Therefore, we will figure out how we can apply computer vision in practice, especially the real world on FYP.</p>		
Future Plan:  <ul style="list-style-type: none"> <li>Write down the working plan for the next reporting period.</li> </ul>	Stage 3 <ul style="list-style-type: none"> <li>To optimize the hardware design in terms of mechanical parts and the control.</li> <li>To practice computer vision on ROS.</li> </ul>		

Group  
Representative'  
s Signature:

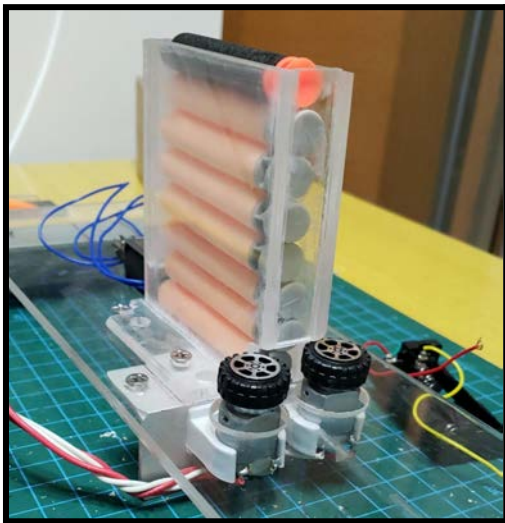
*Tamara*

(Version 2020-10)

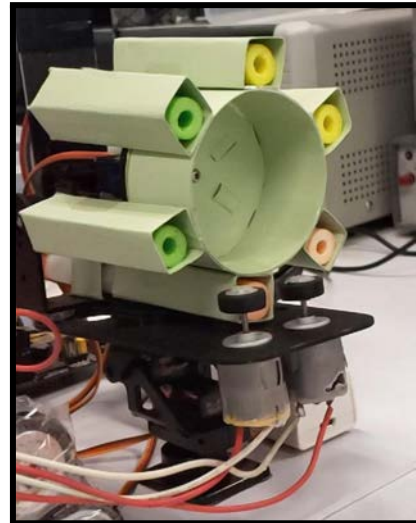
Project Code:	SSJ03a- 21	Supervisor(s):	Prof. SHEN Shaojie
Project Title:	Self-proposed Project		
Group Member(s):	1)TAM Ho Kuen                      2)DENG Xingfa		
Reporting Period:	Report #1 <input type="checkbox"/> Oct (Fall) Report #2 <input type="checkbox"/> Nov (Fall) Report #3 <input checked="" type="checkbox"/> Feb (Spring) (please attach Reports #1-2 to the Progress Report to be submitted in Jan) (please attach Reports #3 to the Final Report to be submitted in Apr)		
Progress Report:  List the work completed in this reporting period. Identify the major difficulties encountered. Comment on the overall progress.	<p>For the current progress, we have designed to use the RoboMaster Board A as the microcontroller of the prototype. And we have made the 3D printing design of the ammo selection system and found the 3D printer to make the prototype of the turret. As well as we have purchased all the electronic components for our embedded system design.</p> <p>From the software parts, we have implemented the computer vision using the Yolo v3 algorithm on the camera of Realsense D435i. And we have cleaned and labelled the data for data training to recognise the specific targets. For the SLAM, we have implemented the slam algorithm of VINS-Mono via the camera of Realsense D435i.</p> <p>For the difficulties, we have to implement the SLam and computer vision via the microcontroller. In our case, we select the Manifold 2G as the microcontroller. However, the development on the systems of Manifold 2G is quite difficult to debug. Moreover, for hardware design, we have to customize the 3D design. Therefore, we may face the actual bias on the hardware design.</p> <p>For the overall progress, although we still have some stages to complete, the overall progress is still optimistic for both software and hardware design.</p>		
Future Plan:  Write down the working plan for the next reporting period.	Future stage <ul style="list-style-type: none"> <li>• To assemble the components on the prototype</li> <li>• To implement Slam and Computer vision algorithms on the microcontroller of Manifold 2G</li> <li>• To control all the electronic components via the microcontroller of RoboMaster Board A</li> </ul>		
Group Representative's Signature:			

(Version 2020-10)

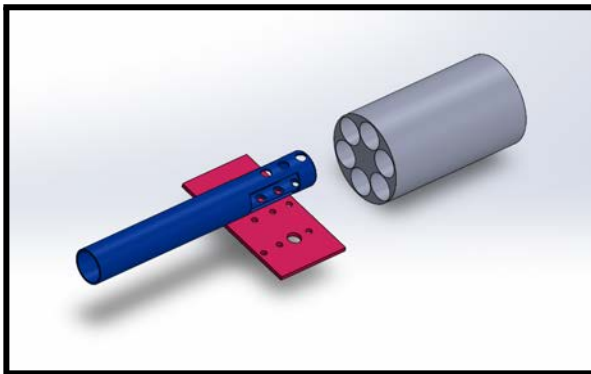
**Appendix G – Different designs of the shoot and loading module for 2.2.1 Task 1**



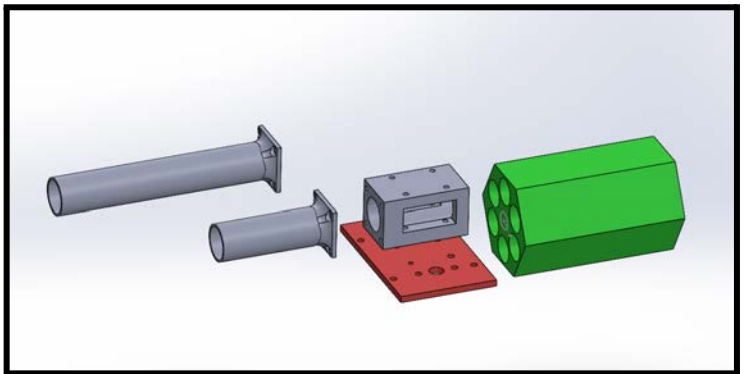
X-loader v1



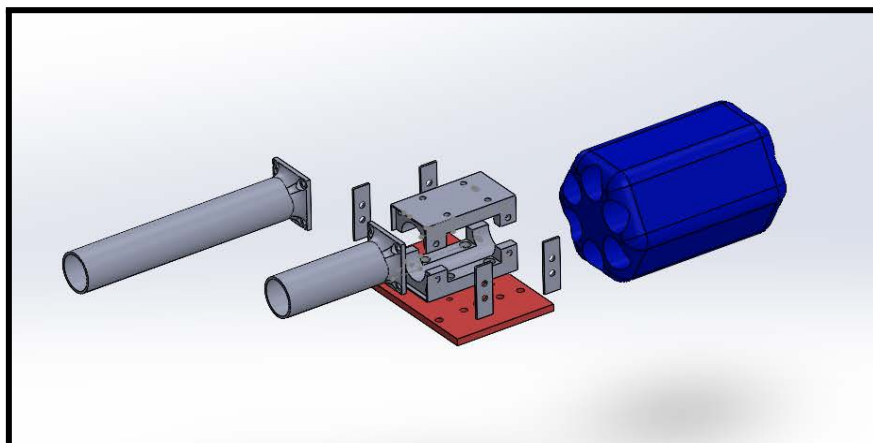
X-loader v2



X-loader v3

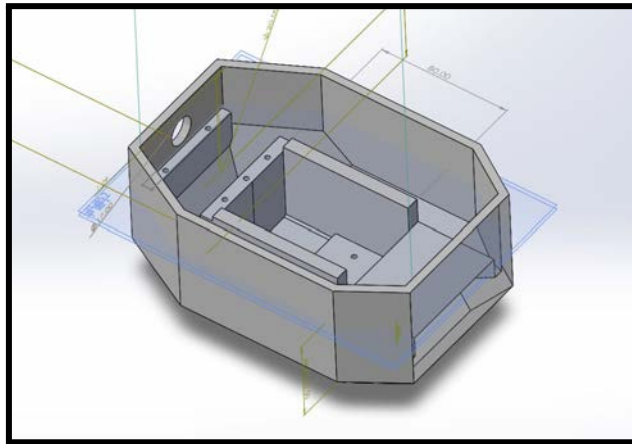


X-loader v4

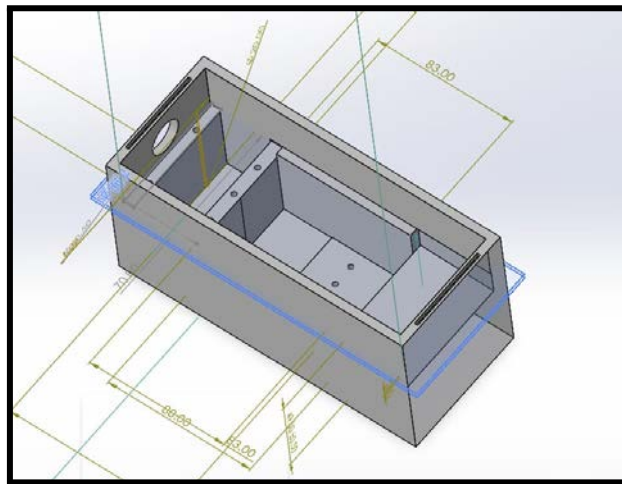


X-loader v5

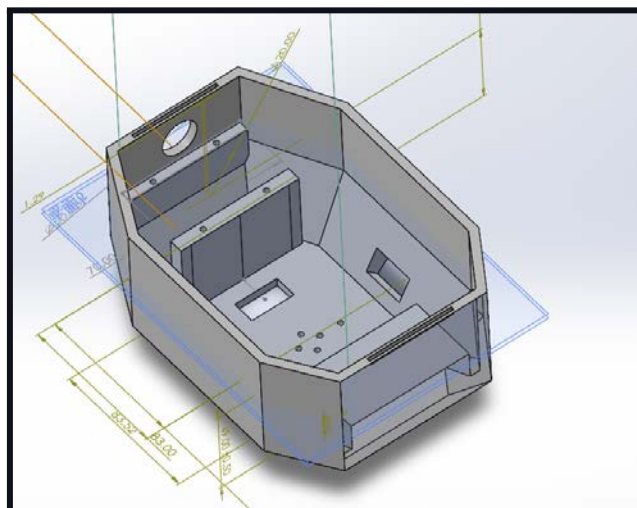
## Appendix H – Different designs of the turret for 2.2.1 Task 2



X-Turret Gen1



X-Turret Gen2



X-Turret Gen3

## Appendix I – Source code for 2.2.2 Task 2

### PS2X\_Example.ino

```
#include <PS2X_lib.h> //for v1.6
#include <Servo.h>
/*****
 * set pins connected to PS2 controller:
 * - 1e column: original
 * - 2e colmun: Stef?
 * replace pin numbers by the ones you use
 *****/

#define PS2_DAT    13 //14
#define PS2_CMD    11 //15
#define PS2_SEL    10 //16
#define PS2_CLK    12 //17

/*****
 * select modes of PS2 controller:
 * - pressures = analog reading of push-buttons
 * - rumble    = motor rumbling
 * uncomment 1 of the lines for each mode selection
 *****/

//#define pressures true
#define pressures false
//#define rumble    true
#define rumble    false

PS2X ps2x; // create PS2 Controller Class

//right now, the library does NOT support hot pluggable controllers, meaning
//you must always either restart your Arduino after you connect the controller,
//or call config_gamepad(pins) again after connecting the controller.

int error = 0;
byte type = 0;
byte vibrate = 0;
Servo downservo;
Servo upservo;
Servo bservo;
int upd=90;
int downd=90;
int bd=102;
int ledState = LOW; // ledState used to set the LED
int g =102;int g1 =135;
int r = 168;
int r1 = 3;
int b =36;
```



```

int b1 =69;
int gcheck=0;
int rcheck=0;
int bcheck=0;
int scheck=0;
void setup(){
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  bservo.attach(3);
  upservo.attach(5);
  downservo.attach(6);

  Serial.begin(57600);

  delay(300); //added delay to give wireless ps2 module some time to startup, before configuring it

  //CHANGES for v1.6 HERE!!! *****PAY ATTENTION*****

  //setup pins and settings: GamePad(clock, command, attention, data, Pressures?, Rumble?) check for error
  error = ps2x.config_gamepad(PS2_CLK, PS2_CMD, PS2_SEL, PS2_DAT, pressures, rumble);

  if(error == 0){
    Serial.print("Found Controller, configured successful ");
    Serial.print("pressures = ");
      if (pressures)
        Serial.println("true ");
      else
        Serial.println("false");
    Serial.print("rumble = ");
    if (rumble)
      Serial.println("true");
    else
      Serial.println("false");
    Serial.println("Try out all the buttons, X will vibrate the controller, faster as you press harder;");
    Serial.println("holding L1 or R1 will print out the analog stick values.");
    Serial.println("Note: Go to www.billporter.info for updates and to report bugs.");
  }
  else if(error == 1)
    Serial.println("No controller found, check wiring, see readme.txt to enable debug. visit www.billporter.info for troubleshooting tips");

  else if(error == 2)
    Serial.println("Controller found but not accepting commands. see readme.txt to enable debug. Visit www.billporter.info for troubleshooting tips");

  else if(error == 3)
    Serial.println("Controller refusing to enter Pressures mode, may not support it. ");

```

```

// Serial.print(ps2x.Analog(1), HEX);

type = ps2x.readType();
switch(type) {
  case 0:
    Serial.print("Unknown Controller type found ");
    break;
  case 1:
    Serial.print("DualShock Controller found ");
    break;
  case 2:
    Serial.print("GuitarHero Controller found ");
    break;
    case 3:
    Serial.print("Wireless Sony DualShock Controller found ");
    break;
}
}
void loop() {
  /* You must Read Gamepad to get new values and set vibration values
  ps2x.read_gamepad(small motor on/off, larger motor strenght from 0-255)
  if you don't enable the rumble, use ps2x.read_gamepad(); with no values
  You should call this at least once a second
  */

  if(error == 1) //skip loop if no controller found
    return;

  if(type == 2){ //Guitar Hero Controller
    ps2x.read_gamepad();    //read controller

    if(ps2x.ButtonPressed(GREEN_FRET))
      Serial.println("Green Fret Pressed");

    if(ps2x.ButtonPressed(RED_FRET))
      Serial.println("Red Fret Pressed");
    if(ps2x.ButtonPressed(YELLOW_FRET))
      Serial.println("Yellow Fret Pressed");
    if(ps2x.ButtonPressed(BLUE_FRET))
      Serial.println("Blue Fret Pressed");
    if(ps2x.ButtonPressed(ORANGE_FRET))
      Serial.println("Orange Fret Pressed");

    if(ps2x.ButtonPressed(STAR_POWER))
      Serial.println("Star Power Command");

    if(ps2x.Button(UP_STRUM))    //will be TRUE as long as button is pressed
      Serial.println("Up Strum");
  }
}

```

```

if(ps2x.Button(DOWN_STRUM))
  Serial.println("DOWN Strum");

if(ps2x.Button(PSB_START))    //will be TRUE as long as button is pressed
  Serial.println("Start is being held");
if(ps2x.Button(PSB_SELECT))
  Serial.println("Select is being held");

if(ps2x.Button(ORANGE_FRET)) {  // print stick value IF TRUE
  Serial.print("Wammy Bar Position:");
  Serial.println(ps2x.Analog(WHAMMY_BAR), DEC);
}
}
else { //DualShock Controller
  ps2x.read_gamepad(false, vibrate); //read controller and set large motor to spin at 'vibrate' speed

  if(ps2x.Button(PSB_START))    //will be TRUE as long as button is pressed
    Serial.println("Start is being held");
  if(ps2x.Button(PSB_SELECT))
    Serial.println("Select is being held");

  if(ps2x.Button(PSB_PAD_UP)) {  //will be TRUE as long as button is pressed
    Serial.print("Up held this hard: ");
    if(upd<180){
      upd+=5;
      upservo.write(upd);
    }

    Serial.println(ps2x.Analog(PSAB_PAD_UP), DEC);
  }
  if(ps2x.Button(PSB_PAD_RIGHT)){
    Serial.print("Right held this hard: ");
    if(downnd>30){
      downnd-=5;
      downservo.write(downnd);
    }
    Serial.println(ps2x.Analog(PSAB_PAD_RIGHT), DEC);
  }
  if(ps2x.Button(PSB_PAD_LEFT)){
    Serial.print("LEFT held this hard: ");
    if(downnd<140){
      downnd+=5;
      downservo.write(downnd);
    }
    Serial.println(ps2x.Analog(PSAB_PAD_LEFT), DEC);
  }
  if(ps2x.Button(PSB_PAD_DOWN)){
    Serial.print("DOWN held this hard: ");

```

```

        if(upd>60){
            upd-=5;
            upservo.write(upd);
        }
        Serial.println(ps2x.Analog(PSAB_PAD_DOWN), DEC);
    }

```

vibrate = ps2x.Analog(PSAB\_CROSS); //this will set the large motor vibrate speed based on how hard you press the blue (X) button

```

if (ps2x.NewButtonState()) {    //will be TRUE if any button changes state (on to off, or off to on)
    if(ps2x.Button(PSB_L3))
        Serial.println("L3 pressed");
    if(ps2x.Button(PSB_R3))
        Serial.println("R3 pressed");
    if(ps2x.Button(PSB_L2)){
        Serial.println("L2 pressed");
        digitalWrite(9, HIGH); }
    if(ps2x.Button(PSB_R2)){
        Serial.println("R2 pressed");
        if(scheck%2==0){
            digitalWrite(8, HIGH); }
        else
            digitalWrite(8, LOW);
        scheck+=1;}
    if(ps2x.Button(PSB_TRIANGLE)){
        Serial.println("Triangle pressed");
        if(gcheck%2 == 0)
            bservo.write(g);
        else
            bservo.write(g1);
        gcheck+=1;
    }
}

```

```

if(ps2x.ButtonPressed(PSB_CIRCLE)) {    //will be TRUE if button was JUST pressed
    Serial.println("Circle just pressed");
    if(rcheck%2 == 0)
        bservo.write(r);
    else
        bservo.write(r1);
    rcheck+=1;
}

```

```

if(ps2x.NewButtonState(PSB_CROSS)) {    //will be TRUE if button was JUST pressed OR released
    Serial.println("X just changed");

}

```

```
if(ps2x.ButtonReleased(PSB_SQUARE)) {           //will be TRUE if button was JUST released
  Serial.println("Square just released");
  if(bcheck%2 == 0)
    bservo.write(b);
  else
    bservo.write(b1);
  bcheck+=1; }

if(ps2x.Button(PSB_L1) || ps2x.Button(PSB_R1)) { //print stick values if either is TRUE
  Serial.print("Stick Values:");
  Serial.print(ps2x.Analog(PSS_LY), DEC); //Left stick, Y axis. Other options: LX, RY, RX
  Serial.print(",");
  Serial.print(ps2x.Analog(PSS_LX), DEC);
  Serial.print(",");
  Serial.print(ps2x.Analog(PSS_RY), DEC);
  Serial.print(",");
  Serial.println(ps2x.Analog(PSS_RX), DEC);
}
}
delay(50);
}
```

**Appendix J – Source code for 2.2.3 Task 3 and 4****yolo.py**

```

# -*- coding: utf-8 -*-
"""
Class definition of YOLO_v3 style detection model on image and video
"""

import colorsys
import os
from timeit import default_timer as timer

import numpy as np
from keras import backend as K
from keras.models import load_model
from keras.layers import Input
from PIL import Image, ImageFont, ImageDraw

from yolo3.model import yolo_eval, yolo_body, tiny_yolo_body
from yolo3.utils import letterbox_image
import os
from keras.utils import multi_gpu_model

import pyrealsense2 as rs
import cv2, math
from decimal import Decimal, ROUND_HALF_UP

class YOLO(object):
    _defaults = {
        "model_path": 'model_data/trained_data.h5',
        "anchors_path": 'model_data/yolo_anchors.txt',
        "classes_path": 'model_data/voc_classes.txt',
        "score" : 0.3,
        "iou" : 0.45,
        "model_image_size" : (480, 640),
        "gpu_num" : 1,
    }

    @classmethod
    def get_defaults(cls, n):
        if n in cls._defaults:
            return cls._defaults[n]
        else:
            return "Unrecognized attribute name '" + n + "'"

    def __init__(self, **kwargs):

```

```

self.__dict__.update(self._defaults) # set up default values
self.__dict__.update(kwargs) # and update with user overrides
self.class_names = self._get_class()
self.anchors = self._get_anchors()
self.sess = K.get_session()
self.bboxes, self.scores, self.classes = self.generate()

def _get_class(self):
    classes_path = os.path.expanduser(self.classes_path)
    with open(classes_path) as f:
        class_names = f.readlines()
    class_names = [c.strip() for c in class_names]
    return class_names

def _get_anchors(self):
    anchors_path = os.path.expanduser(self.anchors_path)
    with open(anchors_path) as f:
        anchors = f.readline()
    anchors = [float(x) for x in anchors.split(',')]
    return np.array(anchors).reshape(-1, 2)

def generate(self):
    model_path = os.path.expanduser(self.model_path)
    assert model_path.endswith('.h5'), 'Keras model or weights must be a .h5 file.'

    # Load model, or construct model and load weights.
    num_anchors = len(self.anchors)
    num_classes = len(self.class_names)
    is_tiny_version = num_anchors==6 # default setting
    try:
        self.yolo_model = load_model(model_path, compile=False)
    except:
        self.yolo_model = tiny_yolo_body(Input(shape=(None,None,3)), num_anchors//2, num_classes) \
            if is_tiny_version else yolo_body(Input(shape=(None,None,3)), num_anchors//3, num_classes)
        self.yolo_model.load_weights(self.model_path) # make sure model, anchors and classes match
    else:
        assert self.yolo_model.layers[-1].output_shape[-1] == \
            num_anchors/len(self.yolo_model.output) * (num_classes + 5), \
            'Mismatch between model and given anchor and class sizes'

    print('{} model, anchors, and classes loaded.'.format(model_path))

    # Generate colors for drawing bounding boxes.
    hsv_tuples = [(x / len(self.class_names), 1., 1.)
                   for x in range(len(self.class_names))]
    self.colors = list(map(lambda x: colorsys.hsv_to_rgb(*x), hsv_tuples))
    self.colors = list(
        map(lambda x: (int(x[0] * 255), int(x[1] * 255), int(x[2] * 255)),

```

```

        self.colors))
np.random.seed(10101) # Fixed seed for consistent colors across runs.
np.random.shuffle(self.colors) # Shuffle colors to decorrelate adjacent classes.
np.random.seed(None) # Reset seed to default.

# Generate output tensor targets for filtered bounding boxes.
self.input_image_shape = K.placeholder(shape=(2, ))
if self.gpu_num>=2:
    self.yolo_model = multi_gpu_model(self.yolo_model, gpus=self.gpu_num)
boxes, scores, classes = yolo_eval(self.yolo_model.output, self.anchors,
    len(self.class_names), self.input_image_shape,
    score_threshold=self.score, iou_threshold=self.iou)
return boxes, scores, classes

def detect_image(self, image):
    start = timer()

    if self.model_image_size != (None, None):
        assert self.model_image_size[0]%32 == 0, 'Multiples of 32 required'
        assert self.model_image_size[1]%32 == 0, 'Multiples of 32 required'
        boxed_image = letterbox_image(image, tuple(reversed(self.model_image_size)))
    else:
        new_image_size = (image.width - (image.width % 32),
            image.height - (image.height % 32))
        boxed_image = letterbox_image(image, new_image_size)
    image_data = np.array(boxed_image, dtype='float32')

    print(image_data.shape)
    image_data /= 255.
    image_data = np.expand_dims(image_data, 0) # Add batch dimension.

    out_boxes, out_scores, out_classes = self.sess.run(
        [self.boxes, self.scores, self.classes],
        feed_dict={
            self.yolo_model.input: image_data,
            self.input_image_shape: [image.size[1], image.size[0]],
            K.learning_phase(): 0
        })

    print('Found {} boxes for {}'.format(len(out_boxes), 'img'))

    font = ImageFont.truetype(font='font/FiraMono-Medium.otf',
        size=np.floor(3e-2 * image.size[1] + 0.5).astype('int32'))
    thickness = (image.size[0] + image.size[1]) // 300

    center_coordinates_array = []
    for i, c in reversed(list(enumerate(out_classes))):
        predicted_class = self.class_names[c]

```



```

box = out_boxes[i]
score = out_scores[i]

label = '{} {:.2f}'.format(predicted_class, score)
draw = ImageDraw.Draw(image)
label_size = draw.textsize(label, font)

top, left, bottom, right = box
top = max(0, np.floor(top + 0.5).astype('int32'))
left = max(0, np.floor(left + 0.5).astype('int32'))
bottom = min(image.size[1], np.floor(bottom + 0.5).astype('int32'))
right = min(image.size[0], np.floor(right + 0.5).astype('int32'))

x_center = (left + right) / 2
y_center = (top + bottom) / 2
center_coordinates_array.append([x_center, y_center])

print(label, (left, top), (right, bottom))

if top - label_size[1] >= 0:
    text_origin = np.array([left, top - label_size[1]])
else:
    text_origin = np.array([left, top + 1])

# My kingdom for a good redistributable image drawing library.
for i in range(thickness):
    draw.rectangle([left + i, top + i, right - i, bottom - i], outline=self.colors[c])
draw.rectangle(
    [tuple(text_origin), tuple(text_origin + label_size)], fill=self.colors[c])
draw.text(text_origin, label, fill=(0, 0, 0), font=font)
del draw

end = timer()
print(end - start)
return image, center_coordinates_array

def close_session(self):
    self.sess.close()

def detect_video(yolo):

    center_coordinates_array = []
    theta = 0
    font = cv2.FONT_HERSHEY_SIMPLEX

    config = rs.config()
    config.enable_stream(rs.stream.color, 640, 480, rs.format.bgr8, 30)
    config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)

```

```

pipeline = rs.pipeline()
profile = pipeline.start(config)

align_to = rs.stream.color
align = rs.align(align_to)

# get camera intrinsics
intr = profile.get_stream(rs.stream.color).as_video_stream_profile().get_intrinsics()

accum_time = 0
curr_fps = 0
fps = "FPS: ??"
prev_time = timer()
while True:

    frames = pipeline.wait_for_frames()

    aligned_frames = align.process(frames)
    color_frame = aligned_frames.get_color_frame()
    depth_frame = aligned_frames.get_depth_frame()
    if not depth_frame or not color_frame:
        continue

    color_image = np.asanyarray(color_frame.get_data())
    #depth_image = np.asanyarray(depth_frame.get_data())

    image = Image.fromarray(color_image)
    image, center_coordinates_array = yolo.detect_image(image)
    result = np.asarray(image)

    if(len(center_coordinates_array) > 0):
        for i in range(len(center_coordinates_array)):
            dist = depth_frame.get_distance(int(center_coordinates_array[i][0]),
int(center_coordinates_array[i][1]))*1000 #convert to mm

            #calculating coordinates of the indentified target
            Xtemp = dist*(center_coordinates_array[i][0] -intr.ppx)/intr.fx
            Ytemp = dist*(center_coordinates_array[i][1] -intr.ppy)/intr.fy
            Ztemp = dist

            Xtarget = Xtemp - 35 #35 is RGB camera module offset from the center of camera
            Ytarget = -(Ztemp*math.sin(theta) + Ytemp*math.cos(theta))
            Ztarget = Ztemp*math.cos(theta) + Ytemp*math.sin(theta)

            coordinates_text = "(" + str(Decimal(str(Xtarget)).quantize(Decimal('0'), rounding=ROUND_HALF_UP)) + \
                ", " + str(Decimal(str(Ytarget)).quantize(Decimal('0'), rounding=ROUND_HALF_UP)) + \
                ", " + str(Decimal(str(Ztarget)).quantize(Decimal('0'), rounding=ROUND_HALF_UP)) + ")"

```

```

        print("x, y : " + str(int(center_coordinates_array[i][0])))
        cv2.putText(result, text=coordinates_text, org=(int(center_coordinates_array[i][0])-160,
int(center_coordinates_array[i][1])),
                    fontFace=font, fontScale = 1, color=(255,255,255), thickness = 2, lineType=cv2.LINE_AA)

    curr_time = timer()
    exec_time = curr_time - prev_time
    prev_time = curr_time
    accum_time = accum_time + exec_time
    curr_fps = curr_fps + 1
    if accum_time > 1:
        accum_time = accum_time - 1
        fps = "FPS: " + str(curr_fps)
        curr_fps = 0

    #depth_colormap = cv2.applyColorMap(cv2.convertScaleAbs(depth_image, alpha=0.08), cv2.COLORMAP_JET)

    cv2.namedWindow("result", cv2.WINDOW_NORMAL)
    cv2.imshow("result", result)
    #cv2.imshow("result_depth", depth_colormap)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    yolo.close_session()

if __name__ == '__main__':
    detect_video(YOLO())

```

## Appendix K – Source code for 2.2.4 Task 3

**realsense\_color\_config.yaml**

```
%YAML:1.0
```

```
#common parameters
```

```
imu_topic: "/camera/imu"
```

```
image_topic: "/camera/color/image_raw"
```

```
output_path: "/home/tony-ws1/output/"
```

```
#camera calibration
```

```
model_type: PINHOLE
```

```
camera_name: camera
```

```
image_width: 640
```

```
image_height: 480
```

```
distortion_parameters:
```

```
  k1: 0
```

```
  k2: 0
```

```
  p1: 0
```

```
  p2: 0
```

```
projection_parameters:
```

```
  fx: 609.705641276698035
```

```
  fy: 609.027567462137247
```

```
  cx: 318.145635222272752
```

```
  cy: 236.045676043567377
```

```
# Extrinsic parameter between IMU and Camera.
```

```
estimate_extrinsic: 2 # 0 Have an accurate extrinsic parameters. We will trust the following imu^R_cam,  
imu^T_cam, don't change it.
```

```
    # 1 Have an initial guess about extrinsic parameters. We will optimize around your initial guess.
```

```
    # 2 Don't know anything about extrinsic parameters. You don't need to give R,T. We will try to  
calibrate it. Do some rotation movement at beginning.
```

```
#If you choose 0 or 1, you should write down the following matrix.
```

```
#Rotation from camera frame to imu frame, imu^R_cam
```

```
extrinsicRotation: !!opencv-matrix
```

```
  rows: 3
```

```
  cols: 3
```

```
  dt: d
```

```
  data: [ 0.99964621, 0.01105994, 0.02418954,  
         -0.01088975, 0.9999151, -0.00715601,  
         -0.02426663, 0.00689006, 0.99968178]
```

```
#Translation from camera frame to imu frame, imu^T_cam
```

```
extrinsicTranslation: !!opencv-matrix
```

```

rows: 3
cols: 1
dt: d
data: [0.07494282, -0.01077138, -0.00641822]

#feature traker paprameters
max_cnt: 150      # max feature number in feature tracking
min_dist: 25      # min distance between two features
freq: 10          # frequence (Hz) of publish tracking result. At least 10Hz for good estimation. If set 0, the
frequency will be same as raw image
F_threshold: 1.0  # ransac threshold (pixel)
show_track: 1     # publish tracking image as topic
equalize: 0       # if image is too dark or light, trun on equalize to find enough features
fisheye: 0        # if using fisheye, trun on it. A circle mask will be loaded to remove edge noisy points

#optimization parameters
max_solver_time: 0.04 # max solver itrations time (ms), to guarantee real time
max_num_itations: 8   # max solver itrations, to guarantee real time
keyframe_parallax: 10.0 # keyframe selection threshold (pixel)

#imu parameters      The more accurate parameters you provide, the better performance
acc_n: 0.2           # accelerometer measurement noise standard deviation. #0.2
gyr_n: 0.05          # gyroscope measurement noise standard deviation.   #0.05
acc_w: 0.02          # accelerometer bias random work noise standard deviation. #0.02
gyr_w: 4.0e-5        # gyroscope bias random work noise standard deviation.  #4.0e-5
g_norm: 9.80         # gravity magnitude

#loop closure parameters
loop_closure: 1       # start loop closure
fast_relocalization: 1 # useful in real-time and large project
load_previous_pose_graph: 0 # load and reuse previous pose graph; load from 'pose_graph_save_path'
pose_graph_save_path: "/home/tony-ws1/output/pose_graph/" # save and load path

#unsynchronization parameters
estimate_td: 0        # online estimate time offset between camera and imu
td: 0.000             # initial value of time offset. unit: s. readed image clock + td = real image clock (IMU clock)

#rolling shutter parameters
rolling_shutter: 0     # 0: global shutter camera, 1: rolling shutter camera
rolling_shutter_tr: 0  # unit: s. rolling shutter read out time per frame (from data sheet).

#visualization parameters

```

save\_image: 1           # save image in pose graph for visualization prupose; you can close this function by  
setting 0  
visualize\_imu\_forward: 0   # output imu forward propogation to achieve low latency and high frequence results  
visualize\_camera\_size: 0.4   # size of camera marker in RVIZ