DEVELOPMENT OF WATER SURFACE ROBOT SYSTEM FOR LAKE SANITATION AND SAMPLING

AHMED ABDULLAH OMAR

FACULTY OF COMPUTER SCIENCE AND
INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

DEVELOPMENT OF WATER SURFACE ROBOT SYSTEM FOR LAKE SANITATION AND SAMPLING

AHMED ABDULLAH OMAR

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of the candidate: **Ahmed Abdullah Omar**

Registration/Matric No: **WOA180012**

Name of the Degree: **Master of Computer Science**

Title of Dissertation: **Development of Water Surface Robot System for Lake Sanitation and Sampling**

Field of Study: **Robotics**

I do solemnly and sincerely declare that:

(1) I am the sole author /writer of this work;

(2) This work is original;

(3) Any use of any work in which copyright exists was done by way of fair dealings and any expert or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the work and its authorship has been acknowledged in this work;

(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;

(5) I hereby assign all and every rights in the copyrights to this work to the Universiti Malaya (UM), who henceforth shall be owner of the copyright in this work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained actual knowledge;

(6) I am fully aware that if in the course of making this work I have infringed any copyright whether internationally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature: Date: 01.02.2022

Subscribed and solemnly declared before,

Witness Signature: Date: 01.02.2022
Name:
Designation:

DEVELOPMENT OF WATER SURFACE ROBOT SYSTEM FOR LAKE SANITATION AND SAMPLING

ABSTRACT

There are many exciting individual robot designs and strategies in either water sanitation or sampling tasks. However, rarely a robot design considers handling both tasks because they do not share similar sensing technology or similar data processing framework. Also, low-cost robot designs have become limited to onboard computational resources; sensor data is captured locally, and the mechanical aspects of the robots are mainly pre-programmed or remotely controlled. In low-cost robots, automation is hardly ever considered. This dissertation explores a design where a low-cost water surface robot can tackle sanitation and sampling tasks. Central to the design is real-time access to mechanical control and sensor data for off-board processing and analytics. Off-board computation allows other researchers to deploy advanced algorithms without restrictions on computing resources. Only the results need to be communicated back onto the robot for real-time actions. The dissertation proposes three objectives; to design and develop the water surface robot structure, control and electronics, design and develop the APIs for remote robot connection, data communication and positioning, and evaluate the robot's performance through case studies. Regarding results, a low-cost robot system that can glide on the lake surface has been developed. The robot system showcases PVC pipes for the body with a load up to 17kg. The electronic design covers the cooling system, battery level monitoring, voltage control and sensor reading. The software design features a mobile app for remote access, live video streaming, mapping system and communication APIs. The robot networking includes an RTK server setup for accurate localization and positioning. Two case studies test the robotic system performance. Case study 1 tests the robot's APIs in implementing a genetic algorithm for wayfinding with RTK-based localization. Case study 2 tests the robot's APIs in training a deep learning model for

garbage detection based on the robot's live stream. The dissertation shows that an open robotic design can coordinate contributions by various parties on a single platform; robotics engineers can focus on improving the robot hardware and controls towards enabling live data. AI researchers can test automation theories on robot data without the need for robot hardware and control know-how. Such design allows anyone to drive the robot over the internet without the physical need to be on site. Remote driving can improve lake sanitation and sampling activities and can significantly promote citizen science.

Keywords: Water surface robot, robot control system, water sanitation, water sampling, intelligent system

PEMBANGUNAN SISTEM ROBOT PERMUKAAN AIR UNTUK PEMBERSIHAN DAN PENSAMPELAN TASIK

ABSTRAK

Terdapat banyak reka bentuk dan strategi robot individu yang menarik baik dalam pembersihan air atau tugas pensampelan. Namun, jarang sekali reka bentuk robot mempertimbangkan untuk menangani kedua-dua tugas tersebut kerana mereka tidak berkongsi teknologi penginderaan yang serupa atau kerangka pemprosesan data yang serupa. Reka bentuk robot kos rendah juga terhad kepada sumber pengaturcaraan atas papan; data sensor dirakam secara tempatan, dan aspek mekanik utama robot diprogramkan atau dikawal dari jauh. Dalam robot kos rendah, automasi hampir tidak pernah dipertimbangkan. Disertasi ini meneroka reka bentuk di mana robot permukaan air kos rendah dapat menangani tugas-tugas kebersihan dan pengambilan sampel. Asas reka bentuk termasuk akses masa nyata ke data kawalan mekanik dan sensor untuk pemprosesan dan analisis pengaturcaraan di luar papan. Pengaturcaraan di luar papan membolehkan penyelidik lain menggunakan algoritma canggih tanpa sekatan pada sumber pengkomputeran, dan hanya hasilnya disampaikan kembali ke robot untuk tindakan masa nyata. Disertasi ini mencadangkan tiga objektif; merancang dan mengembangkan struktur robot permukaan air, kawalan dan elektronik, untuk merancang dan mengembangkan API untuk sambungan robot jarak jauh, komunikasi data dan kedudukan, dan untuk menilai prestasi robot melalui kajian kes. Mengenai hasilnya, satu sistem robot kos rendah yang dapat meluncur di permukaan tasik telah dibangunkan. Sistem robot mempamerkan paip PVC untuk badan dengan muatan hingga 17kg. Reka bentuk elektronik merangkumi sistem penyejukan, pemantauan tahap bateri, kawalan voltan dan bacaan sensor. Reka bentuk perisiannya mempunyai aplikasi mudah alih untuk akses jarak jauh, streaming video langsung, sistem pemetaan dan API komunikasi. Rangkaian robot merangkumi penyediaan pelayan RTK untuk penyetempatan dan

kedudukan yang tepat. Dua kajian kes menguji prestasi sistem robotik. Kajian kes pertama menguji API robot dalam menerapkan algoritma genetik untuk mencari jalan dengan penyetempatan berdasarkan RTK. Kajian kes kedua menguji API robot dalam melatih model pembelajaran mendalam untuk pengesanan sampah berdasarkan aliran langsung robot. Disertasi menunjukkan reka bentuk robot terbuka dapat menyelaraskan sumbangan oleh pelbagai pihak dalam satu platform; jurutera robotik dapat memberi tumpuan untuk meningkatkan perkakasan dan kawalan robot untuk mengaktifkan data langsung, dan para penyelidik AI dapat menguji teori automasi pada data robot tanpa memerlukan pengetahuan kawalan dan perkakasan robot. Reka bentuk sedemikian juga bermaksud sesiapa sahaja yang ingin menggerakkan robot boleh melakukannya melalui internet tanpa keperluan fizikal di lokasi. Ini boleh mengganti dan menggugat teknologi sedia ada untuk aktiviti sanitasi dan pengambilan sampel tasik, dan dapat menarik perhatian komuniti terhadap sains persekitaran.

Kata Kunci: Robot permukaan air, sistem kawalan robot, sanitasi air, pensampelan air, sistem pintar

ACKNOWLEDGEMENTS

For writing this dissertation, I have received tremendous support and assistance.

First, I would like to thank my supervisor, Dr Zati Hakim Azizul Hasan, for guiding each step of the process. I acknowledge her for inspiring my interest in developing a water conservation system. Her outstanding support and funding while building the water conservation robot helped me to finish this dissertation successfully. Few lines are not enough to thank Dr Zati. She also played a significant role while testing the robot. Even though Malaysia was under lockdown during the covid19 pandemic, she did her best and led this project to success.

Secondly, I express my gratitude to Aliff Danial and Aqiff Mursyideen, Dr Zati's students at the Department of Artificial Intelligence. The amount of support they provided during the covid19 outbreak was the key to success. Their tremendous help for testing and evaluating the robot via the case studies helped me complete this dissertation. Another person I would like to thank is Affan Nasaruddin of UM Water Warriors. He provided me with excellent pieces of information about lake pollution.

This journey would not have been possible without the support of my family and professors. To my family, thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams. I am incredibly grateful to my parents, who supported me emotionally and financially. I always knew that you believed in me and wanted the best for me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Background

Water is an essential substance on earth. Without water, evolution and life would not have been possible. The earth's surface is covered by 3% freshwater, and the rest, 97% of the water, is made of saltwater. The water on the earth's surface, such as the lakes, rivers, ponds, and swamps, creates only 0.3% of the world's freshwater. The world's fresh water supply is not as abundant as saltwater, but it is a renewable resource following the hydrological cycle and proper urban management (Barry, 2019).

In urban development, lakes have become an essential variant of freshwater ecosystems. The lake's most significant function is to recharge and maintain the groundwater table. The lakes have great recreational potential in water sports such as kayaking and boat racing in progressive cities. The lakes are also one of the primary water sources in fire emergencies in most cities. Aquaculture relies on lakes, and its increasing popularity can impact urban income generation (Isa et al., 2020). The lakes also add aesthetic features to one's city, attracting tourism.

Awareness is rising globally of the enormous amount of ocean plastic pollution (Avio et al., 2017). On the contrary, lesser people realise that a similar concentration of plastic pollution accumulates in the lakes. Aside from plastic, foam scraps and tree leaves (common among tropical countries) are common pollutants to freshwater reservoirs. There are also occasions where bodies of dead animals are found in the lakes. Interestingly, unlike the ocean, there are no sizeable floating garbage patches found in the lakes. Strong wind and lack of ocean-like current at the lakes have been observed to break up the garbage from gathering into large patches (Cable et al., 2017). With no determining pattern on the garbage accumulation location at the lakes, the garbage can be found near

the shore and floating anywhere else. Figure 1.1 shows a lake in Petaling Jaya showing contamination and garbage accumulation.

According to a report released by NAHRIM (2009), domestic and industrial wastes have led 60% of the 90 lakes studied in Malaysia to be nutrient-rich. When the lakes are nutrient-rich or eutrophic, they will experience algal blooms. Eutrophication is especially common in urban lakes, which affect human use of the water for recreation and aesthetic values, while other lakes can face infestation problems (Sharip & Suratman, 2017; van Beusekom, 2018). It has also been reported that lake eutrophication will increase in the 21st century due to climate change (Sinha et al., 2017).

Figure 1.1: Lake at Taman Jaya Park, Petaling Jaya, taken in June 2020

Authorities collect samples of freshwater to determine the lakes' eutrophication level. Lake sampling is fieldwork associated with the collection and transport of water samples. A sampling expedition may include in-situ analysis and coordination with the laboratory for analysis. For example, temperature, dissolved oxygen (DO), conductivity, and pH are measured in-situ as field parameters while other physical, chemical, and bacteriological parameters are analysed in the laboratory (Al-Badaii et al., 2013). Traditionally, a successful sampling expedition includes a comprehensive checklist (Balance & Bartram, 2002), usually involving water, geographical, and safety personnel.

Lake sampling considers a good selection of sampling locations and methods to ensure minimal bias. Depending on the objective of sampling, strategically, sampling location may include (a) surface dip sample taken at the deepest point of the lake, (b) surface dip sample taken at the edge of the lake, (c) surface dip sample about 30m from the shore and (d) surface dip sample taken along a short transect out from the shore. For big-scale sampling, usually samples taken from the edge of the lake are the most cost-effective. However, it has been observed that multiple sampling locations (or stations) per lake are required for proper profiling of the lake's eutrophication status at a given time. Figure 1.2 shows the 7 locations or stations selected for water sampling at Cempaka Lake in Bangi, Malaysia (Gasim et al., 2015).

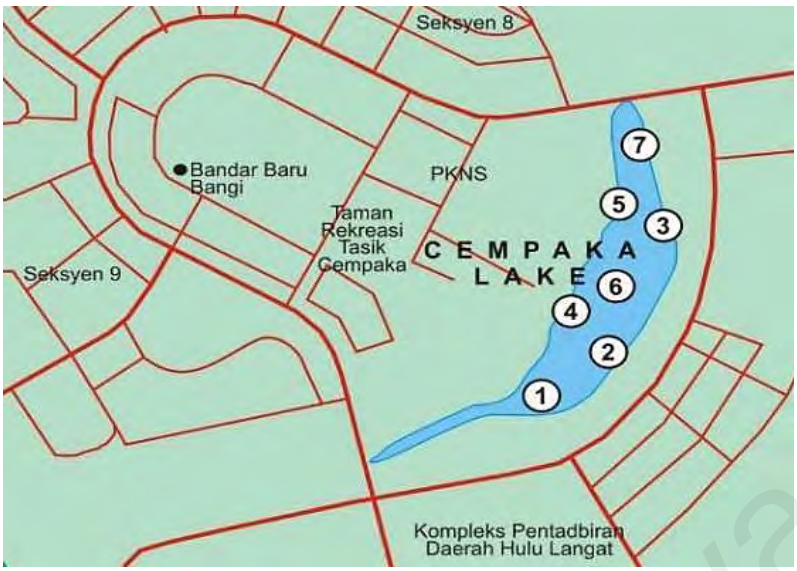

Figure 1.2: Sampling stations at the Cempaka Lake (Gasim et al., 2015)

Figure 1.3: Grab sampling method and the EXO sonde

According to the Surface Water Sampling (2013) guidelines, the sampling method may introduce bias. Sampling can often be done by wading on a small boat to access stream banks, piers, or other low platforms. However, it is observed that movement caused by the wading stirs the shallow water, and re-suspension of the lake's bottom deposits may bias the freshwater sample. There are occasions where the stream is too deep to wade, or the sampling involves more than one water depth. Sampling is usually proposed from an elevated platform, e.g., bridges and high piers, and boats with an engine. Additional sampling gears are sometimes required, and they are usually costly.

Figure 1.3 shows the grab sampling method, which was traditionally practised before sonde technology became available. The sonde is an electronic multi-parameter probe that auto profiles water properties upon contact with water samples. Whenever the sonde is not available, an in-situ test can still be done manually, for example, the litmus paper test for pH, thermometer for temperature, and others. Currently, lab testing, even though time-consuming, is still a popular approach since some parameters, like ammonium nitrate and phosphorus, are more cost-effective when tested at the lab.

1.2 Motivation

Water sanitation and sampling often involve hauling equipment, boats and people to sites. Applying robotics technologies to serve environmental-related tasks has gained interest in recent times. Some popular applications include thermal infrared imaging of geothermal environments (Nishar et al., 2016), coastal surveying for topographic mapping and measurement (Turner et al., 2016) and marine wildlife monitoring (Linchant et al., 2015). Robots are attractive for remote sensing applications due to their capacity carrying sensors, ability to rapidly and remotely travel to locations that are difficult to access and the ability to monitor large areas in a short period (Beloev, 2016).

Applying robotics technology for lake water sanitation and sampling is advantageous because of the modularity and customisable aspects. The first aspect is the frame design and analysis. Lake-bound robots require balancing structure properties like load distribution, displacement, maximum strain, and stress. The materials used, total mass and weight of motors also influence the structural properties to maintain buoyancy and withstand current at the lake. The second aspect is dynamic analysis. The robot design must consider mass and acceleration, propeller thrust, drag force against oncoming flow velocity and disturbances for acceleration estimation.

The third aspect is the electronic system. Off the shelf boards and available open market sources are preferred to ensure robot stability. Central to the electronic system is increasing navigating time and payload while consuming the least power. Power consumption depends significantly on how much thrust is generated due to the motor-propeller combination. A brushless DC motor, propeller, and electronic speed controller (ESC) are some electronic components to consider. ESC is an electrical circuit used to vary the speed of the motor. The speed produced depends on the received frequency from the controller board. The burst current of each ESC is usually 30amp. The controller usually includes integrated gyroscopic sensors and an accelerometer. The remote controller for radio control systems is usually the 2.4 GHz signal technology with 100m range coverage. Lastly is the battery. Most water-bound robot designs opted for a rechargeable sealed lead-acid battery.

Robotics technology does not stop at hardware. Recent development in software control promotes cheaper and smarter functioning robot. Roboticists focusing on water sanitation propose scoping waste design pick up (Dahlan et al., 2019), and others use a conveyor belt (Ruangpayoongsak et al., 2017). Jayawant & Sakpal (2018) propose a skimmer for garbage collection, detecting garbage as obstacles using ultrasonic sensors. There is also a growing interest in identifying various types of garbage such as cans, leaves, small branches, floating debris, plastic bottles, and other floating objects through image processing and the convolutional neural network (CNN) architectures (Kong et al., 2019; Kong et al., 2020; Steccanella et al., 2020).

Roboticists focusing on water sampling show how a water-bound robot can traverse from one sampling station to another following a pre-programmed path (Dunbabin et al., 2009; Prempraneerach & Kulvanit, 2010; Valada et al., 2014). They used a human operator to

select the connecting route manually, so all sampling stations were visited. Performing automated wayfinding using advanced techniques such as the divide and search algorithms commonly used in a travelling salesmen problem (Madani, 2019) and accurate self-localisation (Riaz et al., 2010) can determine robot's shortest path during sampling. These algorithms need sound positioning systems to get location updates.

In summary, robot technologies are modular and scalable to various tasks offering an innovative solution for water sanitation and sampling challenges. For example, combining an accurate positioning system and a searching algorithm can generate waypoints so the robot can visit sampling stations with less human intervention. Integrating the CNN pipeline with a live visual stream can support automated garbage detection and tracking.

1.3 Problem statement

There are many exciting individual robot designs and strategies in water sanitation or sampling. However, rarely a mechanical design considers handling both the water sanitation and sampling tasks because they do not share similar sensing technology or similar data processing framework. Also, low-cost robot designs have become limited to onboard computational resources since sensors data is usually captured locally. The mechanical aspects of the robots are mainly pre-programmed or remotely controlled. In low-cost robots, automation is hardly ever considered. This study proposes a low-cost water surface robot that can tackle sanitation and sampling tasks. Central to the design is real-time access to mechanical control and sensors data for off-board processing and analytics. Off-board computation allows other researchers to deploy advanced algorithms without restrictions on computing resources. At the same time, only the results need to be communicated back onto the robot for real-time actions.

1.4 Objectives of the study

The following objectives are proposed for the study:

1. To design and develop the water surface robot structure, control and electronics.
2. To design and develop the APIs for remote robot connection, data communication and positioning.
3. To evaluate the robot's wayfinding and garbage detection performances.

1.5 Scopes of the study

The following scopes are proposed to manage some pre-requisites and constraints:

1. Visual perception is done through a mono-vision camera, a popular sensor on a low-cost robotic platform. The mono-vision camera cannot provide depth information.
2. The global positioning system (GPS) has an error rate of estimated ± 3 meters for static bodies, or more, if a body is moving, which is useless for the robot proposed. This study will explore other positioning systems such as the Real-time Kinematic (RTK) for enhanced outdoor positioning accuracy.
3. The robot must collect garbage smaller than the robot size, such as plastic cups, foam scraps, and tree leaves. It is worth noting that the robot will not have the capacity to clean algae pollution.
4. For efficient load, balancing, and power distribution, a maximum of 500~600ml of water is estimated at each water sampling activity.
5. This study does not include a water quality sensor as part of the design.
6. All field experiments will be conducted at the Varsity Lake, Universiti Malaya.

1.6 Significance/Impact of the Study

As Malaysia experiences rapid urbanization and population growth, water pollution levels have also increased, severely affecting our natural and human-made lakes' water quality. A collaborative effort can speed up water sanitation and sampling innovation. Robotic technologies are modular and scalable for solving real problems. However, onboard computational processing limits low-cost robot designs in searching and tracking tasks. Additionally, automation is hardly ever discussed in low-cost robots. An open design where the robot states and data are published elsewhere can load off the onboard processing (Azizul, 2019). An open robotic design can significantly coordinate contributions by various parties on a single platform; robotics engineers can improve the robot hardware and controls towards enabling live data. AI researchers can test automation theories on robot data without robot hardware and control know-how. Such design is attractive to anyone wishing to drive the robot over the internet without being physically present. The open robotic design can improve lake sanitation and sampling activities and significantly promote citizen science.

1.7 Dissertation organization

This dissertation is divided into five chapters. Chapter 1 consists of background, motivation, and mapping objectives, scopes, and study outcomes. Chapter 2 reviews the literature relevant to this work. The review includes water sanitation challenges, robot structure designs, garbage collecting mechanisms, robot electronics and communications, external devices and navigation, and water sampling technologies. Chapter 3 describes the research methodology of this work, focusing on the mechanical system design and development comprehensively covering structure, control and communication considerations. In Chapter 4, the performance of the robotic system is analyzed through two case studies. Chapter 5 concludes the dissertation and discusses future works.

CHAPTER 2: LITERATURE REVIEW

2.1 Overview

Garbage pollution and eutrophication of inland water are prevalent issues in Malaysia, threatening lake and reservoir ecosystems. Innovative water sanitation and sampling methods are ways forward in lake conservation to address these widespread challenges. The application of water surface robots to extract garbage and water samples has gained researchers' interest in recent times. This chapter reviews related work on the design and development of water-bound robotic system structure, control and communication specific or lake conservation purposes in recent years. Works like Manjanna et al. (2018) that did not design and develop their robot are not within this study scope.

2.2 Remote control (RC) robots for water sanitation

Researchers have focused on developing remote control robots for water sanitation (Blue Growth, 2018; GREENBAY, 2018; IADYS, 2019; Rahmawati et al., 2019; Ruangpayoongsak et al., 2017; Sumroengrit & Ruangpayoongsak, 2017; Thakur et al., 2019). They define water sanitation as removing garbage and litter from the water surface. The garbage and litter include discarded items like plastic and foam containers, tires, fast-food wrappers, and nonorganic construction debris. These items are unsightly and a sign of human neglect or ignorance for aesthetic values and aquatic ecosystems.

Researchers also consider designing and developing garbage cleaning robots only for lake sanitation (Agrawal & Bhattacharya, 2013; Rajavel & Shantosh, 2019; Su et al., 2009; Wang & Liu, 2010). Others prefer to focus on the river (Abrams, 2018; Dahlan et al., 2019; Sinha et al., 2014; Soumya & Preeti, 2018). Due to the differing environmental conditions, the robot design for lakes is often different from rivers.

Organic wastes can interfere with aquatic plants' natural ecosystem, affecting the reproductive behaviour of fish and other animals and depleting water of dissolved oxygen as the wastes decompose. Therefore, the water sanitation community also considers organic waste and algae a threat. A group of researchers developed a robotic system called ARROS to combat algal bloom (Jung et al., 2017). Figure 2.1 shows ARROS with a maximum weight of 8kg and a length between 50cm and 150cm. ARROS can be considered a mid-size robot.

Figure 2.1: Algal bloom removal robotic system or ARROS (Jung et al. 2017)

2.3 Structural design and material for RC water surface robot

Most of the proposed water surface robot takes a catamaran shape. The catamaran shape has two parallel hulls of equal size. Having the same base of two similar pipes is common (Abrams, 2018; Agrawal & Bhattacharya, 2013; Dahlan et al., 2019; Rafique & Langde, 2017; Rahmawati et al., 2019; Thakur et al., 2019). The materials of the pipes have various options. The steel pipes are chosen by Rafique & Langde (2017), whereas others selected PVC pipes (Agrawal & Bhattacharya, 2013; Dahlan et al., 2019; Rahmawati et al., 2019; Thakur et al., 2019). Figure 2.2 shows two pipes making the RC water surface robot base in Thakur et al. (2019).

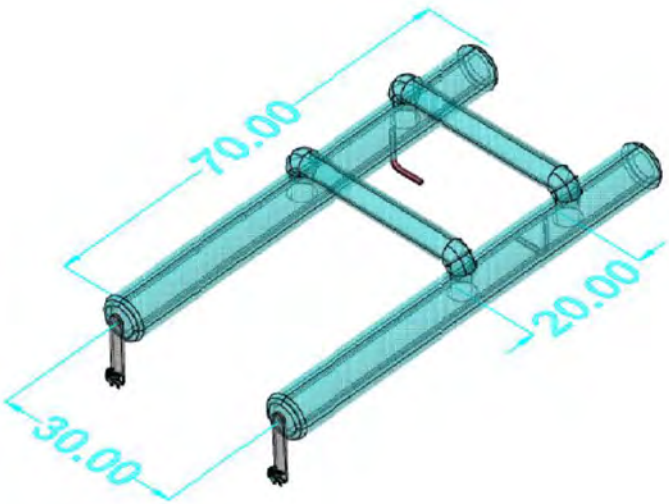

Figure 2.2: Two pipes as based for the RC robot structure (Thakur et al., 2019)

Capturing garbage and trapping them using a net is proposed by many robot developers. Dahlan et al. (2019) prefer nylon net because it is cheaper and easy to replace, while Abrams (2018) opted for the metal net. Figure 2.3 and Figure 2.4 show the net placement on their robot designs, respectively. Additionally, adding a conveyor belt is also considered to facilitate garbage capturing. Figure 2.5 shows Agrawal & Bhattacharya (2013) robot design with a compact looking conveyor belt at the robot's front. Figure 2.6 shows a larger conveyor belt installation placed after the mechanical arm designed to sweep garbage (Rafique & Langde, 2017).

Garbage will go through the conveyor belt and into a small net as the garbage container. The container usually has an open box shape, and it is mounted on the backside of the robot. Both projects with the conveyor belt described having issues with the size of the container, which is small compared to the robot (Agrawal & Bhattacharya, 2013; Rafique & Langde, 2017).

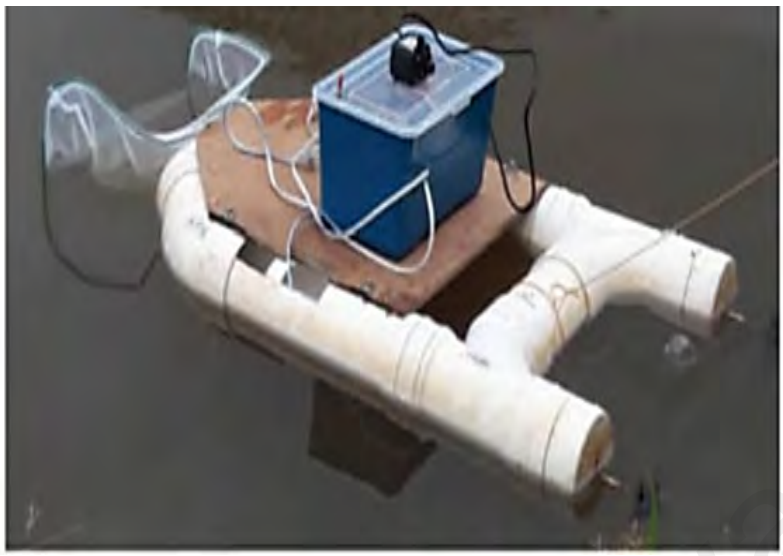

Figure 2.3: Waste Hunter Robot (Dahlan et al., 2019)

Figure 2.4: Garbage-cleaning robot (Abrams, 2018)

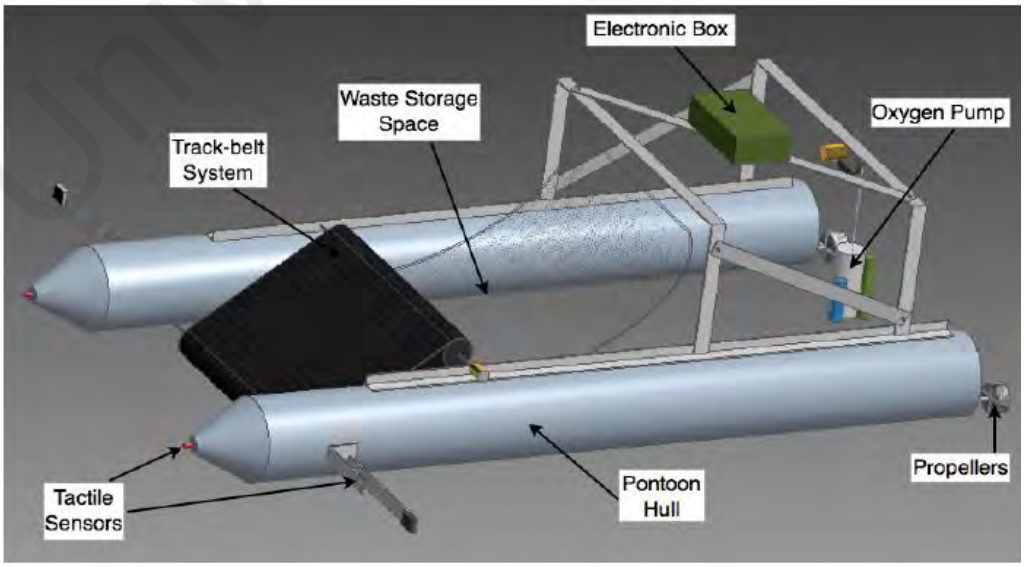

Figure 2.5: Lake cleaning robot (Agrawal & Bhattacharya, 2013)

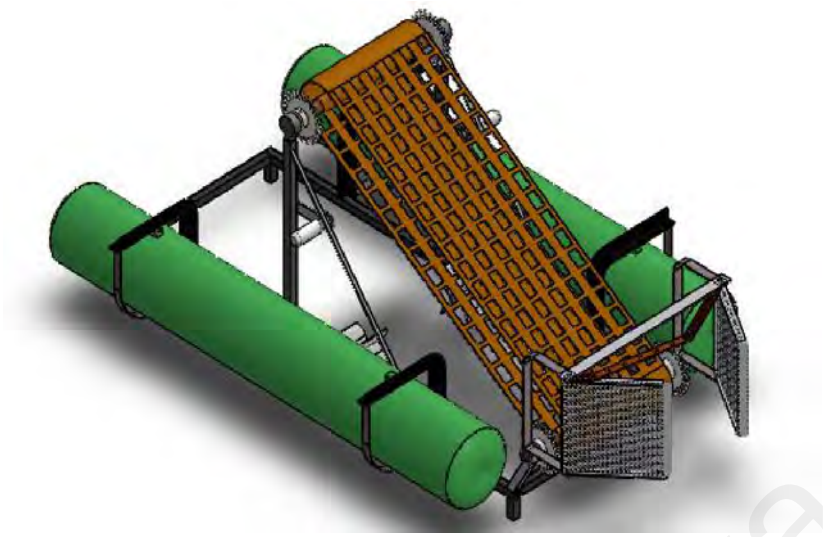

Figure 2.6: River cleaning machine (Rafique & Langde, 2017)

The Waste Hunter teleoperated robot by Dahlan et al. (2019) has an additional function for water purification. The robot system utilises two-stage filtering units. The first stage is a sediment filter, which filters the contamination that the human eyes can see. The second stage is a carbon filter that removes heavy iron contained inside. The Waste Hunter uses two pipes in a U-letter shape connected at the front side. A box that contains electronic components is located on top of the robot. A shovel-like waste container follows garbage and collects them. Meanwhile, two motors are fixed inside the hulls at the back to drive the propellers and create enough force so the robot can move.

In Rahmawati et al. (2019), the robot has a motor-driven arm that closes and opens the garbage container. The arm collects the garbage and pushes them inside the box (see Figure 2.7). The container is an aluminium case covered with a net placed on the robot. Although installing a much bigger container can support collecting a significant amount of garbage, Rahmawati et al. (2019) reported that such design leads to a payload issue since the weight of garbage at total capacity is heavy. Placing two pontoons symmetric along the robot's moving direction and ensuring the centre of the floating force is located at the symmetric axis focuses weights at the centre of mass of the robot.

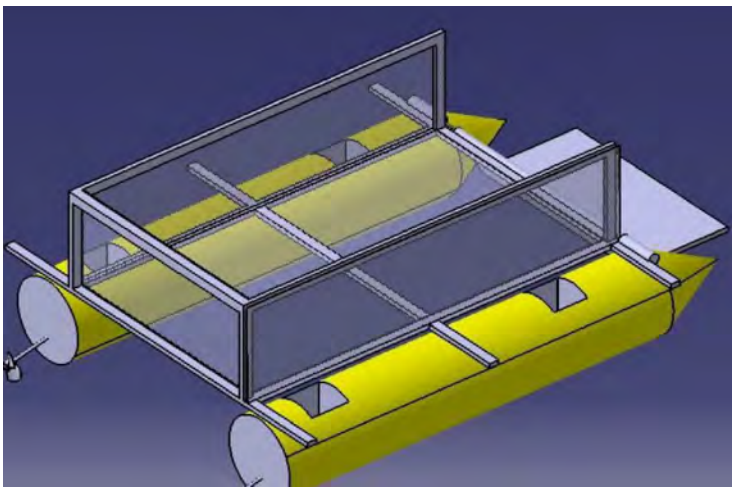

Figure 2.7: Water surface cleaning robot (Rahmawati et al., 2019)

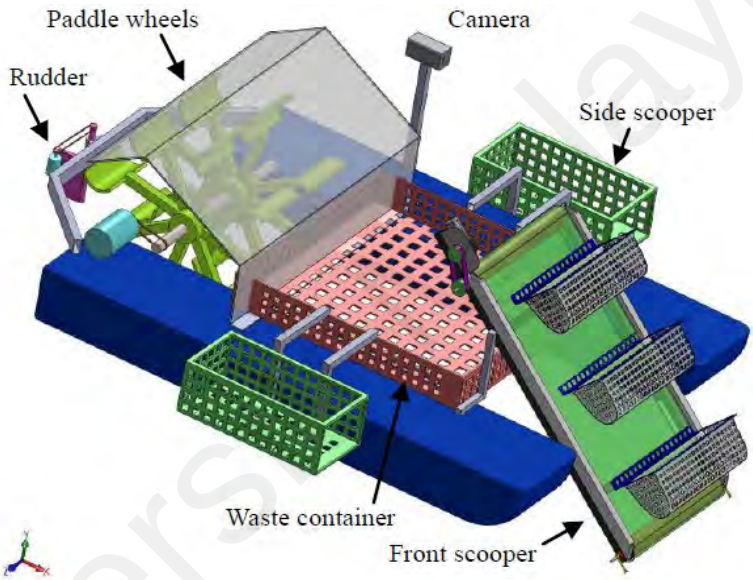

Figure 2.8: Floating waste removal robot (Sumroengrit & Ruangpayoongsak, 2017)

Figure 2.9: The floating waste scooper robot (Ruangpayoongsak et al., 2017)

The floating garbage scooper robot by Ruangpayoongsak et al. (2017) and the floating garbage removal robot Sumroengrit & Ruangpayoongsak (2017) propose double paddle wheels and a rudder (see Figure 2.8 and Figure 2.9). The paddle wheels and the rudder drive and direct the robot. The rudder mounts the paddle wheels on the same shaft between the pontoons, where the rudder is mounted. The robots use two garbage collection mechanisms: a flight conveyor and a scooping arm with a basket. Both mechanisms can overcome water surface tension as the material used is net. The robot's large size helps collect a significant amount of garbage. The tradeoff is the massive size requires a substantial amount of power to move the robot.

The WasteShark (Blue Growth, 2018), USV-SM800 (GREENBAY, 2018), and the JELLYFISHBOT (IADYS, 2019) are robots manufactured by commercial companies to be used at ports and harbours. They all have a catamaran shape (see Figure 2.10, Figure 2.11 and Figure 2.12). The Waste Shark has additional sensors to sense water temperature, pH, conductivity, oxidation-reduction potential, and depth. A net covers the robot's waste container, located between the hulls covering the centre. The USV-SM800 and JELLYFISHBOT have motors and propellers attached to the robot's backside.

In contrast, the WasteShark motors and propellers are connected to the centre from the inside. The WasteShark design seems to be more stable on the water. WasteShark weighs around 40kg, and the length is approximately 1.5 to 2 meters. It can collect 350 kilograms of garbage each time. The USV-SM800 and JELLYFISHBOT use a net attached to the back, making the robot look much more straightforward and giving it the ability to collect a large amount of garbage. The JELLYFISHBOT weighs 18kg, and it can contain 80 litres of waste, whereas the USV-SM800 is 10kg and can collect 80 litres of waste.

Figure 2.10: The WasteShark (Blue Growth, 2018)

Figure 2.11: The JELLYFISHBOT (IADYS, 2019)

Figure 2.12: USV-SM800 (GREENBAY, 2018)

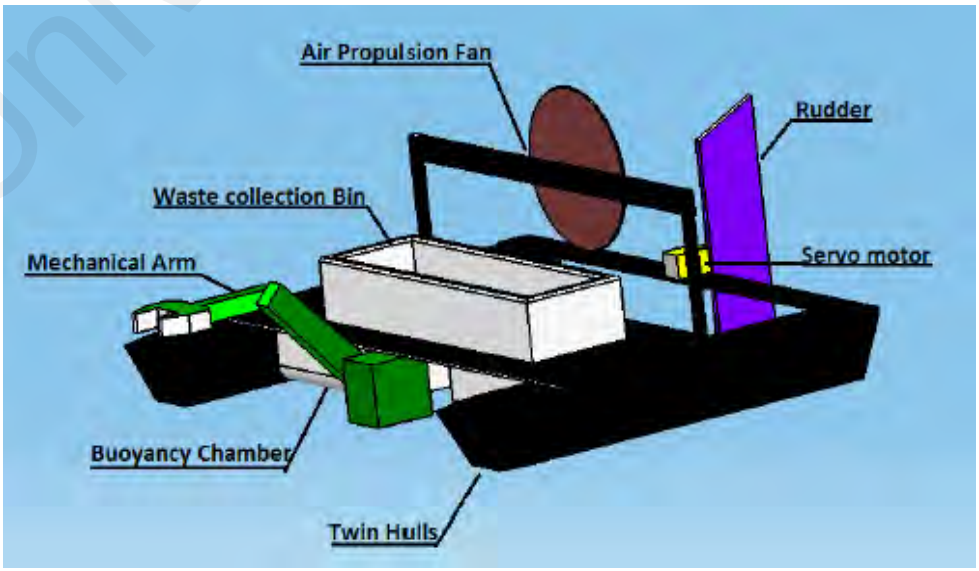

Figure 2.13: Ro-Boat (Sinha et al., 2014)

The Ro-Boat is a pond cleaning robot and is the most miniature robot found with a straightforward design (Sinha et al., 2014). There is a PVC box with two side motors attached to two hulls for movement. Unlike other paddled robots, Ro-Boat has an air propulsion fan to move and a rudder in the back to direct the robot. Figure 2.13 shows the conceptual design of the Ro-Boat mechanical arm for garbage collection. The pond cleaning robot uses the arm to push garbage inside a container on top of the robot. Ro-Boat claims to manage stuck garbage on the edge of the water, but the mechanism looks slow and does not seem valid for most water areas.

All conveyor belt-type robots typically place the conveyor at the robot's front with a garbage collector at the back. However, various extensions were added to the conveyor belt-type design to boost robot connectivity and sanitation activities. Rafique & Langde (2017) proposed an RF radio to control the robot remotely. Agrawal & Bhattacharya (2013) proposed tactile sensors for waste detection because most range-finding sensors, such as ultrasonic, infrared, and others that work well on the land, are unusable in water bodies. They are unable to differentiate between hard rocks and leaves. Su et al. (2009) proposed obstacle detection using two ultrasonic sensors on the robot block diagram that automatically switches 'on' and 'off' of the conveyor belt (see Figure 2.14).

When the RC robot development is shifted to water sampling, there is still some consistency to the structure, shape and moving mechanism. However, there is a considerable change to the sensors, payload and robot agility to rapidly and remotely travel to difficult access locations. One example is the Saramanus & Boonyaroj (2019) robot for water sampling. Their RC robot can collect three samples in one sampling expedition; each sample can go up to 500-ml and separate at a 15-sec interval. Interestingly, the sampling apparatus can go underwater at various depths, for example,

0.50-meter, 0.75-meter, and 1.00-meter. Figure 2.15 shows a schematic diagram of the RC water sampling ship. Water samples are collected using a suction pump. A geographic information system (GIS) is used to locate the sample collection points on the map.

Figure 2.14: The Cleaning Ship (Su et al., 2009)

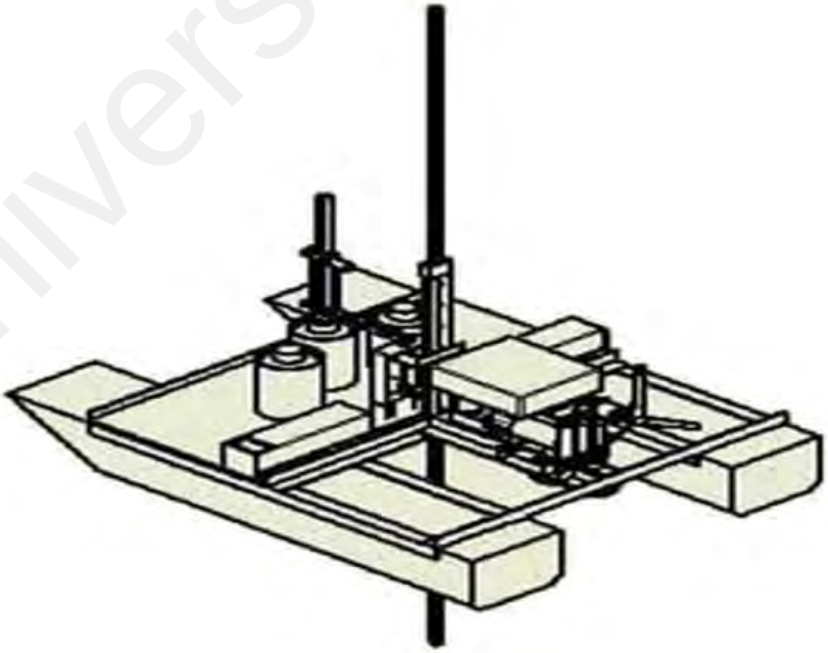

Figure 2.15: Schematic diagram of remote-controlled assisted water sampling ship (Saramanus & Boonyaroj, 2019)

2.4 Development board and electronics for RC water surface robot

A microcontroller unit (MCU) integrates CPU, RAM and an external hard disk on a single chip. In contrast, a single-board computer (SBC) is a complete computer built on a single circuit board, with microprocessor(s), memory, input/output (I/O) and other features required of a functional computer. The MCU and SBC are widely favoured as the development board to manage robots' electronic and communication constraints. The difference is that the SBC offers more options for the GPIO headers, Bluetooth, Wi-Fi, and connectors for cameras, motors, LCD screens, and other essential components in complex robotics development. The Arduino maker board is an example of the more popular MCU, while the Raspberry Pi leads the options for the SBC for low-cost RC robot-proof of concept and academic development.

Many academic researchers have selected the Arduino UNO MCU as their development board (Rahmawati et al., 2019; Rajavel & Shantosh, 2019; Soumya & Preeti, 2018; Thakur et al., 2019). However, these researchers opted for different communication protocols to talk to their Arduino. Rahmawati et al. (2019) use the Xbee S2C configurations as radio receiver and transmitter. In contrast, Rajavel & Shantosh (2019) used the nRF24L01 chip to transfer data to the Arduino. Figure 2.16 shows a Bluetooth model used to communicate with the Arduino in Soumya & Preeti (2018), whereas an RF model handles the communication in Thakur et al. (2019). The Arduino UNO limits the number of sensors and external devices used on the robot due to low computation power and low memory. Using Xbee devices to communicate is more efficient than RF and Bluetooth modules. When in high power mode (up to 1-watt), the Xbee can reach a line of sight range up to 105-km. Toning down the power (about 20-mW) is cost-effective in battery optimized applications.

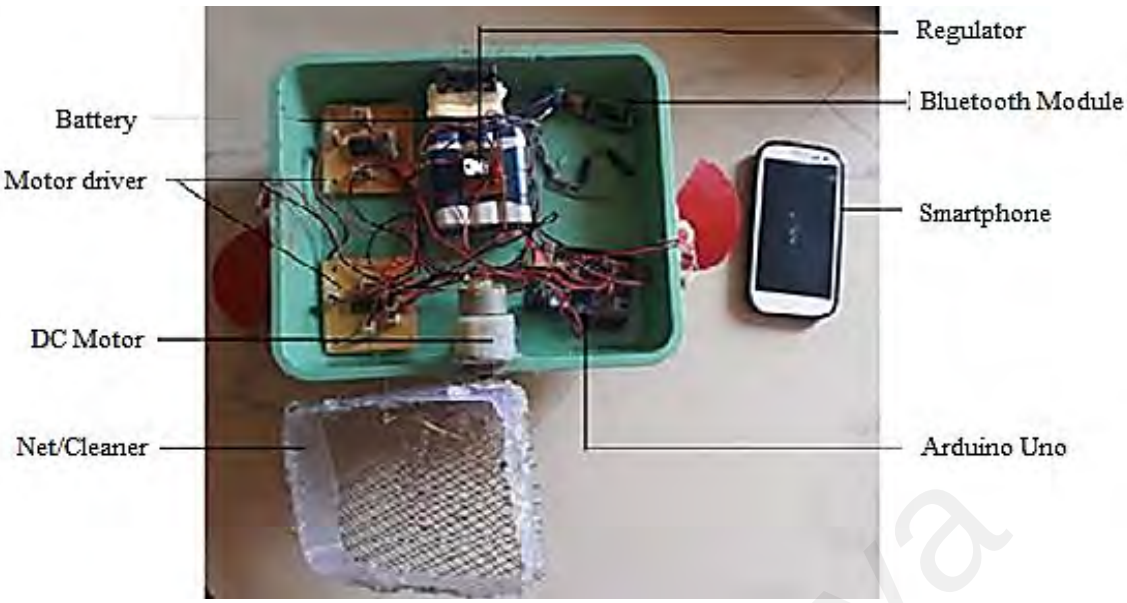

Figure 2.16: The pond cleaning robot (Soumya & Preeti, 2018)

Another Arduino board, the Arduino Mega, is used in Ruangpayoongsak et al. (2017), and Sumroengrit & Ruangpayoongsak (2017). The Arduino Mega has more computation power than Arduino UNO. However, it is still inefficient to handle a camera module on board. A wireless signal is used to send commands to the robot, and a different wireless signal is used to receive camera data on an external device.

The NodeMCU is a firmware or permanent software programmed into a read-only memory. A NodeMCU has a 4-MB of ROM (flash) than an Arduino UNO of just 32-KB, meaning NodeMCU can store more codes than a UNO. Similar to a UNO, the NodeMCU does not need much computation power. A NodeMCU development board is also smaller than Arduino UNO, which attracted Dahlan et al. (2019). In their work, they prepared the robot on an android device and opted for wireless communication via the NodeMCU for data transmission.

The robots are remotely controlled in commercially available projects (Blue Growth, 2018; GREENBAY, 2018; IADYS, 2019). These robots are not low-cost, with the price starting around USD30,000. The maximum distance for controlling the robots is 400 meters (GREENBAY, 2018; IADYS, 2019). Abrams (2018) 's robot is unique compared to the other robots because it can be controlled over the internet. Anyone who has access to the internet can control the robot through a website.

2.5 Sensors and navigation for RC water surface robot

The tactile sensor is helpful to detect waste used in Agrawal & Bhattacharya (2013). They claim tactile is more efficient than other sensors and can differentiate between hard rock and plastic bodies. They also used an oxygen pump to pump oxygen to the water wherever oxygen was needed. They claim that the robot can work autonomously in groups for navigation purposes. They divided the area around the robot into four regions, separated by radii, Rc (critical radius), Rw (working radius), and Ro (outer radius). When another robot or the lake's boundary comes inside the Rc of a robot, an obstacle avoidance algorithm comes into action.

Five different sensors are used as a global navigation system (Ruangpayoongsak et al., 2017; Sumroengrit & Ruangpayoongsak, 2017). They used GPS to get location data from satellites. The inertial measurement unit (IMU) measures orientation, velocity, and gravitational values while the tilt sensor senses robot tilt. Laser digital distance meter and encoder sensor measure speed. GPS, IMU, and tilt sensors are used for robot tracking. The laser is a digital distance meter used to detect floating waste on water surfaces, the encoder sensor used for measuring the robots, and the conveyor belt speed. A camera is also used to stream video from the robot to a laptop used for driving the robot.

Two ultrasonic sensors are used to measure the distance between the robot and the lake edge (Su et al., 2009). The ship can be manually and automatically operated. A motion control strategy based on ultrasonic distance measurement is put forward. By analysing the garbage floating on a lake, cleaning up floating garbage around the bank of the lake has been proposed. A photosensitive resistance is used to detect the daytime and nighttime status. Two ultrasonic ranging systems are used to find the cleaning ship's positioning, orientation, and path planning. As only five situations have been programmed, the motion path of the boat is not very smooth.

The Ro-Boat uses computer vision to detect and recognise pollutants from a gimballed camera with two-axis rotation (Sinha et al., 2014). They claim that multiple object tracking is vital for pollutant detection. They proposed various image processing approaches like foreground and background subtraction, noise filtering, and segmentation. When the camera detects pollutants, the air propulsion system is activated to navigate the Ro-Boat to the contaminant. The navigation is done until the boat reaches within the robotic arm's specified proximity of about 1.5-meter. The underwater propulsion system is activated to steer the Ro-Boat towards the pollutant location.

Ultrasonic sensors are installed to detect obstacles and localise the robot's position and orientation in Wang & Liu (2010). Conceptually, the robot design can integrate IMU, GPS, and vision sensors for garbage detection and tracking. Figure 2.17 shows an illustration of their robot.

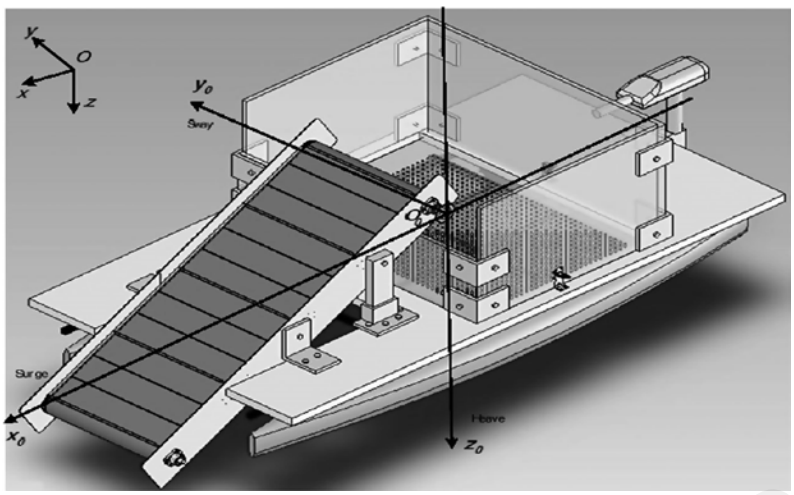

Figure 2.17: Lake surface cleaning robot (Wang & Liu, 2010)

WasteShark uses various onboard sensors like a camera, lidar, depth sounder, and temperature sensor (Blue Growth, 2018). The developer also included water quality sensors like pH, ORP, conductivity, ammonium, dissolved oxygen, turbidity, chloride, nitrate, salinity, ORP, TDS, mV, and resistivity. Their robot is still under development to make it work autonomously. The ability to autonomously return to the base station and re-charging station once the maximum load capacity is reached is under development. They use an algorithm for autonomous path-planning and optical recognition of the docking station entry.

2.6 Water surface robot with automated sampling

The HERON by Clearpathrobotics (2016) is the only literature reporting water surface robots with an automated sampling mechanism (see Figure 2.18). The robot's dimensions are 1350 x 980 x 320 mm, weighing 28kg with a rated payload of 10kg. HERON uses a 2.4ghz signal for communication, and it has a camera, GPS, and IMU sensors attached. HERON uses eight automated syringes for collecting water samples (see Figure 2.19). The syringes can be pre-programmed, so each syringe is sampled at a designated location. NiMH batteries power the robot with 29ah at 14.4v. The battery charging time is 8 hours, for a total runtime of 2.5 hours.

Figure 2.18: Heron USV (Clearpathrobotics, 2016)

Figure 2.19: Sampler syringes Heron USV (Clearpathrobotics, 2016)

2.7 Water surface robot performance in a real-time experiment

Academicians propose many robot designs for water sanitation and sampling. However, only four reported rigorous real-time testing for performance evaluation concerning the robots' communication, electronic system, structural design, or a specific algorithm proposed by the researcher.

Rahmawati et al. (2019) constructed real-time experiments to test their robot's communication performance, the XBee wireless protocol, for navigation. The wireless communication has been successfully carried out by utilising a serial data transmission format. The control system was examined by sending messages remotely, where the robot received all the messages successfully. Successful packet data transferred is verified when

the robot shows an appropriate motion response for the commands. The experiment aims to control the robot moving along a pre-defined trajectory shown in Figure 2.20. They claim that their robot could collect garbage along the path, accommodating garbage loads up to 20 kg per experiment.

Figure 2.20: Path of robot motion during testing (Rahmawati et al., 2019)

Ruangpayoongsak et al. (2017) tested their robot to collect garbage within a selected boundary shown in Figure 2.21. Bamboos enclose the boundary to prevent plastic containers from flowing away. Figure 2.22 shows the robot starts about four meters away, lining its conveyor belt perpendicular to one end of the boundary. After each minute, the robot will count the number of plastic bottles collected using one of its waste scoopers. The result for collecting garbage with the front scooper is varied depending on the robot's speed and the conveyor belt's speed. The collected garbage is 0.81 kg with the lowest speed and 1.252 kg with the highest speed. Using both the front scooper and the side scoopers, the garbage collected increases to 1.18 kg at the lowest speed and 1.71 kg at the highest speed. At the end of this experiment, the results show that the front scooper with the conveyor belt has more capability than the side scoopers.

Figure 2.21: Bamboo enclosed boundary (Ruangpayoongsak et al., 2017)

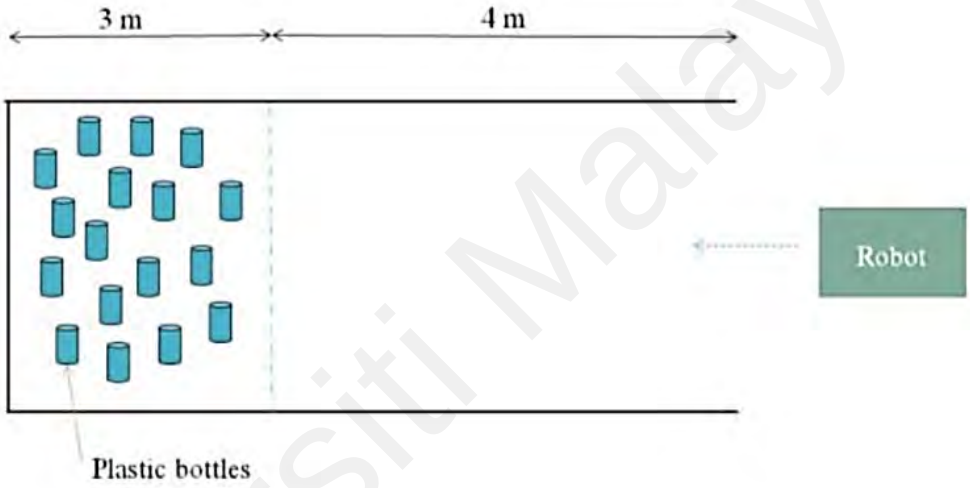

Figure 2.22: Robot starting location (Ruangpayoongsak et al. 2017)

Su et al. (2009) real-time experiment focuses on the sensor performance. Their experiment aims to test the ultrasonic rangers for manual and autonomous operation. The measurement of the ultrasonic sensor is compared to mechanical standards. The ultrasonic sensors' measurement accuracy is one centimetre when measuring the distance of 1 to 300 centimetres. Since the robot's path is formed based on the measured distance, the robot can not move far away from the lake's bank. Their experiment shows that the ultrasonic sensors are sufficient to steer the robot straight, left, right, or make full turns. The ultrasonic sensors do not give the robot metrically precise information. However, Su et al. (2009) claim that higher accuracy is not required to navigate their robot manually or automatically.

Dahlan et al. (2019) carried out experiments in three shallow-water environments, and all the test fields are located in Malaysia. The first test is performed at the river near the Batu Pahat area. See Figure 2.23 for a map of the location. The river canal has a consistent width, and the flow velocity is not high. The robot's size fits nicely to the river canal. The robot is tested with four movements; right, left, backward and forward. The purpose of this experiment is to test the robot's motion controls.

The second test is held at the UTHM lake. See Figure 2.24 for a map of the location. The UTHM lake has a narrow stream going into the open water. The team set up a 5-meter testing path in this experiment and let the robot roam along the path. The team recorded about 0.3 m/s to 1.3 m/s flow velocity at the lake surface due to the light wind breeze.

The third testing location is near the Taman University trench, as shown in Figure 2.25. Since the trench width is about 2-meters, the robot encountered some difficulties to perform right and left movement. Several adjustments are made to overcome the motion issue. The result shows that the robot can reach a selected location under minimum wind conditions. However, the robot is hard to move when the water is not calm.

Figure 2.23: Navigating the river near Batu Pahat (Dahlan et al., 2019)

Figure 2.24: Navigating at UTHM lake (Dahlan et al., 2019)

Figure 2.25: Navigating the trench near Taman University (Dahlan et al., 2019)

2.8 Summary of robot specifications for water sanitation and sampling

Table 2.1 summarises the robot specifications comprised of all robots reviewed in this chapter. Nine robots are physically built and deployed in real-time for research purposes, while three are CAD models that focus only on simulation work. For completion, commercial robots by Blue Growth (2018), GREENBAY (2018) and IADYS (2019) are also included based on available information retrieved from their websites.

Table 2.1: Summary of robots specifications

Author	Dimensions	Weight	Load	Average battery while cleaning	Propeller type	Control system	Is physical robot?
Rahmawati et al., 2019	120 x 80 x 50 cm	8 kg	20 kg	1 hour	three blades	manual (remote control)	yes
Agrawal & Bhattacharya, 2013	115.4 X 55.9 X 29.4 cm	12 kg	6 kg	2 hours	/	Autonomous + Manual	yes
Blue Growth, 2018	190 x 140 x 45 cm	39 kg	350kg	16 hours	2 electric thrusters	remote control (currently) or autonomous (future)	yes
GREENBAY, 2018	80 x 65 x 18 cm	10kg	15kg	45-240 m	2 electric thrusters	manual (remote control) 400m	yes
IADYS, 2019	70 x 70 x 50 cm	18 kg	80 L	/	Three electric thrusters	Radio-controlled guidance: 400 m	yes
Ruangpayoongsak et al., 2017; Sumroengrit & Ruangpayoongsak, 2017	/	/	/	/	Paddlewheels	manually operated over LAN	yes
Thakur et al., 2019	70 x 30 x 17 cm	/	/	/	two blades	manual (remote control)	no (CAD)
Su et al., 2009	/	/	/	/	two propellers	manual (remote control)	yes
Wang & Liu, 2010	/	90 kg	480 kg	/	one propeller	autonomous mode or teleoperation	yes
Dahlan et al., 2019	/	/	/	3.8 h	two propellers	teleoperated	yes
Sinha et al., 2014	/	4kg	/	/	air propeller	/	no (CAD)
Rafique & Langde, 2017	122.2 x 48 x 52 cm	/	/	45-70 m	Water Wheel	manual (remote control)	no (CAD)

2.9 Chapter summary

This chapter reviews the development of water surface robotic platforms by the water sanitation and sampling community worldwide. The majority of the robots are the remote-controlled type where a human operator navigates the robot. There has been an initiative to move into semi-autonomous platforms. Still, challenges remain in balancing structural design, material, electronic, remote communication, and managing the algorithm for autonomy, especially when targeting a low-cost development. None of the low-cost robot design and development proposed an open design; the robot states and data-pushed online to load off the on-board processing and leverage advanced searching and tracking. The following chapter carefully describes the design consideration and the approaches taken in the development.

CHAPTER 3: METHODOLOGY

3.1 Overview

This chapter presents the methodology performed to achieve research objectives 1 and 2. Objective 1 focuses on designing and developing the water surface robot structure, control, and electronics. Objective 2 concerns the design and development of the APIs for remote robot connection, data communication and positioning.

3.2 Robot structure

This section describes the design of the low-cost water surface robot structure. The design includes robot floating and balancing consideration, garbage-collecting mechanical attachment, automated mechanical sampling system, power sources and distribution, cooling system, and the general wiring.

3.2.1 Material and body

The main material used for the body is PVC (Polyvinyl chloride) pipes, widely used to deliver clean water to our houses. PVC is used due to its low effect on the environment, material strength, and easily available material. Three pipe sizes have been used (i.e. 82mm, 50mm, 32mm). The total weight of the body without load is around 6kg. The dimensions are 40cm height, 60cm length, and 66cm width. Two motors are attached to both sides from the outside. The design also includes four ultrasonic sensors and a camera on the front. Two waterproof boxes on the sides guard the batteries, motor controller, cooling fan, and a 100Amp relay. One waterproof box on the top contains Raspberry Pi, a power supply, battery life indicator, external sensors, and an on-off switch for the system. A net with small holes is attached to the inner part of the boat for collecting garbage. Figure 3.1 shows the structural design of the water surface robot.

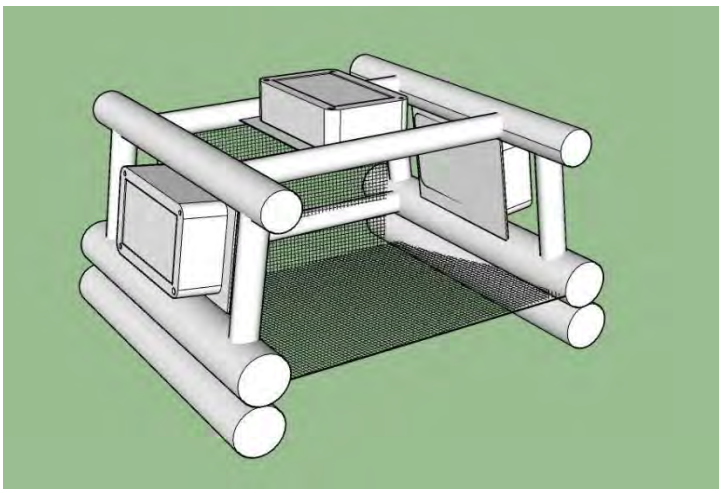

Figure 3.1: Structural design of the water surface robot

Balancing the body is a challenge; layering double pipes parallel along the robot's movement can support buoyancy and ensure the centre of the floating force is fixed at the symmetric axis. The decision to use two 1.2kg batteries adds to the overall weight but placing them on the sides focuses the weight at the centre of mass of the robot. Additionally, foam sticks are also used to support the body. It is cylinder-shaped made from foam, and it can balance the body and give it more resistance to float on the water. The same foam stick is used in swimming life jackets; it helps the person who wears it float on the water. The two pipes at the bottom and the foam sticks are the base parts the body stands on. The body's total volume gives the total weight. Equation (1) is for finding cylinder shape volume.

$$v = \pi * r^2 * h \tag{1}$$

There are four pipes with a radius of $4.12cm$ and $60cm$ in length. There are two cylinder-shaped foam sticks with a radius of $3.352cm$ and a length of $60cm$. Therefore, the total weight is estimated around $4x(3.14x4.12\ x60) + 2x(3.14x3.352\ x60)$, resulting in $12674cm^3 + 4230cm^3 = 16{,}904cm^3 = \sim 17$ litres. Since one-litre water is equal to one kilogram, the body carries ~17kg of weight.

A widely available nylon plastic fishing net is used as a container to trap garbage. The net has tiny holes around 1.5mm in diameter, is light in weight, and is a durable material. It is easy to fasten with simple plastic cables and simple to replace if damaged. The net's dimensions used on the body are 52cm in length, 48cm in width, and 26cm in height. The final volume is $52x45x26 = 60{,}840\ cm^3$ equal to about 60 litres.

3.2.2 Sampling system design

Lake water sampling is collecting water samples from different locations in a lake. Traditionally the grab sampling method (see Figure 3.2) fetches water samples for lab testing. Grab sampling transfers water into clean, airtight containers to not contaminate the sample. Researchers usually carry several containers to collect many samples from many areas of the lake. The samples can come from the lake surface or underneath at a different column or depth. After each sample is grabbed, the container will be labelled with the location, depth and timestamp.

Figure 3.2: collecting a water sample manually (Emma, 2018)

In this work, four containers are proposed onboard the robot's body, and each container can collect 200ml of water for a total of 800ml sample at full capacity. The straightforward manner of designing an automatic water extraction mechanism is by using pumps. However, pumps design becomes an issue when avoiding contamination is considered. Pumps usually work by letting water through the valves before hitting the inside of the container. Therefore, the vacuum method is considered to avoid contamination. Each container has two hoses attached to the header, one used by an air pump to suck the air out of the container, and another sucks lake water into the container. An electronic valve is used to control the water flow. The estimated time to fill the container is 20-seconds.

The air pump will stop when a liquid sensor underneath the container lid senses water when the container reaches total capacity. The liquid sensor sends the air pump a signal to close the valve. Turning counter-clockwise releases the container from its lid so the extracted water is checked in-situ with a water quality probe or can be passed on for analysis in the lab. Figure 3.3 shows the design of the automatic water sampler system. A single pump manages all four containers' activities, but the water extraction cannot be done simultaneously. Only after the pump finishes with a container can the pump move onto another container to be filled.

A clear plastic hose or tube is used to channel water from the lake to the containers, while a reel is used to pull and release the hose underneath the water surface for sampling at different depths. The robot is pre-programmed with two depth settings, (1) 25 cm depth for surface sampling and (2) 100 cm depth for underwater sampling. Heavy metal is attached to the end of the hose so the weight can help submerge the hose underneath. The reel carries a servo motor with 3kg torque to manage the hose. An encoder sensor on the

reel tracks the number of rotations made to estimate how deep the hose is released. It takes just under one minute for the hose's length to reach one meter underwater.

The reason for the different depths is due to an exciting requirement from the water community. It is observed that profiling water quality at the lake is a complex task because the properties of water at the surface can be significantly different going down. Eutrophication can contaminate the water surface with a high value of chlorophyll-a. Microalgal infestation can show a higher density of chlorophyll-a down the column and not at the surface. Sediment problems can cause a high density of turbidity at the bottom of the lake. Column sampling is significant to the water quality monitoring at the lake; thus, the robot design in this study includes two sampling depths for proof of concept.

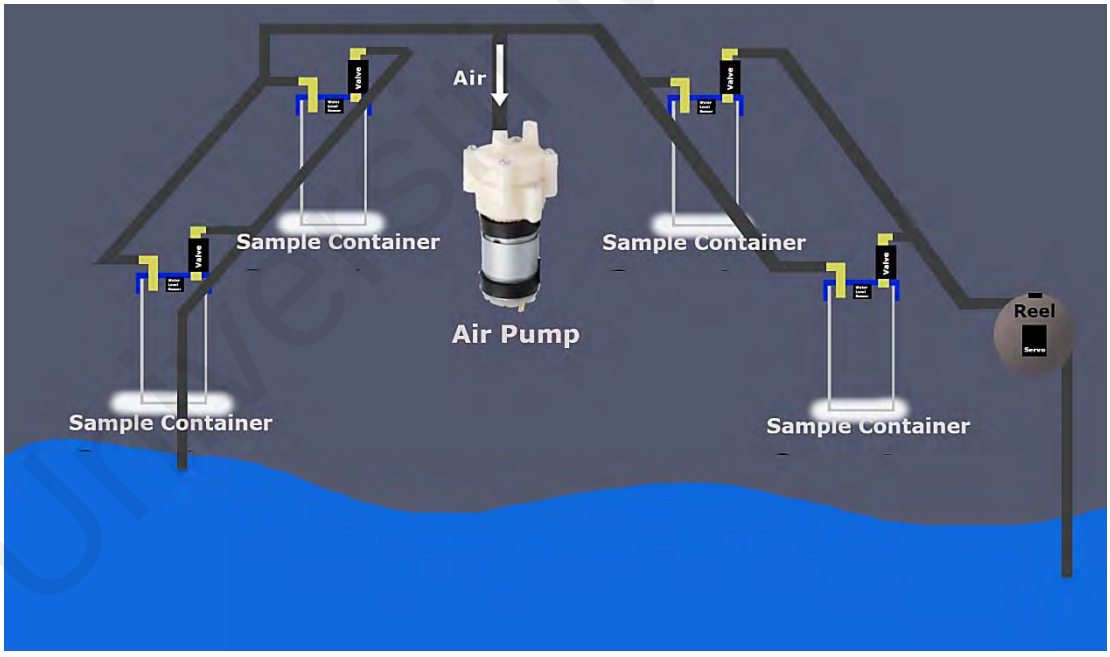

Figure 3.3: Mechanism of the water sampling system with four containers

3.2.3 Power design

The power source used for the robot is two 12v 7.2 AmH lead-acid batteries shown in Figure 3.4. They each weigh approximately 1.2kg. Each battery is positioned inside a waterproof box attached to the robot's sides. This type of battery is used due to its low price and big capacity. The motors are directly connected to the batteries, with two power supplies powering the sensors. Two converters are used to convert a 12v to 5v 5 amp and convert a 12v to 3.3v 1 amp. A 12v 20amp battery charger takes around 20 minutes to charge each battery fully. The power consumption for each component is listed in Table 3.1.

Table 3.1: Power consumption table

Device	Power range
Raspberry Pi 3B+	350mA ideal 950mA heavy used
Two brushless motors	Each 200 W
Simplertk2b	Power consumption: 80mA @ 5V Active antenna power supply at 3V up to 75mA
Two 3.3v relays Two 12v 100amp relays	Each 75mA at 5v
Camera	0.4W to 1.4W
Ultrasonic sensors	200mA all
12v water pump	<700mA
Nine-axis BNO055 compasses	12.3 mA.

Figure 3.4: 12v 7.2AmH lead-acid battery

3.2.4 Cooling system

Working at the lake means the robot is susceptible to different weather conditions that may affect the electric components. Heat is one of the problems, although a fan helps to reduce the heat somewhat. There are in total four waterproof boxes containing various electric components. Three of the smaller boxes has a dedicated fan running fully when the power is turned on. A much larger box, the main box, has three fans operating inside it. Another cooling method is covering the waterproof casing with a heat-reducing foam. The metallic surface of the foam deflects sunlight, so the box does not absorb too much heat. Figure 3.5 shows the fan and heat-reducing foam.

Figure 3.5: Fan and heat-reducing foam

3.3 Electronics

Figure 3.6 shows the electronic diagram proposed in the robot's design. The 12V battery powers any electronic components that can take 12V, such as the fans. Other components, such as the Raspberry Pi 3b+ for example, require conversion to 5v and 3.3v. The Raspberry Pi 3b+ gets heat up too quickly; thus, a dedicated fan is required for the microcontroller alone. The electronic speed controller (ESC) controls and regulates the speed of the left and right brushless motors. The brushless motors are pre-programmed to take nine-speed settings, lowest as speed one and highest as speed nine. The speed setup is analogous, ranging between 1km/hour and 4km/hour. A small LCD screen indicates the battery level. If the battery level drops to 11.2, a full charge is required. The battery requires 2 hours to recharge and can last 1.5 hours on the field.

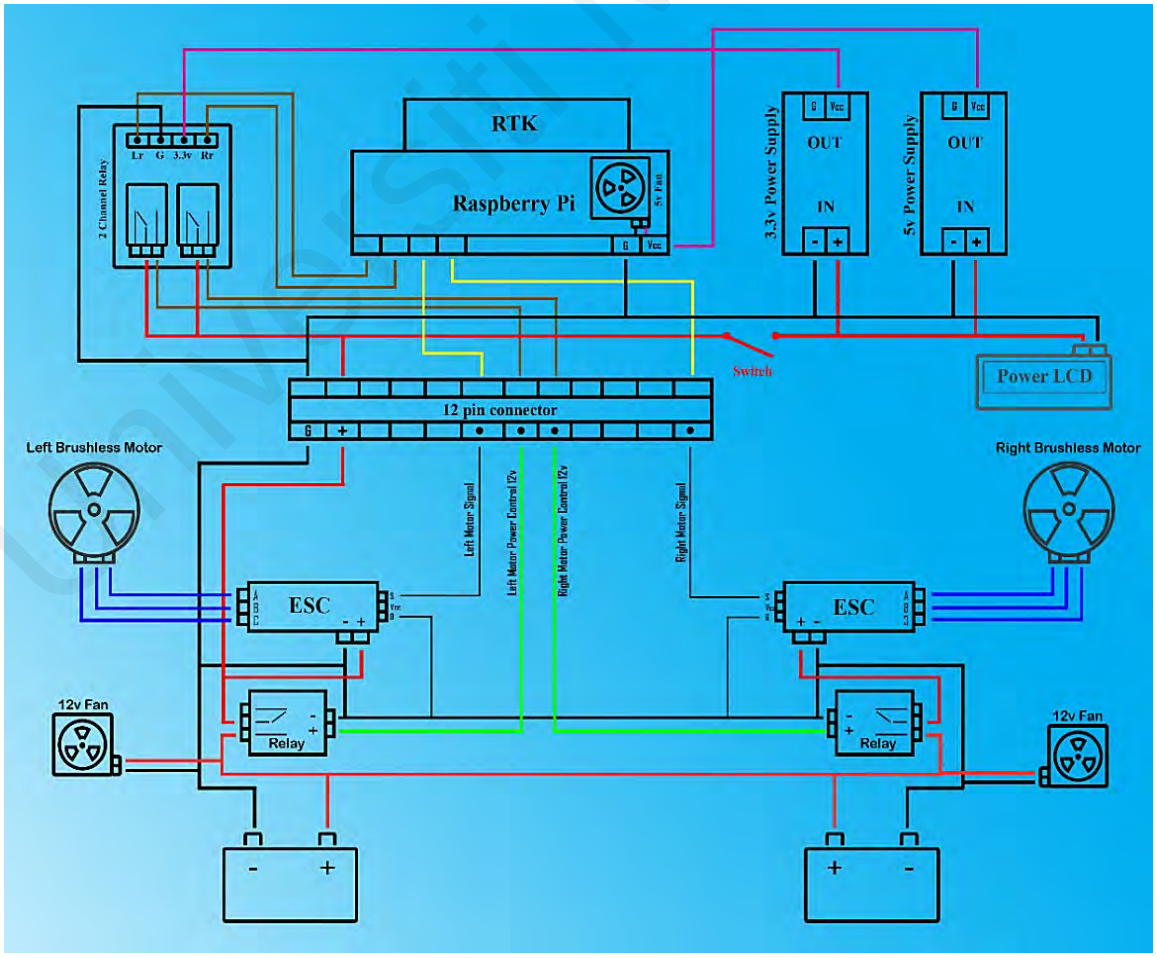

Figure 3.6: The robot electronics diagram

3.3.1 Raspberry Pi 3 B+

Raspberry Pi is a small single-board computer (SBC) that runs on a light version of the Linux operating system called Raspbian. It is a great controller to use on robots since it has 1 GB of RAM and a 1.4GHz 64-bit quad-core processor, with dual-band wireless support 2.4Ghz and 5Ghz Wi-Fi. There are four USB ports to connect devices and sensors that rely on USB connections. The camera port is another helpful feature of the Raspberry Pi since this study requires visual sensing. Raspberry Pi uses a micro-SD card as external storage, saving the operating system and the Raspberry Pi model 3b plus shown in Figure 3.7.

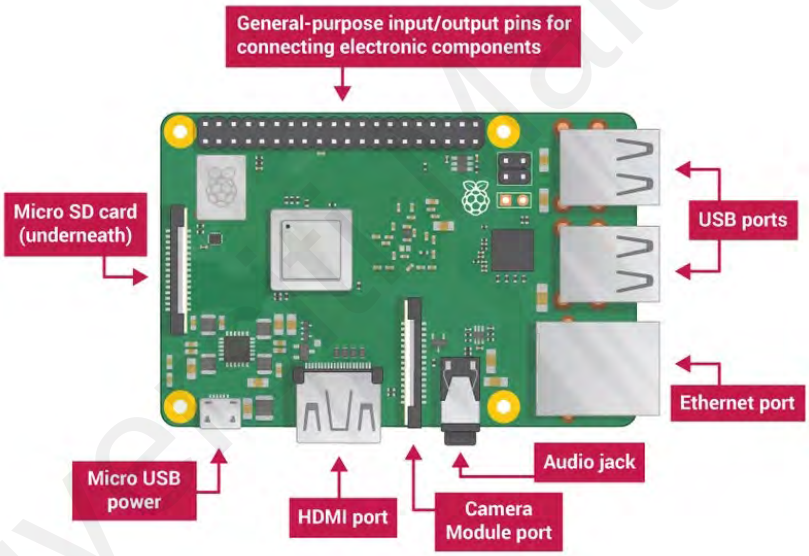

Figure 3.7: Raspberry Pi model3 B+

Figure 3.8: Raspberry Pi 3+ 40 header GPIO pins

Raspberry Pi Model 3 B+ has a micro-USB port used as a power port. The founder company recommends using 5v 2.5Amp as a power source. However, 5v 5Amp has been used in this project to supply extra power for future usage. The Raspberry Pi has an HDMI port, TV output 3.5 mm port, display interface, a gigabit ethernet port for network connection, and 40 GPIO pins. The GPIO pins control the external devices and get feedback from the sensors. Figure 3.8 shows the GPIO pins. The GPIO header provides the following power and interface options.

- 3.3v (on two pins)
- 5v (on two pins)
- Ground (on eight pins)
- The general-purpose input and output 3.3v
- PWM (pulse width modulation)
- I2C
- I2S
- SPI
- Serial

3.3.2 Brushless motor and ESC motor controllers

Brushless DC motors (BLDC) are standard in industrial applications worldwide. There are brushed and brushless motors at the most basic level and DC and AC motors. Brushless DC motors are DC motors that do not contain brushes. These motors provide many advantages over other types of electrical motors. A brushed DC motor has permanent magnets outside its structure, with a spinning armature inside. The magnets, which are stationary on the outside, are called the stator. The armature, which rotates and contains an electromagnet, is called the rotor. Figure 3.9 shows the BLDC motor's internal parts.

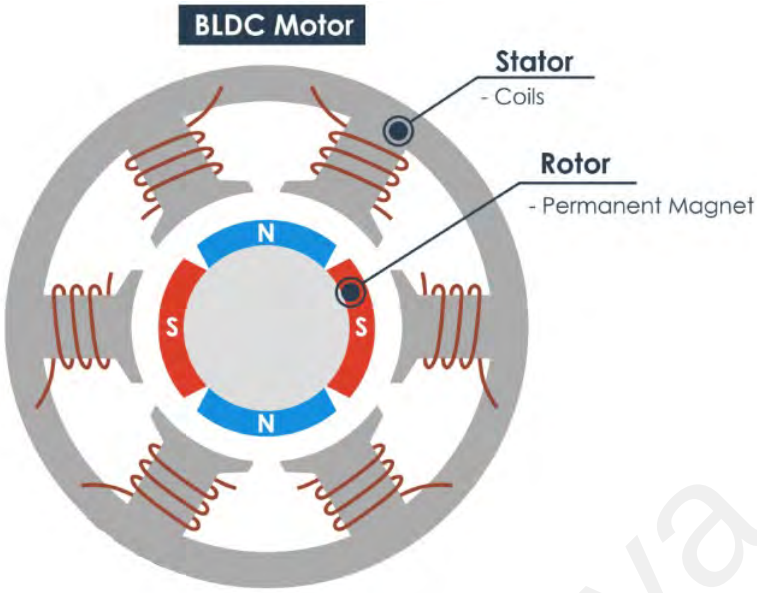

Figure 3.9: BLDC motors internal parts

Figure 3.10: 12v BLDC underwater motors

Two brushless motors are used in this project for driving the robot. The two motors are in the shape of an underwater thruster covered by plastic pipe. Both motors use a propeller to push the water in the desired direction. Figure 3.10 shows the motors. The motors work on 12v power, and the seller company recommends a 40Amp ESC. The motors have 460KV, 5300 rpm speed, and a thruster of 3kg to 5kg. The dimensions of the motors are as follows: 90mm diameter, 80mm Propeller diameter, and a 3.5mm bullets connector to connect with the ESC.

Figure 3.11 shows the connection between the motors and the ESCs. The ESCs control brushless motor speed and movement by activating the appropriate metal–oxide–semiconductor field-effect transistors (MOSFETs). The MOSFETs are responsible for creating the rotating magnetic field so that the motor rotates. The universal battery eliminator circuit (UBEC) ESC is selected due to its power efficiency. UBEC is a switch-mode DC regulator that can take high-voltage (5.5v to 26v) power from the battery pack and convert it to a consistent, safe voltage for the receiver, gyro and servos.

One significant advantage of using a brushless motor is efficiency, as these motors can continuously control maximum rotational force (torque). It needs more giant magnets for a brushed motor to have the same torque as a brushless motor. Therefore, even small brushless motors can deliver considerable power. Motors can be controlled using feedback mechanisms that deliver precise torque and rotation speed. Precision control reduces energy consumption and heat generation, and in cases where motors are battery-powered lengthens the battery life.

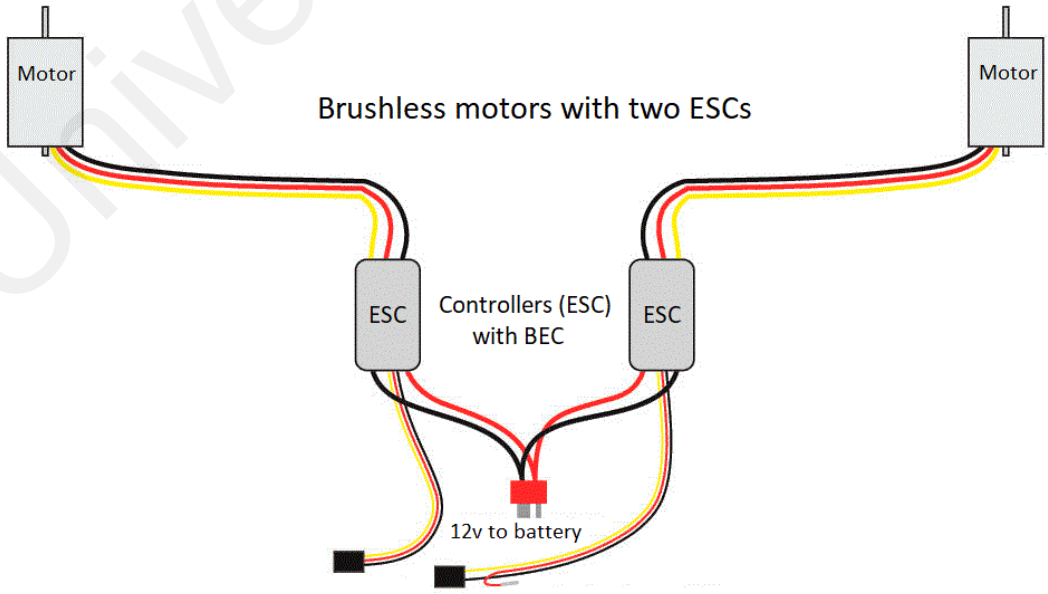

Figure 3.11: ESC motor controller

3.3.3 Water sampling system

Section 3.2.2 describes the water sampling system's structure comprising two main parts: the water extraction mechanism and the hose reeling mechanism. Briefly, the water extraction module requires an air pump, four electronic valves, and four water level sensors. The reel requires a 3kg servo motor, an encoder sensor, and a limit switch. This section describes the electronics in the water sampling system. The air pump runs on 12v with 1.2L/s motor speed. The water level sensor will send feedback to the system stopping the motor when the container is full. The air pump used is shown in Figure 3.12.

The water level sensor is a straightforward sensor used to detect the liquid level, and it returns voltage between 1v to the supplied voltage. This sensor was used inside the water sampling containers, and it returns feedback to the system when the container is full to stop collecting water. The sensor was modified to switch to turn off the air pump after the container is full of water. The water level sensor is shown in Figure 3.13. An electronic valve is attached to the water sampling container. Similar to the air pump, the valve also uses 12v power. The valve has a 3.5mm inner and outer diameter. The robot system controls the valve. When the robot receives the collected water signal, it will open the selected container and close all the others to let the air pump suck the air from that specific container. The electronic valve is shown in Figure 3.14.

The servo motor for the reel is a 3kg motor with vital gears that can handle the water hose. Servos in the market typically move at specific degrees, such as 180 or 90 degrees. The servo was modified to move continuously in this work, so the reel action is smooth. The encoder is a cheap laser encoder that calculates the holes while the encoder disk is moving. The sensor has limited accuracy, and it works perfectly when the movement is

slow. The limit switch is used to stop the servo from drugging the hose when it reaches its end. Figure 3.15 shows the underwater roller.

Figure 3.12: Water Sampling System Air Pump

Figure 3.13: The water level sensor

Figure 3.14: The electronic valve

Figure 3.15: Proposed reel system that pulls and releases the hose underwater

3.3.4 Sensors

External sensors like the ultrasonic, IMU, camera, and voltage indicator for battery consumption, are essential components in most robot designs. The sensors are conventionally used to perceive obstacles in the surrounding, set and observe robot heading and bearing direction, and monitor the battery's capacity.

The ultrasonic sensors measure the distance by using an ultrasonic wave. The ultrasonic sensor emits an ultrasonic wave and receives the target's reflection; it measures the distance following the time taken between the transmission and reception. The distance will be calculated using Equation (2).

$$L = 0.5 * T * C \tag{2}$$

L represents the distance between the sensor and the target, T is the time between the emission and reception, and C is the signal speed. The value is multiplied by 0.5 because T is the time for the go-and-return distance. The sensor used in this project is the same as the one used on cars for detecting obstacles while parking the vehicle. The sensor is waterproof, and it can measure distances from 30cm to 3m. Figure 3.16 shows the ultrasonic sensor used on the robot. The following list shows specialities enabled by the ultrasonic sensor:

1. A transparent object is detectable. Ultrasonic waves can reflect from a glass or liquid surface and return to the sensor.
2. Resistant to dirt detection is not affected by the accumulation of dust or dirt.
3. Complex-shaped objects detectable presence detection is stable even for targets such as mesh trays or springs.

Figure 3.16: 3-meter range ultrasonic sensor

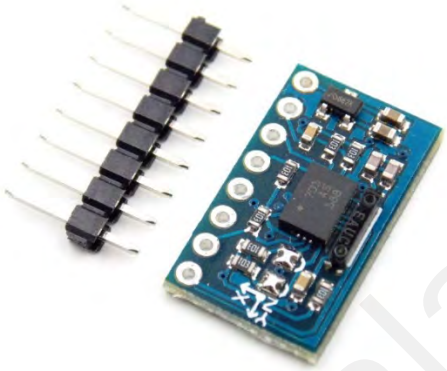

Figure 3.17: The sensor BNO055

Figure 3.18: 5 megapixel Raspberry Pi camera with its 3D casing design

Figure 3.19: Battery voltage indicator with a voltage sensor

The sensor BNO055 is a System in Package (SiP) solution that combines a three-axis 14-bit accelerometer, an accurate close-loop three-axis 16-bit gyroscope, a three-axis geomagnetic sensor and a three-axis geomagnetic sensor 32-bit microcontroller running the BSX3.0 FusionLib software in a single device. The sensor is significantly tiny, yet the sensor fusion of the BNO055 makes integration easy. The BNO055 is advantageous in applications requiring context-awareness, such as for robot navigation in this work. Figure 3.17 shows the smart sensor BNO055.

This study's other context awareness application is garbage detection through vision sensing. A 5mp camera with 1080p video recording capability is connected through a ribbon cable to the CSI (Camera Serial Interface) port of the Raspberry Pi 3 B+. The camera is mounted on the highest point of view of the robot facing forward. Figure 3.18 shows the 5mp Raspberry Pi camera.

The battery voltage is monitored to indicate the battery life. Two types of voltage readers are used in this project, one with the LCD, and the other is a sensor that returns an analogue signal to the Raspberry Pi. Figure 3.19 shows both readers. The LCD voltage reader is waterproof and attached inside the main electronic box. The reader has a button that changes the display mode from voltage reader to battery capacity indicator or off mode where the LCD turns off. It is connected directly to the battery with two wires. The voltage sensor can read a range from 0 to 25v, and it provides the Raspberry Pi with the battery voltage to live show battery capacity to the user through the android app.

3.3.5 Control components

Raspberry Pi is the central computer used to control the external sensors and devices connected to the robot. However, the Raspberry Pi has limitations in handling many external devices and controlling high voltage devices. Several adjustments are required to handle the limitation. For example, relays have been used to control the higher voltage components. The IO Expansion Board extends the number of IO ports, and the analogue to digital converter (ASD) gives the Raspberry Pi the ability to read analogue signals. The adjustment also involves a logical level converter for controlling all 5v devices. The use of relays, ASD, and logical level converter are further discussed next.

Relays are switches that open and close electromechanically. Relays control one electrical circuit by switching on and off another circuit. A 5v relay is used for controlling low current, and it is held directly by the main Raspberry Pi since the 5v relay can use a 3.3v signal. The model has three output pins working as two switches. When one is on, the other becomes off. A 12v 100amp relay is used to control the high currents. Since the central controller cannot control the 12v signal, the 12v relays are controlled by the 5v relays, giving a high current-controlled circuit for turning on and off the 40Amp ESCs. The 5v and 12v relays are shown in Figure 3.20.

The analogue-to-digital converter (ADC) can read the analogue signal for microcontrollers without an ADC chip like the Raspberry Pi 3b+. The ADS1115 has 16-bit precision at 860 samples/second over I2C. The chip has four input channels to read analogue signals or two differential channels to measure the two signals' differences. It even includes a programmable amplifier to help boost to the full range as a bonus. This ADC can run from 2v to 5v power/logic, measure a broad range of signals, and is easy to use. It is an excellent general-purpose 16-bit converter.

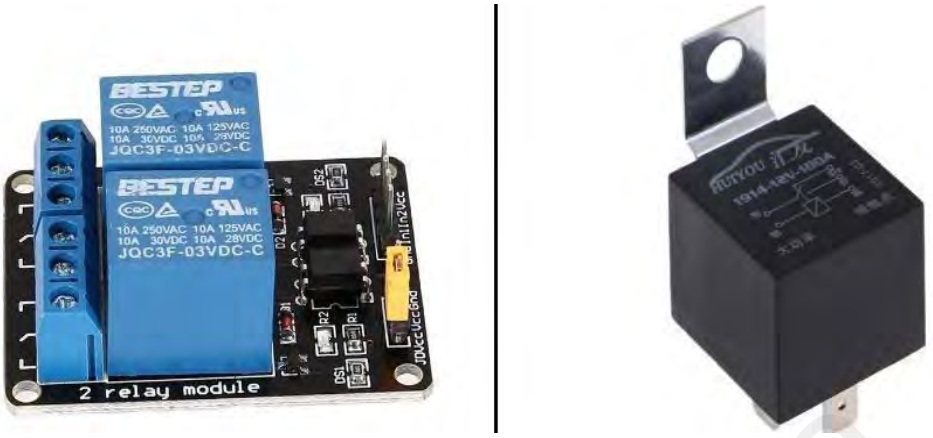

Figure 3.20: 5v and 12v relays

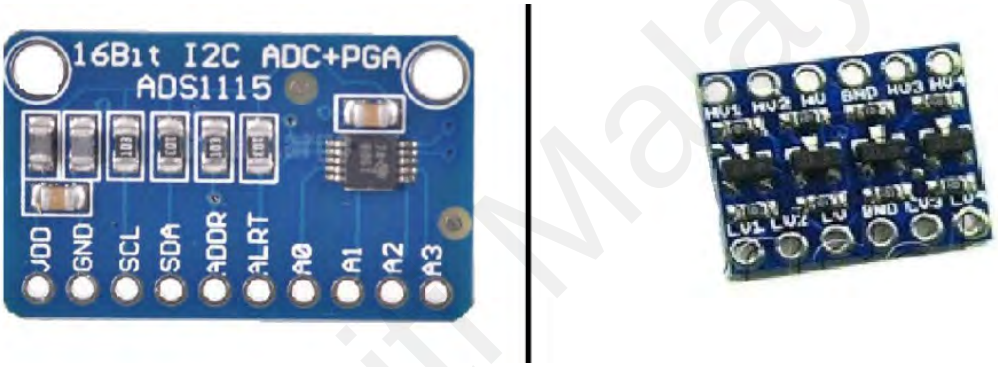

Figure 3.21: 16bit ASD1115 and Logical level converter

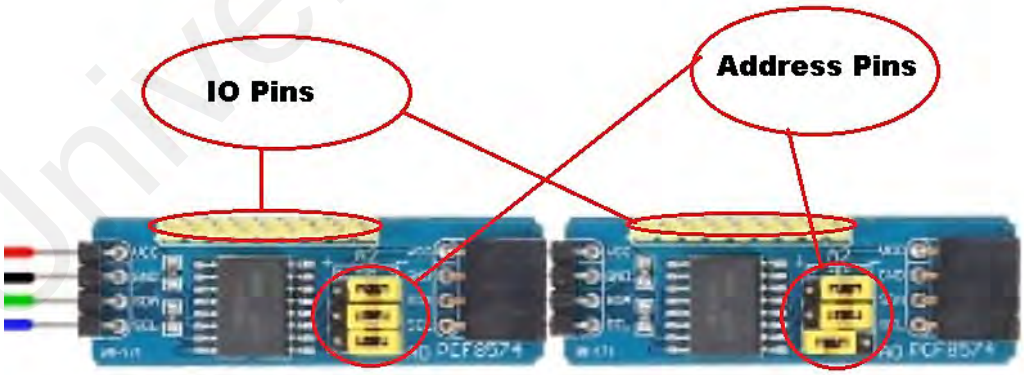

Figure 3.22: two IO expander PCF8574 attached

The connection is made via I2C. The address can be changed, and it has four options. The address control allows having up to four ADS1115 connected on a single 2-wire I2C bus for 16 single-ended inputs. That will give the Raspberry Pi the ability to read 16 analogue signals simultaneously with only two wires. Another chip used to make the Raspberry Pi more functional is the logical level converter (LLC). This chip is a small-sized chip that can convert a 5v signal to a 3.3v signal and vice versa effectively at high speed. This converter works with 2.8v and 1.8v devices as well. Each level converter can convert four pins on the high side to four on the low side, setting four channels for each site. The level converter is straightforward to use. The board needs to be connected to two voltages sources. The high voltage 5v connects to the HV pins while the low voltage 3.3v connects to LV. Grounding is done through the GND pins. Figure 3.21 shows both chips.

The IO extender is a chip used to extend the input and the output on the main computer. From 40 pins on the Raspberry Pi, only 13 are digital IO pins, and the other IO pins are usually multi-function pins. PCF8574 was used to extend the IO of the Raspberry Pi, which allowed us to add more sensors and control more external devices. PCF8574 has 8 IO ports. It uses the I2C protocol for communication and serves any 5v and 3.3v systems. PCF8574 uses 3 bits for the address, allowing more PCF8574 to get up to 64 IO pins. The address can be selected easily by changing the position of the jumper connector on the card. Figure 3.22 shows two PCF8574 cards attached.

Once the robot structure and electronics are completed, the development continues with software development. The critical considerations for the software design are presented next. Designing the APIs with diligence can help the robot system leverage the hardware resources fully.

3.4 Software design

The software platform to control the robot has been specifically designed for Android devices. Android is an open-source operating system installed on many other operating systems, like Windows, macOS, and Linux. This section describes the Android brief description and all the developed app interfaces.

3.4.1 Robot controlling app with Android

Android is a mobile operating system built on Linux kernel and other open-source software designed for touch-screen mobile devices, such as smartphones and tablets. Android device is widely available worldwide, and it is easy to set up android emulate on any personal computer (PC). Android Studio is the platform used for developing android apps, and it uses Kotlin and Java. However, it is possible to use other languages like C and C++ for programming too. An Android app is under development to be used as a controller for this robot, using a combination of Java and Kotlin codes. Only Android 7 and above can run the app. The app has four main views to configure and control the robot: Pin Code View, Main View, Settings View, and Map View. All these views are explained below.

The Pin Code View is the first thing the user can see after opening the app. From Pin Code View, the user can search for the Robots available on the network and select one to control it. It also has a security feature that asks the user to enter a Pin Code before accessing the controlling part. The pin code is a simple 6-digit code saved on the app to add app security level. The user can modify the pin code later from the app settings. Whenever the user exit from the app or press the home button on the phone, the app will again ask the user to enter the pin before entering the app. Figure 3.23 shows the Pin Code view.

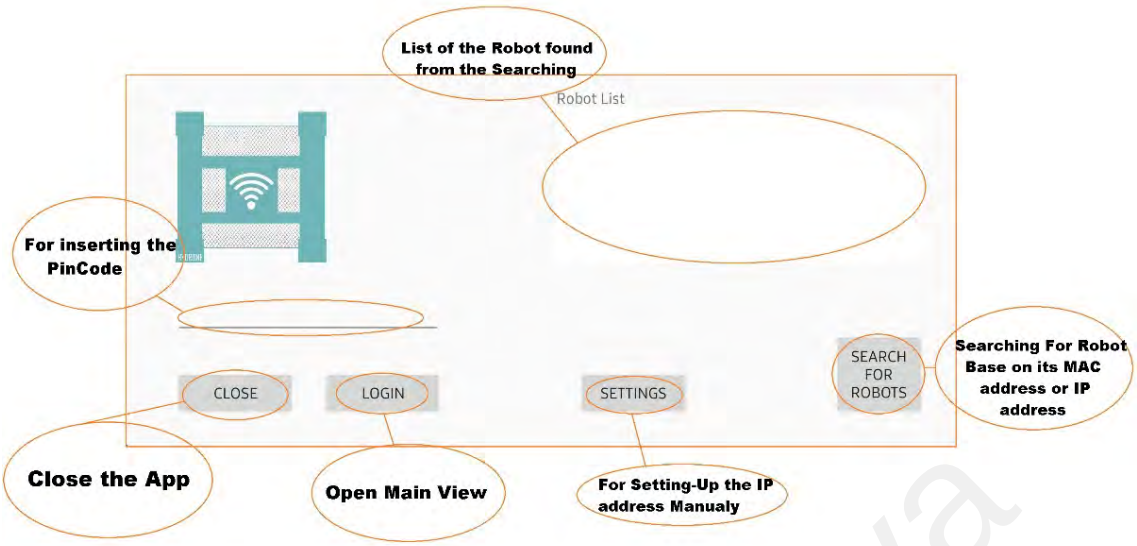

Figure 3.23: Pin code view

Figure 3.24: The Main View

The app uses a saved list of MAC addresses to find the robots on the network. A button is used to search for the robots, as shown in Figure 3.24. After pressing the button, it takes 10 to 15 seconds for the app to find all the Robots. When the search is done, the Robot list up all the found robots, and the user will select one of the robots. If the search did not find any device, the user could also enter the robots IP address using the setting button, which takes him to the Settings view described in the upcoming section.

The Main View is the central part of the app, where the user can control the robot remotely. The Main view contains many components to control and monitor the robot sensors, as shown in Figure 3.24. The start and stop button on the top right of the screen is used to start communication and receive data from the robot. After the connection is established successfully, it displays '*Stop*' so a user can kill the communication at any time. The centre of the screen displays a live stream from the robot's camera. Pressing the recording button will save the live stream as videos on the external USB drive attached to the robot.

The Main View also shows two sets of slider buttons. The horizontal slider button controls the robot's left and right turn, while the vertical slider button panels the robot forward and backwards. A compass is shown at the top of the screen to support the navigation. Below the compass, a motor power level allows the user to manually select the motor's speed. There are nine-speed settings that will increase or decrease the propellers' rotation. The robot has two side batteries. Two indicators are included in the Main View to monitor the batteries' level. The Raspberry Pi's performance is influenced by heat. Two indicators are placed at the top panel to display the pi's temperature and signal strength.

There are four buttons created for water sampling purposes. Each button is dedicated to a single water container. All four containers can be used at any time; however, the design considers filling one container at a time. When one button is pressed, the icon colour will change from green to red, indicating it is in action. The other buttons cannot function, and their icon colour remains green. After the container is filled, the dedicated button for that container remains red. A user can only reuse the containers after emptying them. Resetting the buttons requires the user to press each button three-time, so it becomes green again and available for water sampling.

Section 3.2.2 depicts the water container fetching water at two sampling depths; the surface and underwater. Two containers are reserved for surface sampling, while two others are underwater. Another slider button is created in the Main View to release the underwater sampler reel according to the desired depth. The sampler hose is kept to a maximum of 1-meter underwater in this project. In the future, a longer hose can be installed if desired.

The robot has sensors to support navigation; thus, the console features a text field that displays readings from the ultrasonics and the IMU sensor. These readings constantly change when the robot is moving as the sensors correspond to the external feedback. A log file keeps the raw data onboard for reference. There is also a '*Map*' button where the console can take the user into a map view. The Map View is a Google map that can mark the area where the robot should navigate. Figure 3.25 shows the Map View, specifically the *Tasik Universiti Malaya*, the lake selected to clean and perform sampling in this study.

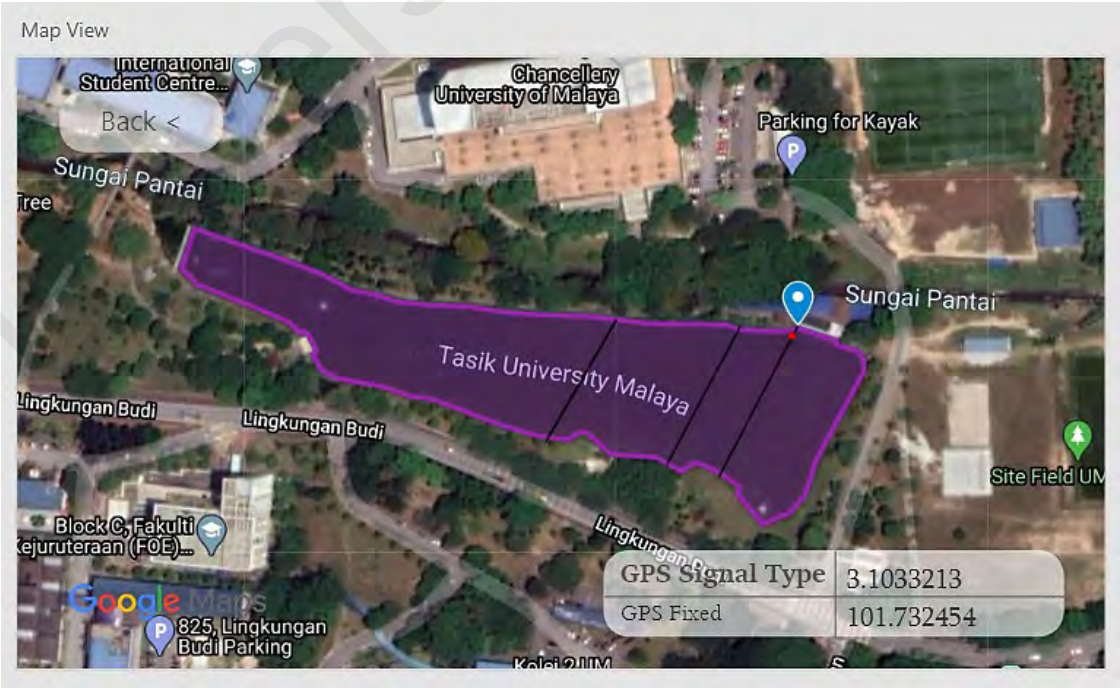

Figure 3.25: The Map View

The Map View contains several buttons. There is a button to switch back to the Main View. A square area at the bottom right of the console is constantly fetching the location and GPS data. A small red colour circle is used to show the live location of the robot on the map. The Google Map API is used as a framework to display a live map based on the desired location. The location can be searched online via the Google My Map or Google Earth app. A user can zoom in or out of the area shown on the Map View console.

In the Main View, pressing the '*Setting*' button allows the user to get into the app setting page, as shown in Figure 3.26. The setting page controls the communication between the Android console and the robot. The IP Address field refers to the IP address of the robot to communicate. The IPv4 format is used and by default is set to '192.168.100.225'. The second editable field refers to the camera port. This port is used to stream the camera, and it can be any positive integer number between 1 and 65535. In this study, the port used for the camera is fixed at '8081'. The control port field is the same as the camera port, but it directly connects to the robot. The control port in this study is fixed at '21567'. The last field is called the pin code and has the pin '1234'. The setting is confirmed as long as the save button is used.

The setting page also has settings for the Map View. KML is a file format used to display geographic data in an Earth browser such as Google Earth. For example, this study extracted the KML file for the *Tasik Universiti Malaya* and loaded it into the Map View console. The robot user can create KML files to pinpoint locations and add image overlays. This feature is handy when the user has waypoints to overlay onto the KML file. The '*Select Way Point*' function accepts the waypoint coordinates file.

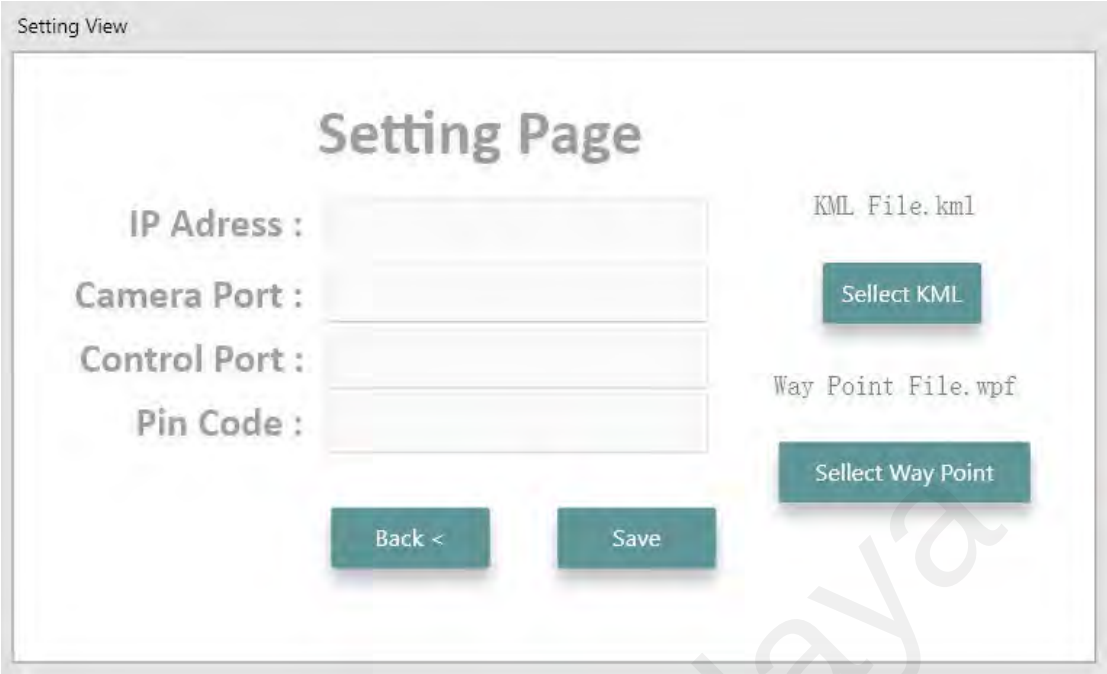

Figure 3.26: The Settings View

3.4.2 Robot control API

The Robot control API is the API designed to control the robot over the network. The API is available in two programming languages, Java and Python. The API allows the user to control and get feedback from the robot. The functions provided by the API are listed in Table 3.2. Figure 3.27 shows the API example codes in Python, while Figure 3.28 shows the example in Java. Table 3.3 shows the sensors name list used for requesting data from the robot.

```
import HyD
from sensors import Sensor

p = HyD.HyDrone(ipAddress=,portNumber=)
p.getSensorData(Sensor.ULTRASONIC1)

# it can be like this as well
s = Sensor
p.getSensorData(s.ULTRASONIC1)

# it is more efficient to read all the sensors when you need to read more than one sensor at the time to preven
p.getSensorData(s.ALL)
```

Figure 3.27: Robot Control API code in Python

```
void moveForward(int lsp, int rsp){
    sendMess("CONTROL,MOVE_FORWARD,"+lsp+","+rsp+"");
}

void moveBackward(int lsp, int rsp){
    sendMess("CONTROL,MOVE_BACKWARD," + lsp + "," + rsp + "");
}

void startMotors(){
    sendMess("CONTROL,MOTOR_ON");
}

void stopMotors() {
    sendMess("CONTROL,MOTOR_OFF");
}

void recordVideo() {
    sendMess("CONTROL,START_RECORDING");
}

void stopVideoRecording() {
    sendMess("CONTROL,STOP_RECORDING");
}
```

Figure 3.28: Robot Control API Java Sample Code

Table 3.2: Robot Control API Functions

The Function	**Parameters**	**Returns**
connectToRobot()	Null	Null
disconnectWithRobot()	Null	Null
startMotors()	Null	motor is on
stopMotors()	Null	motor is off
recordVideo()	Null	video recording started
stopVideoRecoding()	Null	video recording stop
moveForward (leftSpeed, rightSpeed)	Speed is an integer between 1000 to 1500	The robot is moving forward at (speed)
moveBackward (leftSpeed, rightSpeed)	Speed is an integer between 1000 to 1500	The robot is moving backwards at (speed)
moveRight(Speed)	Speed is an integer between 1000 to 1500	the robot is moving right (speed)
moveLeft(Speed)	Speed is an integer between 1000 to 1500	the robot is moving left (speed)
collectWaterSample (container)	The container in integer 1,2,3,4	water sampling started (container)
rollMove(distance, direction)	Distance from the encoder disk 96 is one cycle. The direction is either Up or Down.	roll moving to (distance) (direction)
getSensorData (sensorName)	Sensor name listed in table 3.3.	["SensorName","SensorValue"]
startDataRecord()	Null	Null
stopDataRecord()	Null	Null

Table 3.3: Sensor List used for getSensorData(sensorName)

Sensor List used for getSensorData(`sensorName`)
ULTRASONIC1
ULTRASONIC2
ULTRASONIC3
ULTRASONIC4
IMU_PITCH
IMU_ROLL
IMU_YAW
GPS_SIGNAL_TYPE
GPS_LATITUDE
GPS_LONGITUDE
GPS_ALTITUDE
GPS_TIME
BATTERY1_VOLTAGE
BATTERY2_VOLTAGE
COMPASS_BEARING
CPU_TEMPERATURE
WIFI_SIGNAL_STRENGTH
VIDEO_RECORDING_STATE
IMU_CALIBRATION_STATUS
DATA_RECORDING_STATUS
ALL_DATA
ALL_IMU

Getting sensor data is a strong message that shows each sensor's reading and the sensor's name—the ALL_DATA keyword request for all the sensor data. A comma separates each sensor data, and a colon separates each sensor data. The feedback message of "getSensorData(ALL_DATA)" function is shown in Figure 3.29.

```
C:\Users\user\Desktop\HyDroneAPI-master>python test.py
Received:ultraSonic1:    0.0, ultraSonic2:    0.0, ultraSonic3:    0.0, ultraSonic4:    0.0, IMU_PITCH: 5.83, IMU_ROLL:
-5.73, IMU_YAW: 8.61, GPS_TYPE: 1, GPS_LAT: 3.11401417, GPS_LONG: 101.66482467, GPS_ALT: 60.4, GPS_T: 1, BAT1_V: 12.6352
69597827083, BAT2_V: 4.269082282052064, MAG_BEAR: 8.61, CPU_T: 35.9, WIFI_S: 31 dBm, REC: 0,,,
```

Figure 3.29: Sample of the return message on getSensorData(ALL_DATA) function

3.4.3 PCF8574 API

The PCF8574 API has been designed and published in this study. The API is designed for PCF8574 to be used as an IO expansion board for the robot. In other words, this API helps the user to add more senses to the Raspberry Pi. The Python language has been used in developing this API. An example of the API code is shown in Figure 3.30.

```
import pcf8574_io

# you can use up to 8 PCF8574 boards 0x20 and 0x21 are the I2C addresses
# true will set all the pins HIGH +3.3v false will set them to LOW 0v
p1 = pcf8574_io.PCF(0x20)
p2 = pcf8574_io.PCF(0x21)

# p0 to p7 are the pins name
# INPUT or OUTPUT is the mode
p1.pin_mode("p0", "INPUT")
print(p1.digital_read("p0"))

# you can write and read the output pins
# use HIGH or LOW to set the pin HIGH is +3.3v LOW is 0v
p1.pin_mode("p4", "OUTPUT")
p1.digital_write("p4", "HIGH")
print(p1.digital_read("p4"))

# you can read and write up to 8 boards at the same time just make sure you ech board has a different
p2.pin_mode("p7", "OUTPUT")
p2.digital_write("p7", "LOW")
print(p2.digital_read("p7"))
```

Figure 3.30: PCF8574 API example code

3.5 Networking

Figuring a way to send and receive data from the controller to the robot and vice versa is essential to control the robot wirelessly. As a rule of thumb, any radio transmitter and receiver can communicate on the same frequency. The wireless network selected for this study is the 2.4Ghz WIFI, following the standard internet protocols. The TCP/IPv4 protocol helps control the robot anywhere long as the internet connection is established. This section covers the communication between the robot and the Android app, including all the services used for archiving and settings to boost the system's security level.

3.5.1 Robot network connections

The robot can be controlled in two ways, over either LAN or WAN networks. A Wi-Fi router or 4G LTE modem connects the robot with the external devices. Both device details are as follows:

a) Wi-Fi Router: The WIFI Router used is the WL-WN570HN2 model from WAVE LINK. This router model is a high-power Wi-Fi router that works on a 2.4Ghz frequency for outdoor long-range signal transmission. The wireless link rate up to 300Mbps supports WPA-PSK/ WPA2-PSK encryption. It has 2 x 7dBi Omni Directional Antennas, and It works with PoE power source DC 24v 0.5Amp. This Wi-Fi router can cover up to 500 meters in open areas with no obstacles to block the Wi-Fi signal.

b) 4G modem: The Huawei USB modem model (E3276) provides the robot with Internet over 3G and 4G networks. The device is a small USB device that can connect with the Raspberry Pi quickly. The device has two ports for an external antenna and has an SD card reader. It supports FDD / TDD / UMTS / HSUPA / HSPA+ / GSM / GPRS / EDGE networks, and a High-speed LTE packet data service of up to 150 / 50 Mbit/s.

The devices that need to be connected to the network are shown in Figure 3.31.

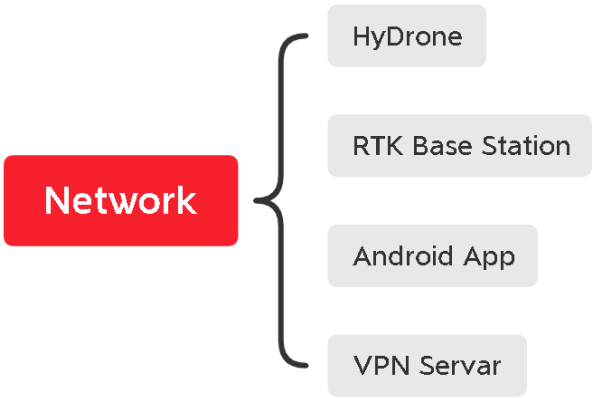

Figure 3.31: Robot devices connecting to the network

LAN network is a local area network that connects the external devices locally. In comparison, a WAN network connects external devices over the internet. VPN (a virtual private network) is used to create a local network over the internet. TCP/IPv4 allows connection to the robot with the external devices in many different topologies. The topologies used in this project are as follows:

a) Locally connected over mobile hotspot: Mobile hotspot is the simplest way to control the robot since no external router or modem is needed. Most Android devices have a Hotspot that shares Wi-Fi and makes the phone work as a Wi-Fi router. This topology is only used to control the robot within a maximum range of 10-meter. It does not require the internet, and it has an immaculate connection with a very low delay. The topology is shown in Figure 3.32.

Figure 3.32: Locally connected over mobile hotspot

b) Locally connected over Wi-Fi: This topology does not require an Internet connection. The Wi-Fi router needs to be set up at the lake, where the robot must cover. All the connected devices must be within the range of the Wi-Fi router. This topology does not need a VPN server since it is connected locally. The connected devices are shown in Figure 3.33.

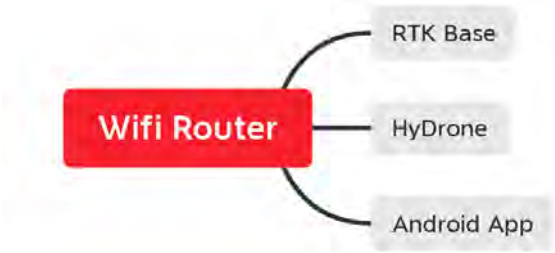

Figure 3.33: Locally connected over Wi-Fi

An IP address is used on each device to connect with the Wi-Fi router. The robot and the RTK Base station must have a static IP address assigned manually. The Android App is getting the IP address from the DHCP server of the Wi-Fi router. The robot uses three different ports to communicate with the external devices, a port for controlling the robot, another for streaming video from the robot camera, and the last one for getting RTCM data from the RTK base station. This topology has the minimum delay over the other topologies, making it the best way to control the robot since the low delay gives it very smooth control. The only problem is that it is hard to set up, and it needs a power source to provide the Wi-Fi router with power.

c) Internet-connected over Wi-Fi router: All the connected devices must be connected to the internet in this topology. The VPN server is used to connect all the devices. No configuration needs to be changed with VPN since VPN works the same way as the LAN network. Every external device needs to connect with the VPN server, and the network is good to go. A Wi-Fi router is used to connect to the internet. The robot can be connected from anywhere, and the RTK Base station could be fixed within 20km from the robot; this topology is shown in Figure 3.34.

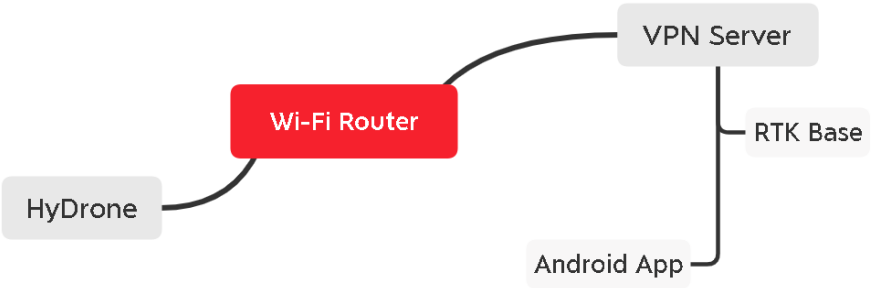

Figure 3.34: Internet-Connected over Wi-Fi Router

d) Internet-connected over a 4G modem: This topology connects the devices the same over VPN. The Wi-Fi router is replaced with a 4G modem connected to the internet. The good thing about this topology is, it does not need any external routers around the robot working area. The modem is placed on the robot, and it only needs a 4G LTE signal to work. The RTK Base station needs to be within 20km from the robot, and the android app can control the robot from anywhere using the internet, as shown in Figure 3.35.

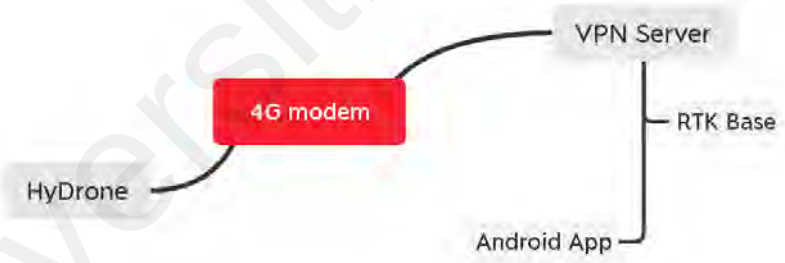

Figure 3.35: Internet-Connected over a 4G modem

3.5.2 Video broadcasting

MJPEG-Streamer library is used to stream the video from the camera on the robot to the controller. It is an enormous challenge to stream video from the Raspberry Pi without any delay. A lot of methods and codes have been tested for this purpose. Finally, the MJPEG-Streaming library is used to solve this issue. The library is built initially for a project called "CNCjs", and it is a full-featured web-based interface for CNC controllers. The streaming process starts by installing a few software and running a one-line code on Raspberry Pi. The library offers features like using a specific port for the streaming process, ‘8081’ by default. Accessing the video link is passed via a username and password. The video works on any JavaScript browser, keeping the streaming videos' original quality of 1080 pixels.

3.5.3 VPN server

OpenVPN software is used for setting up the VPN server. It is a free tool for running a VPN service. It is a small software that needs to be installed on a PC with Linux, Windows, or other operating systems. Configuring the server requires the creation of some keys. A text file with the extension ‘OVPN’ should be created if the keys are used to secure the connection. The OVPN file will contain the server configuration. A key and ‘OVPN file should be created for each user for security. Getting the OVPN keys can be done via the software. The Raspberry Pi or an Android device can install OpenVPN and run the configuration to generate the key. All software and apps available for this process are free for a lifetime. The VPN uses port 1194 to communicate and UDP protocol since it is faster than the TCP.

3.5.4 Network security

Balancing between performance and security is a challenge, but controlling a robot in the real world, is needed since the communication over TCP protocol transfer the messages in plain text. SSH has been used for securing the connection between the app and the robot on the LAN network. Another SSH has been used for the VPN since VPN typically uses SSH for communication, increasing robot's security on the LAN and WAN networks. A username and password also secure the camera streaming to gain more control over the robot's privacy.

3.6 Positioning system

GPS (global navigation system) is generally used to locate anything on the map of the world. GPS uses 24 satellites to cover the world. The United States of America owns it. The satellites broadcast signals to the earth. Any device with GPS can detect the signals and provide their position on the map. The position is provided to the user in latitude and longitude form. A device needs at least four satellites to get its position. The problem with GPS is accuracy. It has low accuracy. A smartphone with GPS is typically accurate within 4.9 meters (GPS, 2020a). This low accuracy makes it difficult for applications where accuracy is needed.

Another system that gives better accuracy than GPS is the global navigation satellite system (GNSS). GNSS is working the same way as GPS work, but it has better accuracy than GPS. Since the US owns GPS, other nations fielded their systems to provide positioning (GPS, 2020b). GNSS combines all positioning systems globally and uses its satellites to position the user. Example of positioning systems used by GNSS is BeiDou Navigation Satellite System (BDS) owned by the People's Republic of China, Galileo owned by Russia, Indian Regional Navigation Satellite System (IRNSS) / Navigation

Indian Constellation (NavIC) owned by India, and Quasi-Zenith Satellite System (QZSS) owned by Japan. The GNSS has better accuracy since it uses more satellite signals to calculate the position. Standard GNSS receivers are accurate within about two meters Galileo GNSS (2018), making them better than GPS. Another system used for positioning works is based on the GNSS system, called RTK. The RTK has better accuracy than GNSS, but it is more difficult to use than GPS and GNSS.

RTK stands for Real-Time Kinematics, a GNSS technique used to enhance the precision of position data received from satellite-based positioning systems. RTK uses two GNSS devices that have RTK capability to calculate an accurate position. One device works as a base station, and the other works as Rover. Together they can provide a very accurate position up to 1-3 centimetres of accuracy. The RTK Base station is a GNSS receiver that receives the signal from multiple sources, and it should have a known fixed location. The Base Station uses its fixed location and the satellite signal to calculate an error rate. The error rate will be sent to the Rover to calculate its location. The RTK Base Station should be fixed within 20km of the Rover for better accuracy.

This study has developed a separate system to work as the RTK Base Station. The system allows change to the base station's location quickly. It provides the base station with battery power to work remotely and provides a communication medium for sending out the correction messages. The system contains SimpleRTK2B, a Raspberry Pi model Zero WH for transferring the RTCM messages, a 5v 5A power supply, a 2000mah Li-po battery, and a 12v fan cool down the system. A switch to turn on and off the system is also included in the design shown in Figure 3.36. All the electronic components are attached to a tripod that makes it easy to move the station from one location to another (see Figure 3.37).

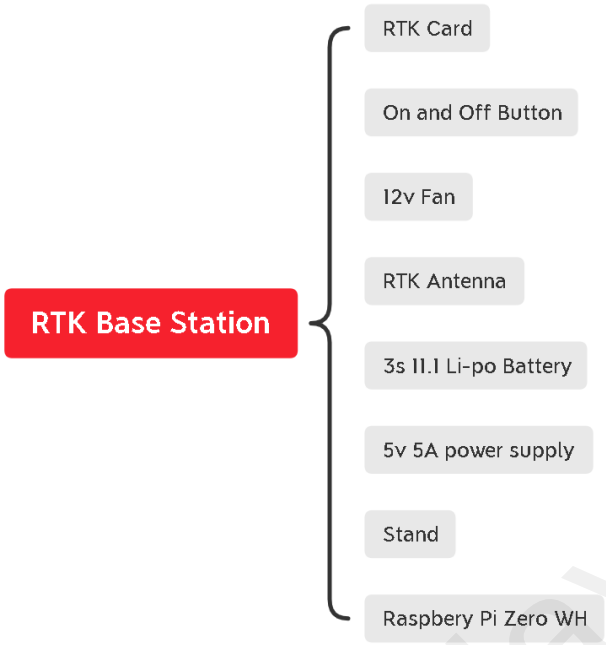

Figure 3.36: The RTK base station system components

Figure 3.37: RTK base station system

The RTK Rover is the device that gets an accurate position while moving. The RTK Rover receives RTCM messages from the RTK Base Station containing error rates. The Rover will then calculate the accurate location using the error rate and its location from the GNSS satellites. The RTK Rover is attached to the robot, and it uses the Raspberry Pi 3b plus on the robot to get its correction data from the Base Station.

The RTK purchased for this study is two SimpleRTK2B boards with two U-Blox GNSS Multiband antenna ANN-MB-00 (IP67). Applying the RTK to a robot is quite costly usually, but recently the U-Box company released a powerful chip that lowered the system's price to almost 10 per cent of before. The SimpleRTK2B and U-Blox antenna is shown in Figure 3.38. The SimpleRTK2B is a standalone application board that evaluates multi-band GNSS technology, including the RTK functionality. It is based on the u-Blox ZED-F9P chip, and it is fully compatible as a shield with Arduino, STM32 Nucleo, Raspberry Pi.

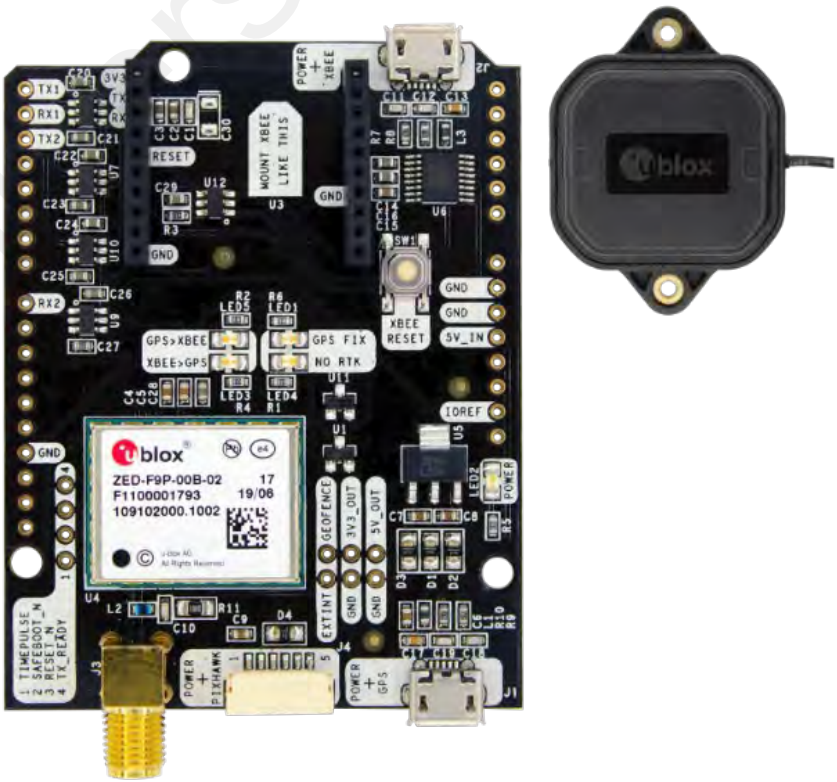

Figure 3.38: simpleRTK2B and U-Blox antenna

The SimpleRTK2B can be configured in three ways; each gives a different accuracy level. All the possible ways of configuration are listed below with more details:

- Standalone: the most straightforward way to reach 1-meter accuracy without range limitations. The most basic configuration is to use simpleRTK2B as a standalone GNSS application board.
- Base – rover configuration: gives <3cm accuracy in theory with a maximum range of 20km from the base station. Two simpleRTK2B are used and configured as a base station and rover. The base must communicate with the rover to send the RTCM correction messages to the rover. Xbee radio model needs to be used on both base and station to communicate within a few kilometres based on the radio model's range.
- Base – rover NTRIP configuration: gives <3cm accuracy in theory with a range of 20km from the base station. Two simpleRTK2B are used as a base station and a rover. Here, the NTRIP protocol transfers RTCM correction data from the base to the rover. The rover should stay within a 20km radius for accuracy. However, the accuracy will lower when the rover exceeds the range.

Central to robot networking is a base-rover configuration that shows the base connecting to the same network as the robot. There are two RTK base configurations, either portable or fixed mode. After changing its location, the user must configure the base station's location in the fixed mode. The portable mode will automatically get the fixed location after booting the system and waiting for a few minutes. After the base station gets its location, it calculates the correction data. Sending out the correction data in RTCM-formatted messages is achieved via a free opensource software called RTKLIB, which works on the Raspberry Pi. Similarly, the rover uses RTKLIB to receive the RTCM messages and calculate the correct location.

3.7 Chapter summary

This chapter presents the design and development of a water surface robot system comprising the robot structure, electronics, software, networking and positioning. This chapter shows how the robot is constructed from scratch, beginning with considerations for buoyancy then balancing load and power. Next, the chapter discusses the mechanism for sampling on the water surface and underwater. Then, a comprehensive presentation on the software design and implementation provides an interface between a user and the robot. The chapter is completed with a dynamic networking environment to allow off-board processing of robot data, mainly the live streaming of the robot's camera and the robot's current location on the lake. Making these data available off-board for further processing is massive for a low-cost robot such as the robot developed in this study. Off-board processing allows automation to be adopted on low-cost robots.

The following section shows two case studies. One takes advantage of the live off-board visual feed to run deep learning to detect the robot's garbage. The other leverages the robot's live coordinates as it moves at the lake to run a genetic algorithm in recommending automated waypoints for sampling. The robot system performances for the two case studies underline the robot's performance in this study. Without reliable design and successful construction for robot structure and electronics and an open design for the software, networking and positioning, the automation features are less likely experimented with and achieved on a low-cost robot. The following chapter presents and discusses the results.

CHAPTER 4: RESULTS AND DISCUSSION

4.1 Overview

This chapter presents the results and a discussion on the robot system performances. Section 4.2 to 4.6 discusses testing the robot structure, software, networking, sensors, and positioning system. These results address objectives one and two of this dissertation. Section 4.6 and 4.7 cover two case studies to complete the chapter. The case studies aim to evaluate the open robotic system design in supporting non-robotic researchers implementing advanced algorithms on a low-cost robot. For example, the genetic algorithm (GA) automates wayfinding, and deep learning detects and tracks floating garbage. The case studies address objective three of this dissertation.

4.2 Experimental Setup

The early field testings were conducted at a mini pond and outdoor pool. Once the robot achieves floating stability or buoyancy, the field testing shifted to the UM varsity lake with special permission by the UM Deputy Vice-Chancellor of Research and Development. The *Tasek Varsiti* is an artificial lake built even before the Universiti Malaya began. The estimated size is about 90 meters at the largest width and about 270 meters in length. A kayak station at Tasek Varsiti with a proper landing releases the kayak into the lake. The landing has a stable grounding, complete with steps easing the 17kg robot release into the lake.

Figure 4.1 shows the largest width area at Tasek Varsiti, where the experiments are conducted. Selecting the area ensures the robot is always within the testers' point of view since the testers are located at the kayak station. Keeping the robot within sight is crucial because the robot is not equipped with an obstacle avoidance algorithm, and the lake has an ecosystem with groups of ducks and geese swimming around.

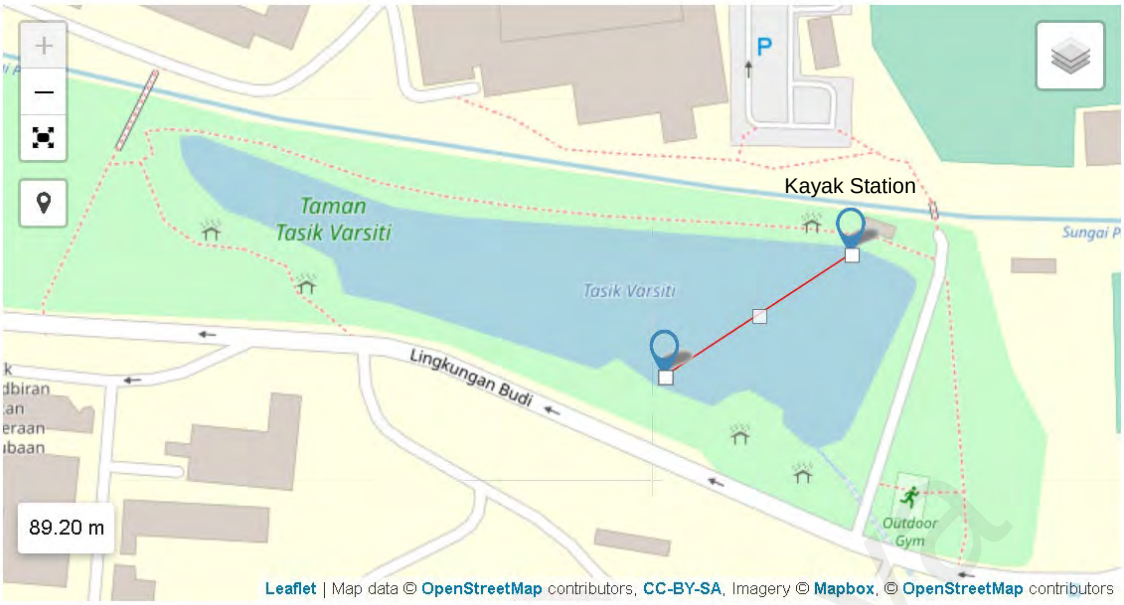

Figure 4.1: Tasek Varsiti 90-meter width as the field test location

4.3 Robot structure, material, propulsion and power system

The hull structure impacts the robot's performance significantly. The catamaran-style, i.e., twin hulls in parallel, offers the robot stability, safety against sinking, broad deck, simplicity of steering and redundancy in hull buoyancy. The robot developed did not have rolling motions or huge drag, which usually happens in monohull structures. The PVC material selected for the hull is cheap and sufficient in preventing corrosion. Other advantages observed include high strength ratio to weight, lightweight transportation and installation, and high fatigue resistance. The nylon plastic fishing net used is durable, easy to fasten and remove using the nylon cable tie. There is ample space between the hulls for the net to trap garbage. Figure 4.2 shows the early day testings when buoyancy is yet achieved, and Figure 4.3 shows the robot cruising on the lake with adequate load balancing. Figure 4.4 shows fastening the fishing net to trap garbage.

Figure 4.2: Early day testings when buoyancy not achieved

Figure 4.3: Robot gliding and achieving load balancing

Figure 4.4: Fastening the net to trap garbage

Two rechargeable lead-acid batteries power the robot. Unlike profane, they are environmental-friendly and last long, supplying power to all components on the robot, including propeller driving. The capacity of the batteries remained around 10% even after 1-hour driving has been observed on separate occasions. The disadvantage of lead-acid batteries is their size and weight. However, the load is not a problem for the water surface robot; thus, lead-acid batteries are a great option. The battery pack installation extends the robot's endurance capability and period of operation. Overall, these rechargeable lead-acid batteries show high energy efficiency, without memory effects, long cycle life, low discharge rate and low carbon footprint.

The robot is equipped with two brushless motors for propulsion. The thrust is controlled in the forward and backward directions. For example, steering to the left requires the right propeller to rotate forward while the left propeller rotates backwards. Speed adjustment is essential when steering or changing direction. The speed can be higher when driving forward but lower when performing steering. The robot has nine-speed settings. Speed nine can reach up to 4km/hour and is only suitable for long forward driving. The comfortable speed for remote driving in the experiment is setting number five and six. When turning is needed, the recommended speed is between two and three. Speed one is comfortable for docking and sharp turning.

The system's temperature remains within the acceptable range even after driving the robot under the direct sun; the maximum observed is 47 degrees. Figure 4.5 shows the live temperature of the system taken at an experiment by an outdoor pool.

Figure 4.5: Live system temperature from the Android app main view

4.4 The robot software system

Deployment of the water surface robot in remote operation demands high assurance that the robotic software and interfacing operate. Software development has to perform predictably well at the lake which the physical environment may be uncertain. Several factors determine whether the robotic software supports the robot in acting efficiently and responding while offloading the processing off-board. The measuring factors include distributed processing, modularity, cross-platform, code aesthetics, design choices, message-passing topology, and data flow.

4.4.1 Distributed processing

The software application runs on a single robot that performs both hard and soft real-time constraints. The software also computes, in parallel, tasks to support lake sanitation and sampling activities. Raspberry Pi-centric's onboard computational resources cannot sustain the lake-related tasks computation; thus, the computing resources are distributed and shared off-board. Testing the distributed processing include driving the robot over 20-meter, 40-meter and 60-meter while sending the live streaming to a Raspberry Pi streamer-friendly application. One example is the MJPG-Streamer that takes JPGs from the robot’s Linux-UVC compatible webcams and streams them as M-JPEG via HTTP to the web browser. The user may drive the robot while sending a live stream to the streamer application for viewing, storage and further processing.

4.4.2 Modularity

An open design robotic system aims to encourage various researchers to contribute their expertise in specific areas of the study. Getting many researchers involved may create a large codebase, so isolating each software component is crucial. Keeping the software modular has reduced debugging time for this study and speeds up the verification processes. Figure 4.6 shows the modular software components for the robotic system. There are eight major modules for the software system, and one is categorized for other electronic features of the robot system. Each of these modules has its methods and subroutines. Each subroutine is considered the smallest unit of code in the robot system.

Several testing is done to ensure modularity of the robot software system, beginning with basic functionality testing. Every screen's front-end buttons are first checked to see if they work and respond in real-time. Second, another pair of eyes do a code review for every module to check for bugs, which were plenty initially. The bugs are fixed following reviewers' suggestions. The static code analysis examines the source codes before executing the program to ensure the program serves the desired task. The third step is unit testing. Unit testing involves testing all robot functions and subroutines such as the pump control, speed control, map upload, camera feed, and RTK position.

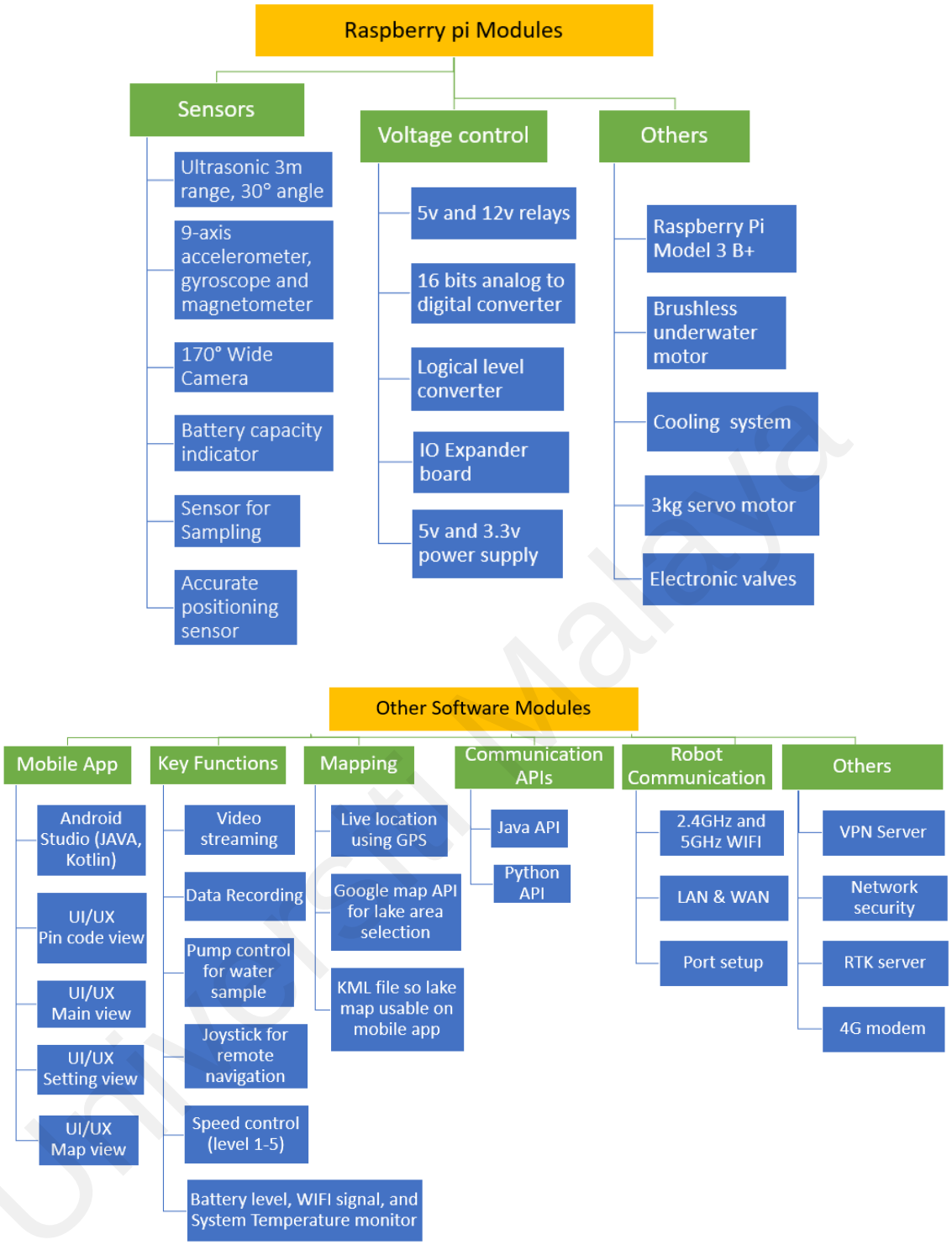

Figure 4.6: Modular software design improved debugging

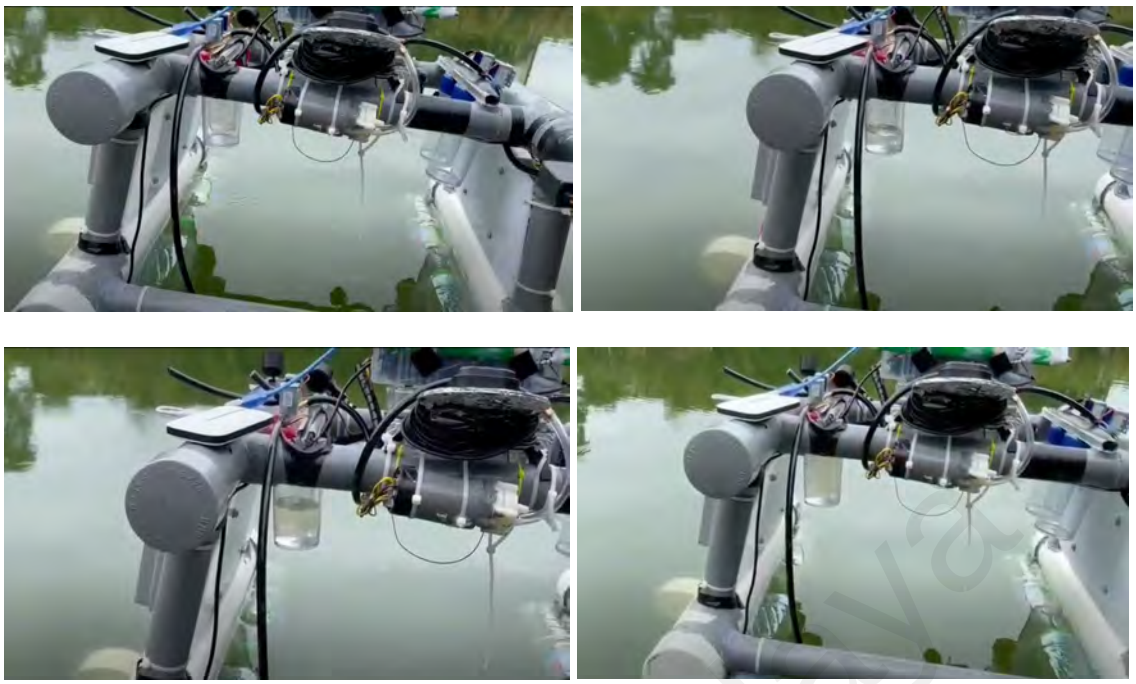

Figure 4.7: Surface sampling conducted at the UM lake

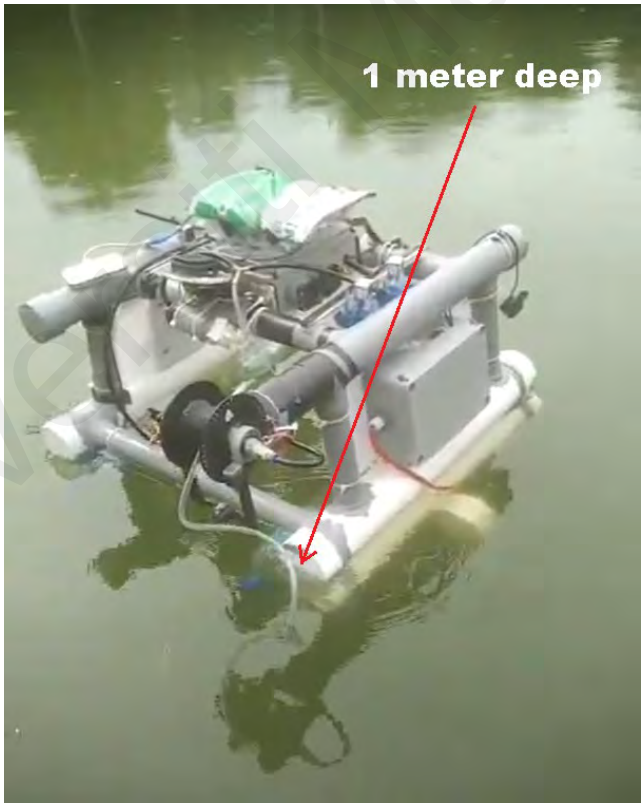

Figure 4.8: Underwater sampling (1-meter depth) conducted at the UM lake

The following are observed when performance testing is conducted. UI/UX buttons respond when pressed so the user can control the propellers remotely. Forward driving and steering show the accurate rotation of the propeller, and the speed control responds accordingly. Since the software is modular, users can kick start live streaming anytime they wish. The user can stop the propeller from rotating so the robot can hover somewhere on the lake for sampling. The pump works the water containers in serial order. Only when one container is full, then another container can begin collecting. The time to fill the 200ml container takes 22-seconds for the surface sample (see Figure 4.7, left-side container). The underwater sample takes about 30-seconds (see Figure 4.8, right-side container).

Additionally, the sensor buttons generate log files when activated. Figure 4.9 shows an example of a log file saved in an experiment at Tasek Varsiti on 2020-12-14 at 4.30 PM. The log files can be retrieved from the Raspberry Pi external flash drive (SD card). Viewing live sensors reading is done via a command prompt in a Linux workstation, where users can scroll for history for a particular entry. The mobile-based platform shows the reading next to the sensor field (see Figure 4.10). The Main View of the mobile app only displays a reading of the instance without access to history. In the experiment, the log files saved show they read the correct sensor type and reported the data in the correct format.

```
Now: 2020-12-14 16:30:08.219062, uS1:   37.0, uS2:   38.4, uS3:  127.3, uS4:  129.6, IMU_P: 1.25, IMU_R: 4.125,
IMU_Y: 367.9375, GPS_TYPE: 1, GPS_LAT: 3.10203067, GPS_LON: 101.41188617, GPS_ALT: 13.9, GPS_T: 1, B1_V:
12.370377513962218, B2_V: 4.830147404400769, MAG_B: 367.9375, CPU_T: 44.0, WIFI_S: 0, V_REC: 0, CL: 0x3x0x0,
D_REC: True, W_TEMP: 30.0,,,Now: 2020-12-14 16:30:09.497146, uS1:   37.0, uS2:   37.1, uS3:  107.9, uS4:  120.4,
IMU_P: 1.25, IMU_R: 4.125, IMU_Y: 367.9375, GPS_TYPE: 1, GPS_LAT: 3.10203150, GPS_LON: 101.41188517, GPS_ALT:
13.2, GPS_T: 1, B1_V: 12.370377513962218, B2_V: 4.830147404400769, MAG_B: 367.9375, CPU_T: 42.9, WIFI_S: 0, V_REC:
0, CL: 0x3x0x0, D_REC: True, W_TEMP: 30.0,,,Now: 2020-12-14 16:30:10.651658, uS1:   37.0, uS2:   37.5, uS3:
120.1, uS4:  130.8, IMU_P: 1.25, IMU_R: 4.125, IMU_Y: 367.9375, GPS_TYPE: 1, GPS_LAT: 3.10203233, GPS_LON:
101.41188500, GPS_ALT: 13.4, GPS_T: 1, B1_V: 12.370377513962218, B2_V: 4.830147404400769, MAG_B: 367.9375, CPU_T:
43.5, WIFI_S: 0, V_REC: 0, CL: 2x3x0x0, D_REC: True, W_TEMP: 30.0,,,Now: 2020-12-14 16:30:11.825475, uS1:   37.0,
uS2:   37.4, uS3:   98.5, uS4:  112.4, IMU_P: None, IMU_R: None, IMU_Y: None, GPS_TYPE: 1, GPS_LAT: 3.10203400,
GPS_LON: 101.41188500, GPS_ALT: 13.2, GPS_T: 1, B1_V: 12.360377208777123, B2_V: 4.79014618366039, MAG_B: None,
CPU_T: 43.5, WIFI_S: 0, V_REC: 0, CL: 0x3x0x0, D_REC: True, W_TEMP: 29.9375,,,Now: 2020-12-14 16:30:13.042001,
uS1:   37.0, uS2:   37.5, uS3:  135.6, uS4:  117.4, IMU_P: None, IMU_R: None, IMU_Y: None, GPS_TYPE: 1, GPS_LAT:
3.10203550, GPS_LON: 101.41188383, GPS_ALT: 12.5, GPS_T: 1, B1_V: 12.360377208777123, B2_V: 4.79014618366039,
MAG_B: None, CPU_T: 44.0, WIFI_S: 0, V_REC: 0, CL: 2x3x0x0, D_REC: True, W_TEMP: 30.0,,,
```

Figure 4.9: Sensor reading live view on a Linux workstation

Figure 4.10: Sensor reading live view on a mobile app

4.4.3 Cross-platform support

The Raspberry Pi is the primary computational resource for the robot system. The Raspberry Pi controls all devices connected to the robot, including sensors, mechanical pumps, propellers, and others. Python and C++ write the source codes for the Raspberry Pi. The Python and C++ languages are compatible with Windows and Linux operating systems; however, the sensors used on the robot work explicitly in a Linux environment. Therefore, the primary computational resource in this study is a Linux-based workstation, where all the backend processing is handled. The user can interface with the robot via two channels, a mobile operating system and a web-based app. The JAVA via Android Studio and Kotlin performed the front-end's heavy lifting.

In the experiment, users concerned with backend processing can log in to the robot via a Linux workstation to monitor live sensor reading, positioning, and robot states and interact via command prompt and the keyboard. See figure 4.11 for an example of the robot state view on the Linux workstation. Users on-site may have a better experience with the mobile app when driving the robot remotely. A tablet or mobile phone is lightweight and more comfortable to carry when monitoring the robot's movement on-site. The touchscreen feature combined with a friendly interface help users to select functions using buttons and control robot steering and speed using slider bars. Figure 4.12 shows a user controlling the robot through a first-person view using a mobile phone.

```
C:\Users\user\Desktop\HyDroneAPI-master>python test.py
Received:ultraSonic1:    0.0, ultraSonic2:    0.0, ultraSonic3:    0.0, ultraSonic4:    0.0, IMU_PITCH: 5.83, IMU_ROLL:
-5.73, IMU_YAW: 8.61, GPS_TYPE: 1, GPS_LAT: 3.11401417, GPS_LONG: 101.66482467, GPS_ALT: 60.4, GPS_T: 1, BAT1_V: 12.6352
69597827083, BAT2_V: 4.269082282052064, MAG_BEAR: 8.61, CPU_T: 35.9, WIFI_S: 31 dBm, REC: 0,,,
```

Figure 4.11: One of the many robot states viewed with a Linux workstation

Figure 4.12: Driving robot remotely using the Android mobile app

4.4.4 Code aesthetics and design choices

As with any large software project, keeping code clean and streamlined makes research progress on the robot significantly easier. From an aesthetics perspective, the library is in C, C++ and Python classes which each module extends to provide the required functionality. Networking, routing, and scheduling code do not show up in the software modules, as the superclasses provide them. Using the superclasses allows most modules to have very little boilerplate code. Figure 4.13 shows the final design choices for the superclasses. Researchers contributing algorithms for the case studies in Sections 4.7 and 4.8 use them to complete their implementation requirements.

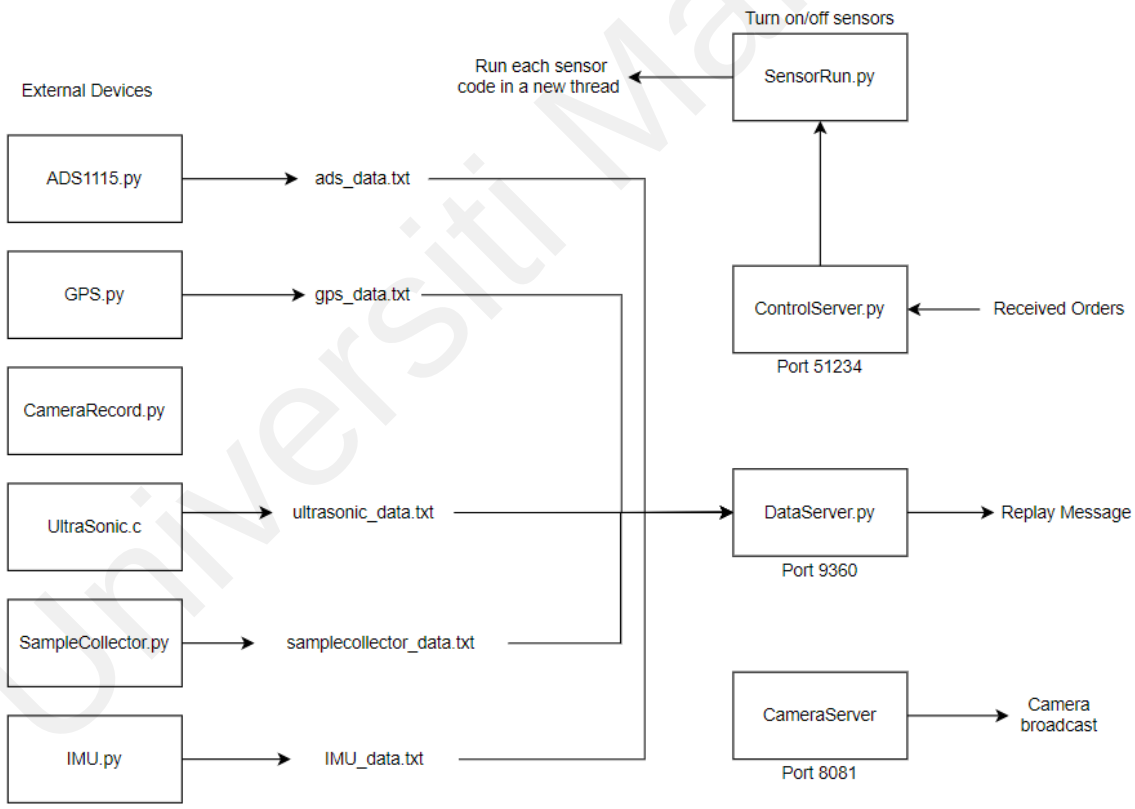

Figure 4.13: Superclass design for the robotic system

4.4.5 Digital I/O on Raspberry Pi

The robot connects to many devices, but the Raspberry Pi GPIO pin set is 40 total by default. The pins are insufficient to cover all devices and sensors the robot is connecting. Therefore, the Raspberry Pi requires an expansion board. The expansion board of the Raspberry Pi in the market is not cheap compared to the Arduino expansion board. Furthermore, an Arduino expansion board can reach 64 additional I/O pins, compared to 40 I/O pins of the Raspberry Pi. The only requirement is writing a PCF8574 API that works for the robot and sacrificing two pins on the Raspberry Pi for the I2C of the expansion pins. There are PCF8574 APIs available online, but none works for the Raspberry Pi robot developed.

One PCF8574 API is published on GitHub, and everyone can use it. The GitHub repository link is (https://github.com/ahmed9378/pcf8574_io), and it has also been published as a Python library. The user can use this pip3 command "pip3 install pcf8574-io" to download the API for any Python project. The PCF8574 API codes have been viewed 171 times between publication time on June 12th, 2021 and June 25th, 2021 (see Figure 4.14). The API is gaining attention from other Raspberry Pi users, and it is encouraging to note that some are working in robotics.

Figure 4.14: PCF8574 API gaining traction on GitHub

4.5 Message-passing topology and networking

The Raspberry Pi (model 3B+) is reliable as the central hub for message-passing for the robot system. The Raspberry Pi manages messages to IMU, I/O extender, and RTK via the I2C and UART protocols. Communication with other sensors is achieved using the GPIO connector. The Raspberry Pi shows good performance in exchanging messages with devices and applications over the internet using the TCP connection. Figure 4.15 shows the Raspberry Pi message-passing topology that gives the best communication performance for the robot system.

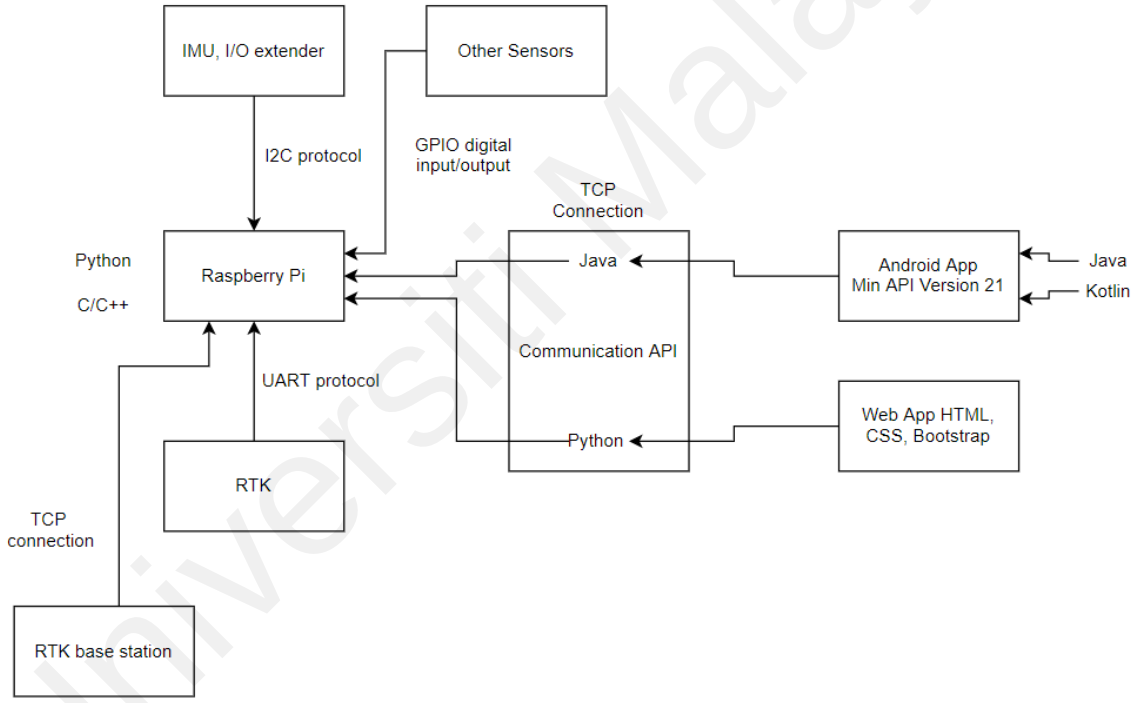

Figure 4.15: Raspberry Pi efficient message-passing topology

The robot network designed successfully connects to all the devices and mobile apps while testing the robot at the lake. The robot relies heavily on the VPN server through a 4G LTE modem, with the RTK base station located about 8km away from the robot. A mobile 4G data is added to the controller device to allow an internet connection so the app can scan the network and detect the active robot. The app is relatively smooth with low latency except for lag due to a drop in network speed. However, the user can

overcome this issue by turning off the live video streaming. With live camera streaming and active robot control, the maximum data load is 3 Mbps, while around 25 Kbps without the camera live feed. The VPN as the server for the robot is an influential concept as it allows users to connect to the robot without having to be on-site. The furthest a user has controlled the robot is Kurdistan, an area in Iraq, while the robot functions at UM lake.

4.6 Robot sensors

An experiment to test sensors performance have been conducted. The experiment lasts over five minutes focusing on Raspberry Pi internal temperature, the outdoor temperature sensor, battery voltage level, and the IMU. Figure 4.16 shows the Raspberry pi internal temperature ranges between 43.5 and 45.1 degrees Celsius. Figure 4.17 shows the outdoor temperature sensor that ranges between 30.56 and 30.75 degrees Celsius. Figure 4.18 shows the battery level sensor fluctuates between 11.61 and 11.69 volts. The last sensor to test was the IMU sensor. The IMU sensor has reasonable accuracy with angle read ranging between 0 and 359.9 degrees, when turning the robot to face toward north, south, east, and west. The sensor provided stable Yaw data, as shown in Figure 4.19.

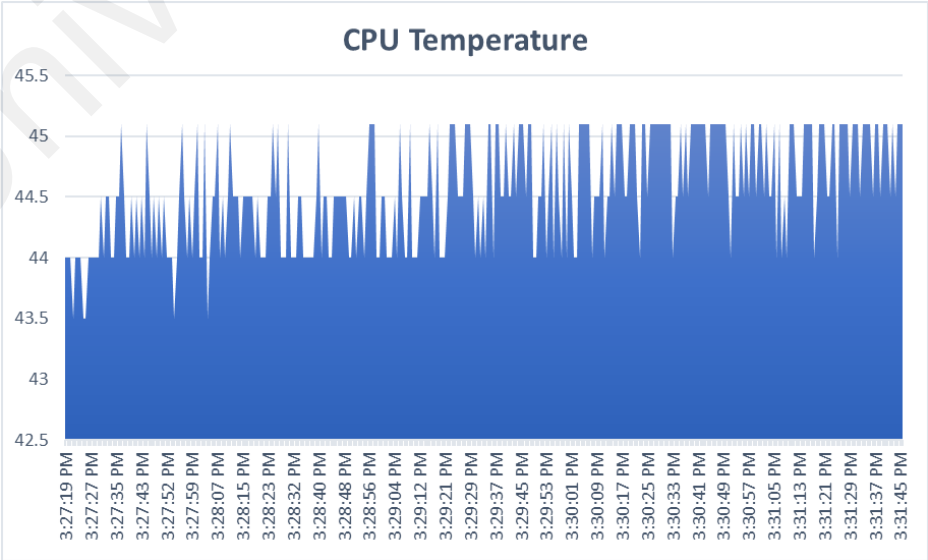

Figure 4.16: CPU temperature for 5 minutes testing

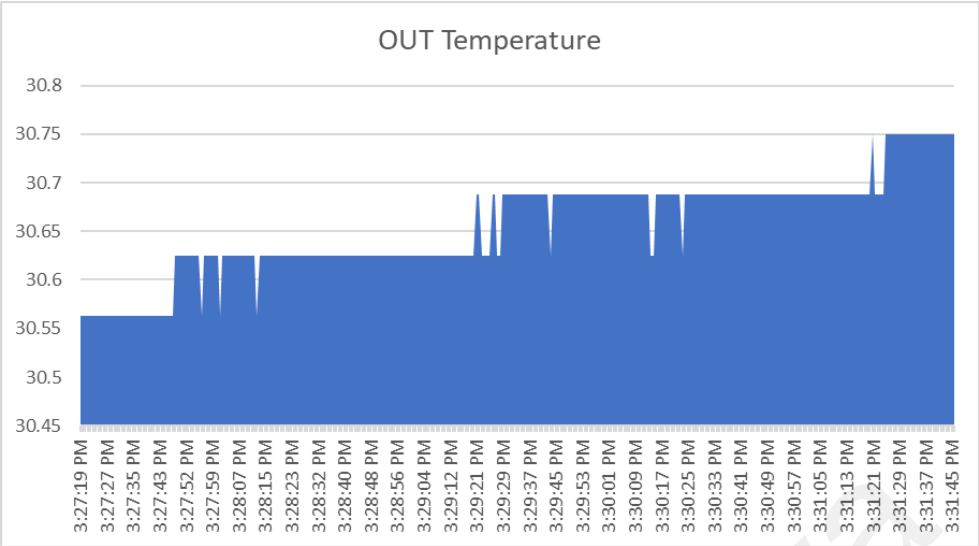

Figure 4.17: Outdoor temperature

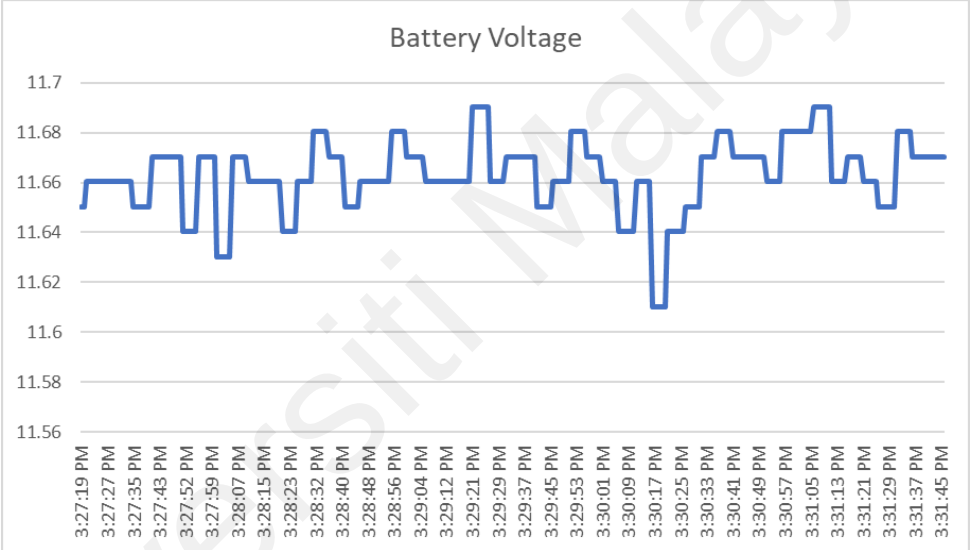

Figure 4.18: Battery voltage sensor record for 5 minutes

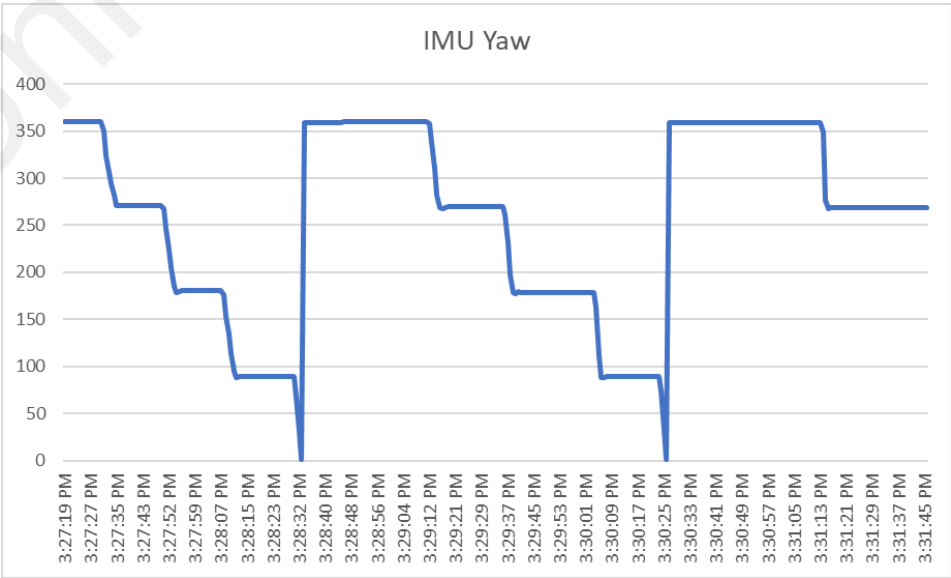

Figure 4.19: IMU sensor Yaw data

4.7 RTK positioning system

The RTK device has been tested in two forms: stand-alone and a base-rover connection. The system's accuracy is about 1-meter. The RTK device has been responsive and took around 60 seconds to be in static GPS mode every time the system starts. With the base-rover connection mode, the RTK could get up to 50 centimetres of accuracy. A 50-centimetre error in positioning is acceptable for a robot gliding at an average of 2km/hour. The performance of the RTK has been tested using the U-Center software. The weather was the main issue with the RTK. The system losses performance when the sky is cloudy.

The results for testing the RTK are shown in Figure 4.20. X-axes represent the received messages' index for 5600 messages recorded during the test. Y-axes represent the accuracy in meters; the value ranges between 0.18 to 0.22 meters. The duration of the test was 12 minutes. The RTK provided accurate data while the robot stopped and during motion. Figure 4.21 shows the robot's speed measured by the RTK system during the same test. The speed ranges between 0 to 1.2 meters per second. The results show that the accuracy drops while the robot stops and goes higher in motion.

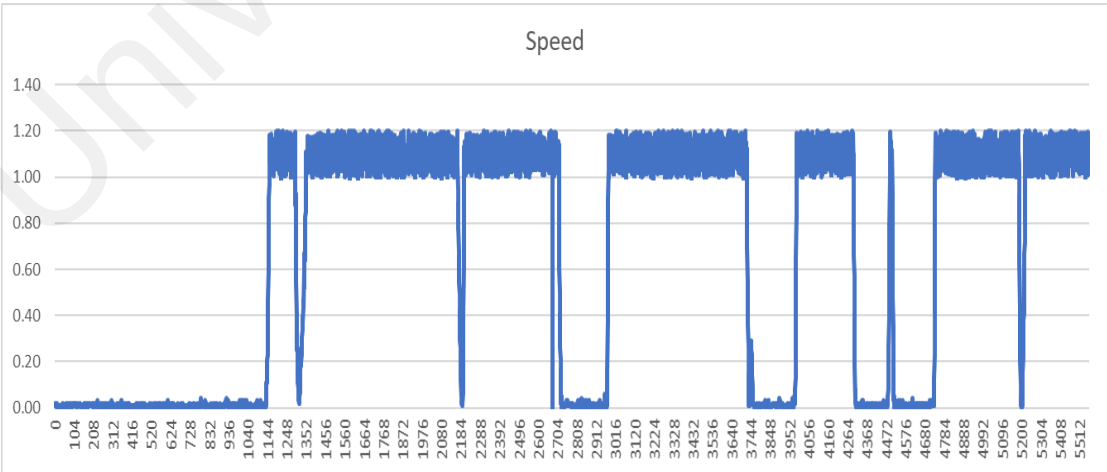

Figure 4.20: The robot speed in meters per second recorded from the RTK system.

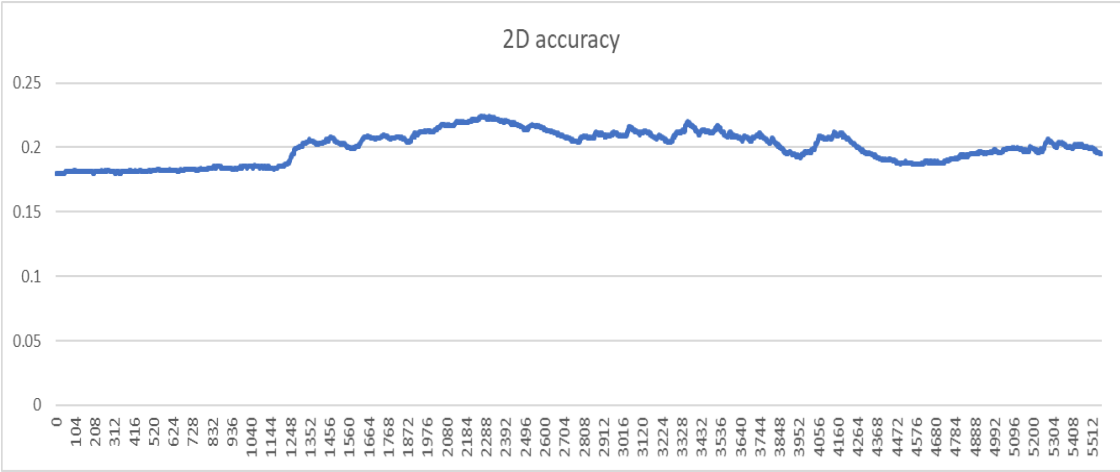

Figure 4.21: The accuracy of the RTK system in meters.

4.8 Case Study 1 – SLAM for water surface robot

Remote sensing robots that can glide on the water are practical for sampling. But remote access would still require humans to supervise. For an autonomous solution, path planning becomes the key. The first case study hypothesizes that an autonomous path planning solution like simultaneous localization and mapping (SLAM) can track where the robot's location is and where the robot is going. The SLAM can use sampling stations as waypoint landmarks. A split algorithm can extract the sampling stations by dividing a region of interest on the map. The centroid of each sub-regions can be the point to sample water. The genetic algorithm can arrange these centroids for optimal navigation.

Figure 4.22 shows the KML map splitting into equal-sized sub-regions, X-axes represent the longitude of the lake, and Y-axes is the latitude. Their values a subtracted value and is shown in Figure 4.22. The centroid for each sub-region can act as a sampling station, i.e., the point where the robot performs water collection. See centroid coordinate extraction in Figure 4.23. Optimal navigation conserves power and sampling time. Figure 4.24 shows how the genetic algorithm decides on a start point and follows an optimised trajectory for the water surface robot.

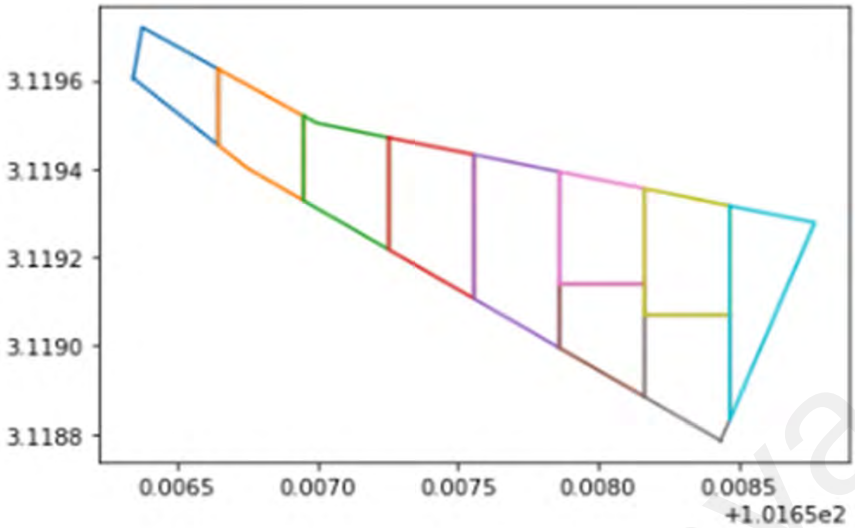

Figure 4.22: Dividing map into equal-sized sub-regions using the split algorithm

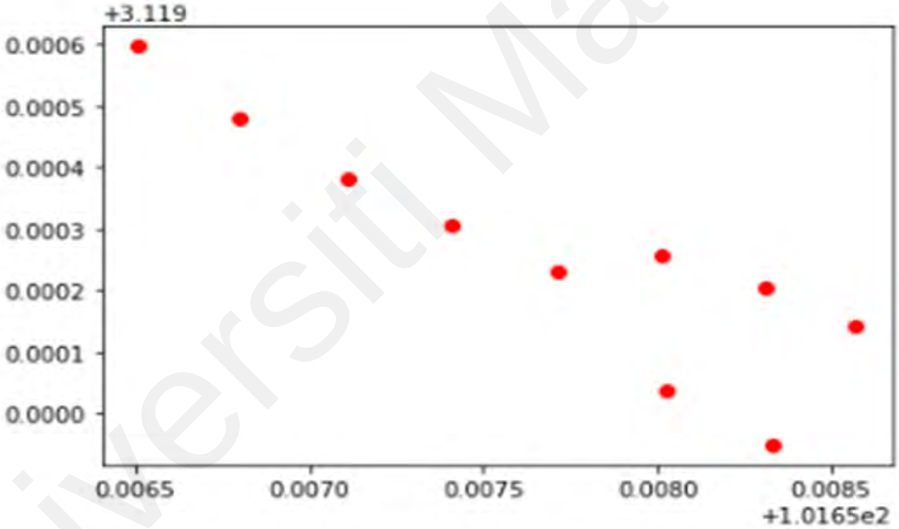

Figure 4.23: Extract centroid and its coordinate as SLAM landmarks

Figure 4.24: Order centroids for optimal navigation using the genetic algorithm

4.8.1 System specifications

The design and development for case study 1 are not within the scope of this dissertation. Another student with an AI background completes it. The student does not have robotic know-how. The full implementation is reported elsewhere (Zahiruddin, 2020). The specification for this subsystem is as follows:

Objective

1. To develop an optimal path planning system by extracting sampling stations as waypoints using the split and genetic algorithm.
2. To implement the SLAM method for autonomous navigation
3. To test the performance of the proposed system on the water surface robot

Functional Requirements

1. Allow user to upload map file.
2. Display map, date, and time.
3. Allow users to choose the size of the sampling station.
4. Display waypoints in the mapping and their order number.

Non-Functional Requirements

1. Parse KML file up to 10 KB.
2. Convert KML file into coordinates.
3. Split map into equal size sub-regions.
4. Label centroids and order them.
5. Update centroid order.
6. Localise with <50 cm error.

4.8.2 Results and analysis

Case study 1 shows that the APIs from this study can be used to perform tasks such as:

1. Connect to the robot remotely and fully.
2. Divide KML file for mapping.
3. Split the KML map into sampling stations using a split algorithm (off-board extension).
4. Extract best waypoints/routes to all sampling stations using the genetic algorithm (off-board extension).
5. Drive the robot following the waypoints generated in (3).
6. Update the map with the current robot location.
7. Use extensions to develop a website to control the robot remotely.

The student proposed the codes for steps (3) and (4) while the robot APIs handles the remote robot connection, KML map download and driving mechanism. The RTK positioning is crucial in robot localization and map updates. The RTKLIB is continuously called to fetch the robot's position. The map updating has higher accuracy when the robot experiment is conducted in bright and sunny weather.

Figure 4.25 shows the map fetched using the robot's live map request function. The map is generated using Google Map based on the longitude and latitude given by the RTKLIB. Figure 4.26 shows the KML map with four stations selected for sampling. The kayak base is the starting point in this experiment. The robot is instructed to navigate to four different stations following the trajectory set by the genetic algorithm (see arrow in Figure 4.26). The live system updates the robot's location every 8-seconds following the SLAM algorithm. A virtual marker marks the robot's position in the live system as a reference (see Figure 4.27).

Figure 4.25: Live map request fulfilled by the RTKLIB

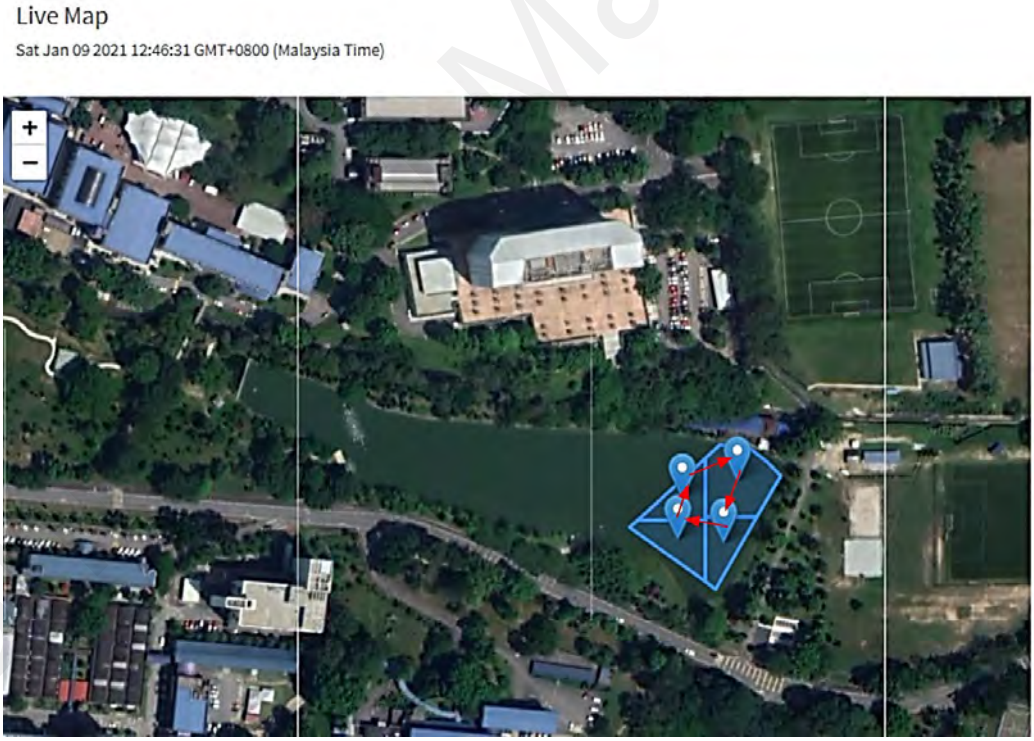

Figure 4.26: Four sampling stations selected in the experiment

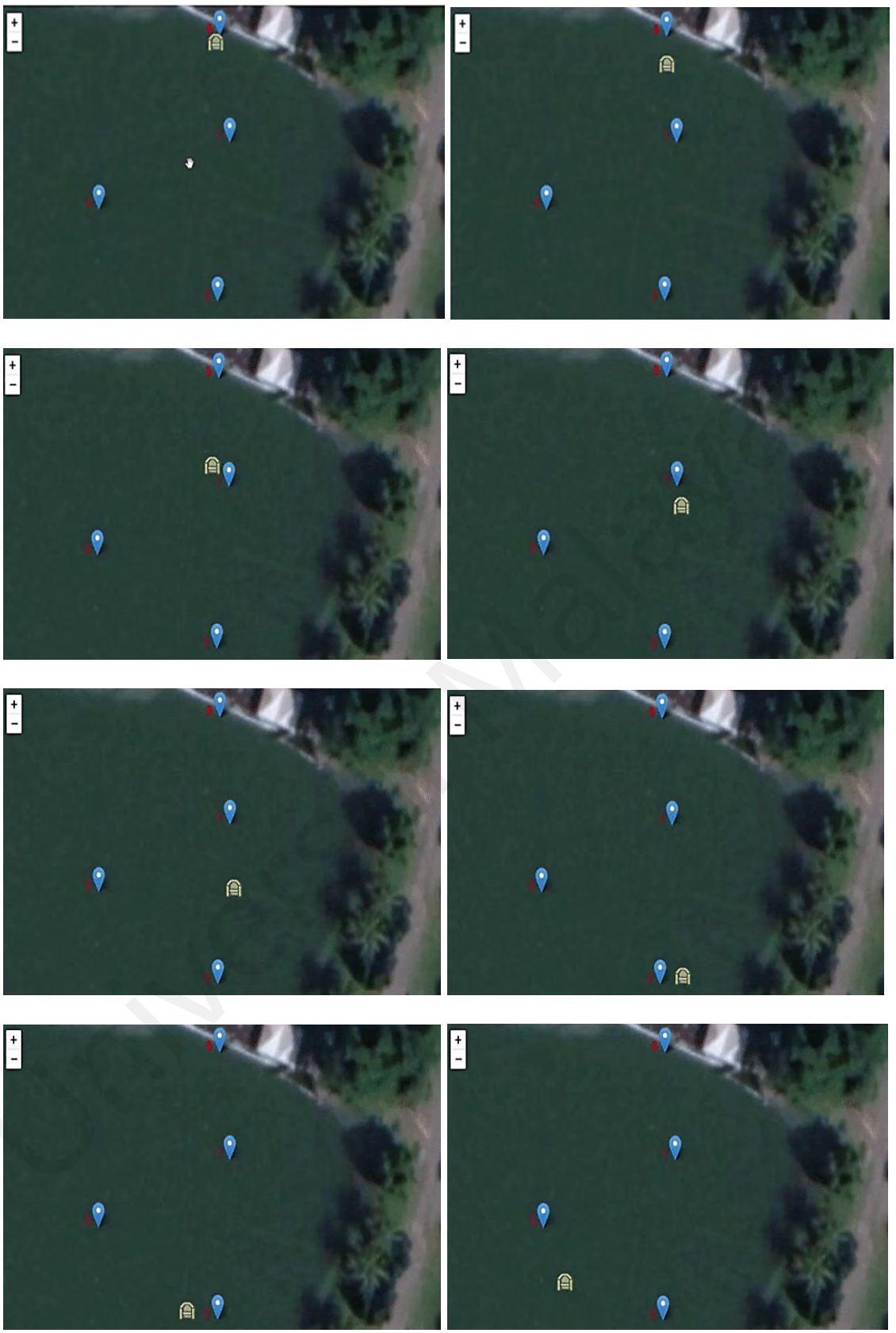

Figure 4.27: Robot markers following a round-trip from the start point shows RTK data is real-time and achieves minimal error

Figure 4.28: Robot turning at each station to direct heading towards the next target

The time to generate stations depends on the GA calculation. The GA begins by creating the population, and after that, selection, mating, crossover/breeding, and mutation. The process repeats following the number of generations decided. For the experiment, the population size is fixed at 100, meaning the GA recommends 100 possible routes per cycle. A fitness function decides the route with the shortest distance for a round trip from the starting position. Here, the system takes about 5 to 6 minutes on average to complete the task. The task involves a user loading and sending the KML map (see Figure 4.25) off-board so the split and genetic algorithms can be processed on a separate workstation. The result is sent back to the robot so the live map can show the sampling stations and best route (see Figure 4.26).

Case Study 1 achieves good error-handling performance satisfying the project requirements. The localization from the RTK system is sufficient, although a slight signal distortion can cause an error position to the robot. The 8-second interval seems optimal in getting <50 centimetres errors for the updating as it allows the robot to move before recalculating the next heading. The weather plays an important role; the RTK signal accuracy drops significantly during cloudy days or when it rains. The RTK records about ten centimetres of average on good days and up to three meters of an error on bad days. The RTK does not send or receive any usable signal when the rain is pouring.

4.9 Case Study 2 – Deep learning for water surface robot

Lake is different from the ocean. Ocean has a current that makes garbage float together. Due to surrounding factors such as strong wind, garbage disperses and floats everywhere on the lake surface, making garbage collection tedious. Furthermore, existing water surface robots for lake cleaning are usually remotely controlled. They are also expensive when equipped with a mechanism to detect floating garbage. Rarely does a low-cost version achieves automation for visual perception. For this reason, the adoption of intelligent robots to perform such tasks has been in demand. The advancement of AI algorithms such as computer vision and deep learning can solve a water surface robot visual perception. The second case study investigates this hypothesis.

The Yolo-v3 is a popular real-time object detection algorithm that identifies objects in videos, live feeds or images. The algorithm requires a user to define garbage types or classes, such as plastic bottles, styrofoam and plastic bag. The Yolo-v3 can reach an accuracy of 91% on live feeds with a minimum of 45 fps. The floating garbage dataset at lakes is common publicly, but the context may not be specific for this case study. Thus, a new dataset of floating plastic bottles, paper cups, and styrofoam containers is gathered

for the case study. Images are derived from google download using keywords such as ‘food waste on water’, ‘waste on water’, ‘plastic on water’, ‘paper waste on water’, ‘floating plastic bottles’, ‘floating paper’, ‘floating containers’. Data augmentation increases the dataset volume on the different orientations of the garbage. Data annotation is required next, and one can use *LabelImg*, a python library that draws bounding boxes of a region of interest in the image before annotating them. The darknet-53 architecture trains the model following the transfer learning and pre-trained weights of Yolo-v3.

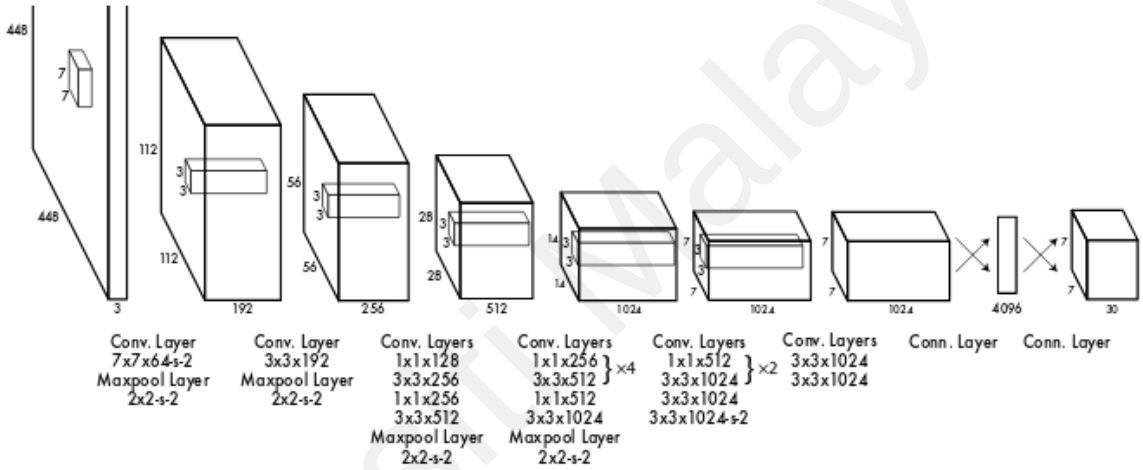

Figure 4.29: Darknet-53 architecture

Figure 4.30: Garbage detection example

Google Colab provides a free GPU for model training. The model training requires a specific library configuration, CMake building and OpenCV on the darknet (see Figure 4.29). The training consumes around 6 hours, and the Google Colab disconnects whenever the system is left idle or over 12 hours. The free account loses the trained weights once disconnected. Therefore, storing the training weights elsewhere is vital to make the system work as off-board processing for the water surface robot. Figure 4.30 shows an example of garbage detection. The MS COCO dataset is used to test if the model built achieves appropriate training weights.

4.9.1 System specifications

The design and development for case study 2 are not within the scope of this dissertation. Another student with an AI background completes it. The student does not have robotic know-how. The full implementation is reported elsewhere (Shamsul, 2020). The specification for this subsystem is as follows:

<u>Objective</u>

1. To develop object detection algorithm using computer vision and deep learning
2. To develop a visual perception system to locate floating garbage
3. To test the performance of the proposed system on the water surface robot

<u>Functional Requirement</u>

1. Display real-time video stream from Robot's POV.
2. Display bounding box when the garbage is detected.
3. Display line and distance from the robot to the garbage.

<u>Non-Functional Requirement</u>

1. Annotate a minimum of 2000 images for the garbage class.
2. The model must be able to detect garbage with at least 75% of accuracy.

3. The model must be able to process at least five frames per second.

4.9.2 Results and analysis

Case study 2 shows that the APIs from this study can be used to perform tasks such as:

1. Connect to the robot remotely and fully.
2. Connect to the robot's camera remotely and fully.
3. Perform live streaming of the camera.
4. Use deep learning to model garbage detection (off-board extension)
5. Determining garbage position from visual processing (off-board extension)
6. Collect garbage.

The student proposed the codes for steps (4) and (5) while the robot APIs handles the remote robot connection, live streaming and remote driving. Figure 4.31 shows the system publishes error messages if the robot connection fails. An experiment conducted at the Tasek Varsiti shows how well the robot performs with off-board processing in detecting floating garbage. The robot gets good accuracy when the garbage is near and static. Figure 4.32 shows the robot detecting near and static garbage with over 95% accuracy.

Figure 4.33 and Figure 4.34 show the robot in action while detecting and tracking floating garbage at Tasek Varsiti. The human view is taken from the observation deck, while the robot view is the live feed from the robot’s camera. The robot connection shows real-time consistency, allowing the human operator to easily track the garbage. Figure 4.35 shows the net trapping garbage from the sanitation experiment.

```
Total BFLOPS 65.290
Loading weights from ./model/yolov3_custom_fyp_last.weights...
 seen 64
Done!
Loaded - names_list: ./model/obj_fyp.names, classes = 1
[tcp @ 000001d5eca03a40] Connection to tcp://100.96.1.19:8081 failed: Error number -138 occurred
```

Figure 4.31: Robot connection error handling

Figure 4.32: Accurate detection when floating garbage is near and static

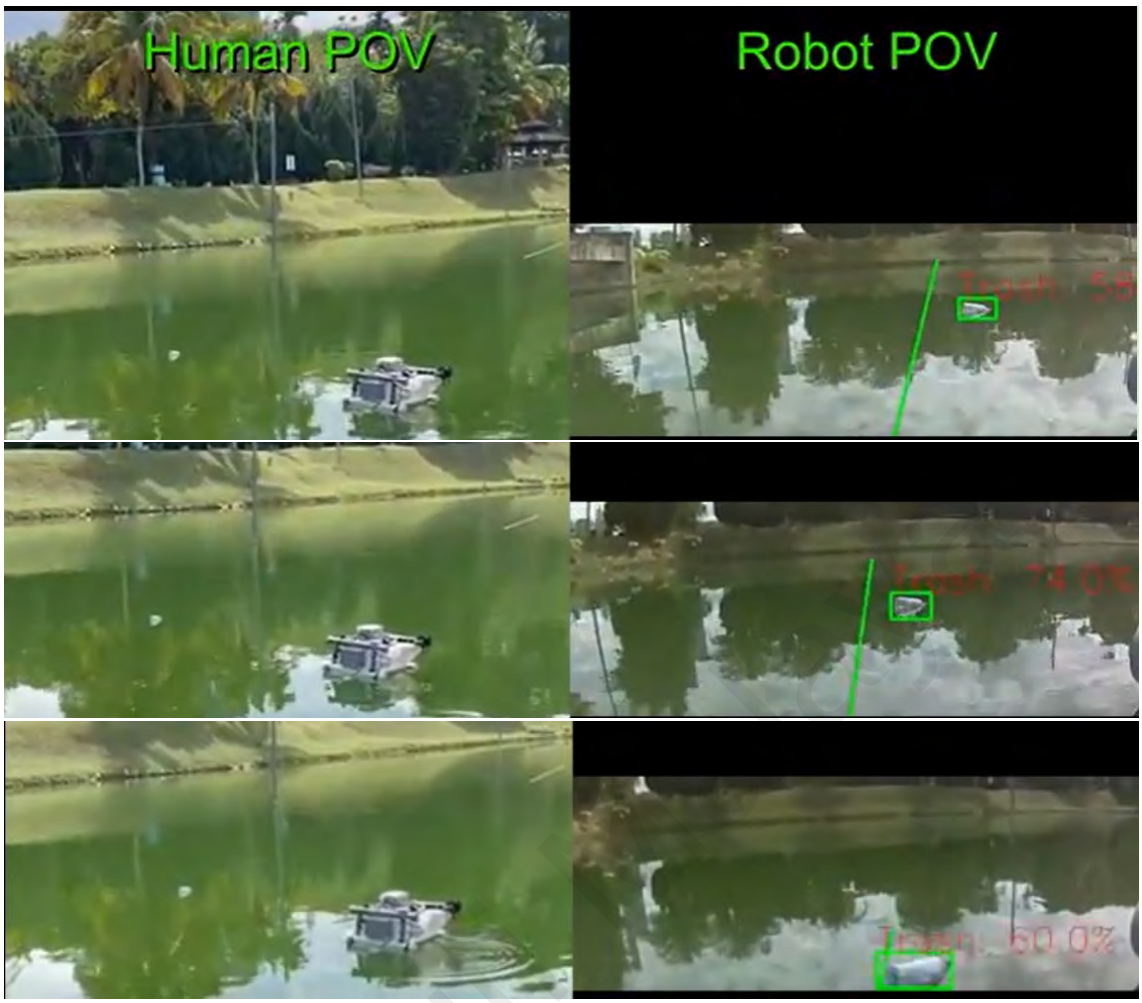

Figure 4.33: Robot detects floating plastic bottle and collects it

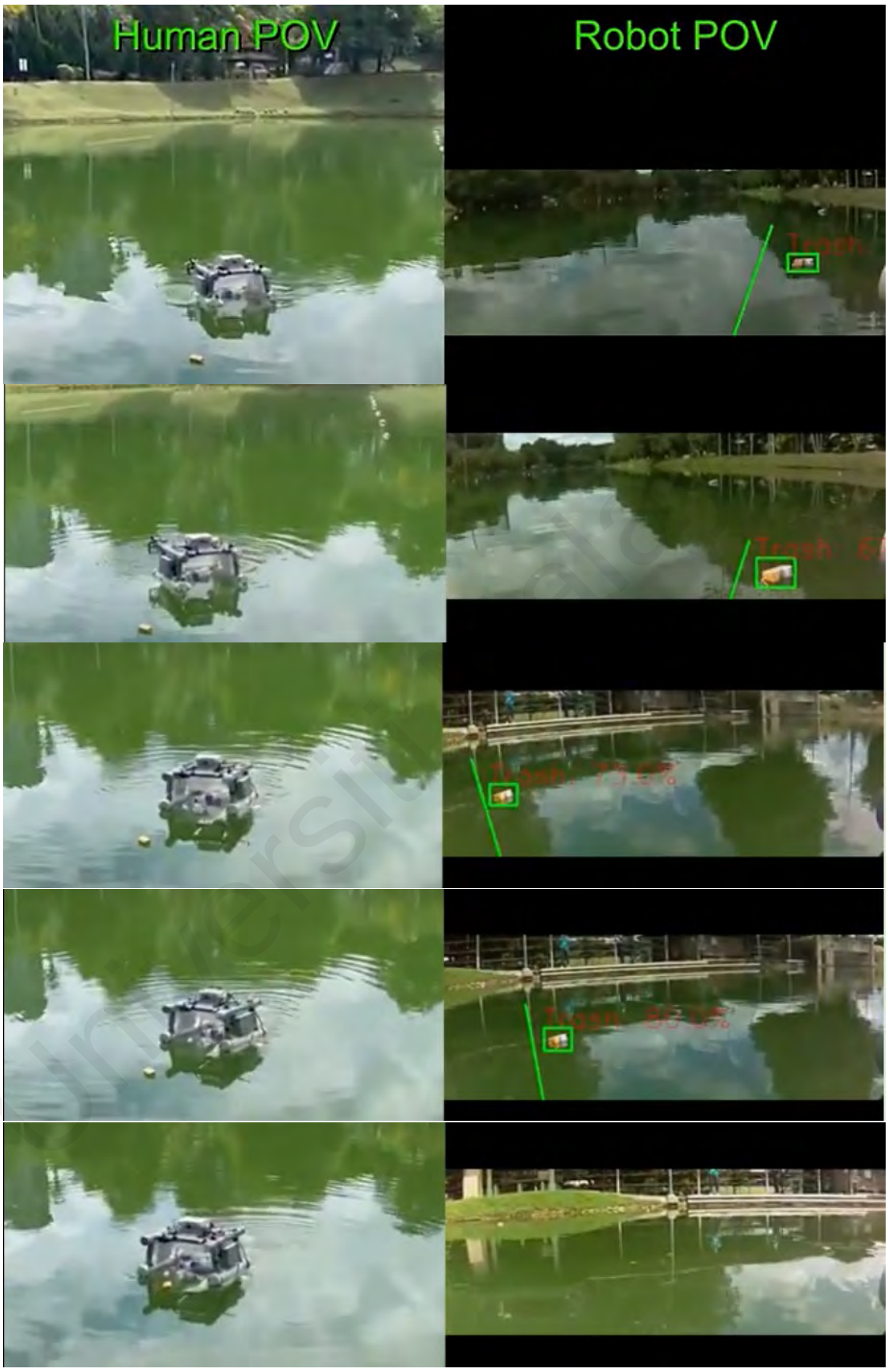

Figure 4.34: Robot detects drink container and tracks it

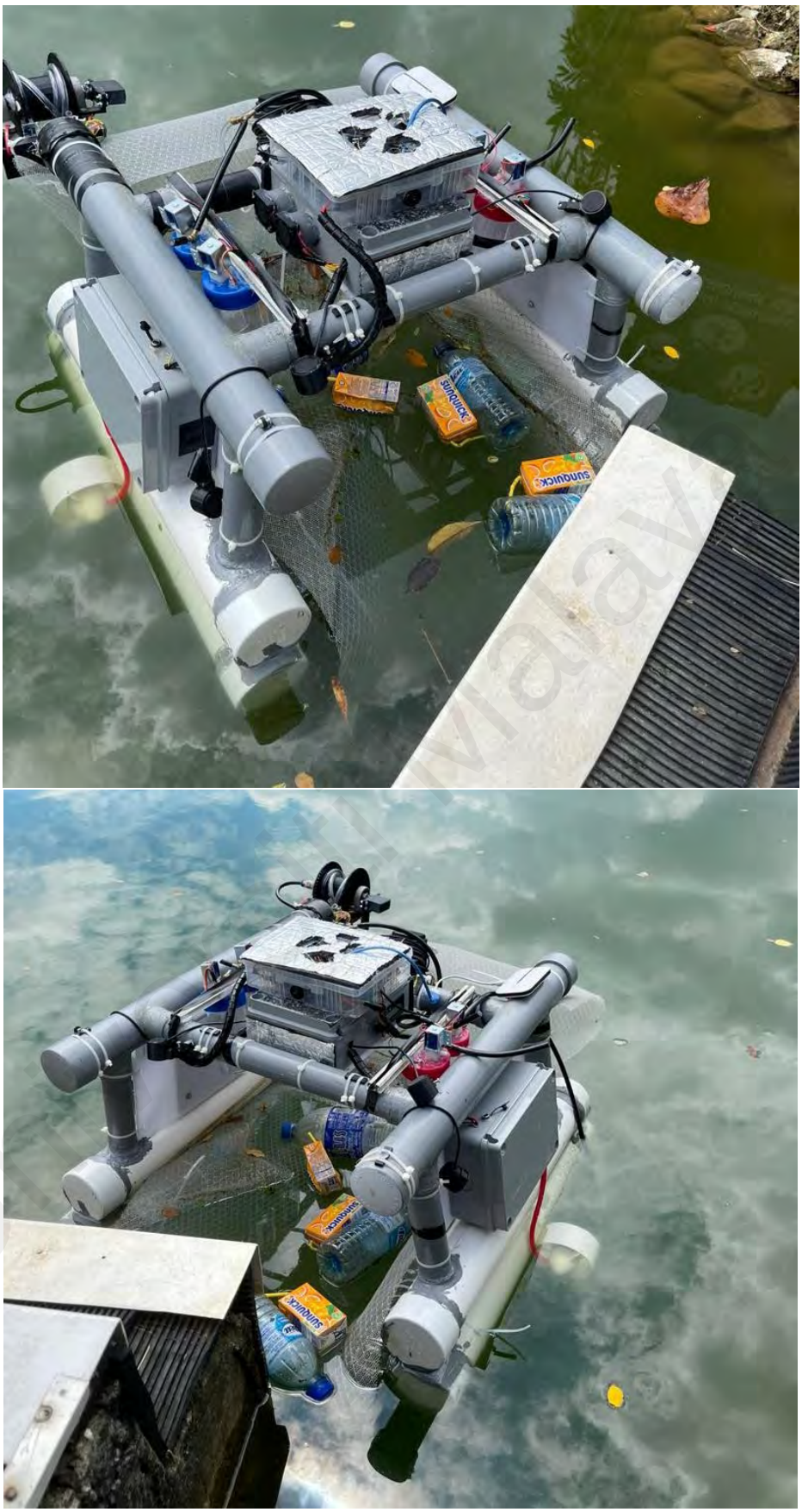

Figure 4.35: Nylon net trapping floating garbage from sanitation task

Case study 2 showcases the APIs for robot connection, live streaming and remote driving work well with off-board processing for object detection. The robot has to stay connected so that the deep learning running on the live streaming can give real-time feedback. The detection accuracy is high when the robot and garbage are near and static (see Figure 4.32). However, the detection accuracy drops significantly when the robot drives while the garbage moves with the current (see Figure 4.33 and Figure 4.34). The Raspberry Pi camera focal range also limits the accuracy detection. The robot can perform detection below 5 meters range on average.

The brightness level is a factor that influences detection. So does reflection. When the area is bright, the accuracy gets higher. The accuracy is lacking when the water surface reflects and diffuses lights. Adjusting the placement and tilt of the camera may improve the detection accuracy. A higher position with a low tilt may give the robot a better vantage point of view and likely be effective against the reflectance surface. The robot's wide opening with a nylon net helps trap floating garbage. The operator may drive through the garbage and *eats* it. So long the robot does not drive backwards, the garbage gets trapped and is easy for human collection.

4.10 Discussion

A water surface robot requires a structure that can float and maintain buoyancy at different speeds and load settings. The water surface robot should consider an open robotic system design to cut hardware costs with a limited budget. Fundamental to the open system is the software, networking and positioning subsystems. The robot software system must feature distributed processing, modularity and cross-platform support. In this case, a low-cost system relying on a single central processor like the Raspberry Pi must consider branching the I/O to handle sensors and devices. Suitable message-passing

topology is crucial for quality networking. Again the central processor plays a significant role in exchanging messages with devices and applications over the TCP connection. The Raspberry Pi message-passing topology is commendable and recommended for low-cost robot development.

The positioning system is an investment when autonomous navigation is considered for the robot. While GPS is available outdoors, reducing pose error is beneficial to a system that floats and moves, unlike mobile robots that can pause for satellite calibration. Case study 1 shows that the SLAM extension only works when parsed reliable landmarks. These landmarks are coordinates of sampling stations extracted by the RTK positioning system. If the pose error is high, robot localization and updating will get distorted. The robot will fail to get to the correct sampling stations. Weather affects the RTK performance. When it gets cloudy, the accuracy drops significantly.

Off-board processing is vital when considering automation on low-cost robots. Case study 1 and case study 2 leverages the open robotics design. Two undergraduate students with an AI background and no robotics know-how implemented the water surface robot extensions. One contributes a module that splits the sampling areas into sub-regions and suggests the best route for the sampling trip. The robot APIs feed the split algorithm, GA and SLAM to achieve real-time autonomous navigation.

The other contributes a visual perception module that detects floating garbage on the robot's live stream. A deep learning darknet-53 architecture trains the model before the experiment. The robot APIs maintains full connection and video streaming for real-time detection so long a minimum of 2Mbps internet connection is available.

The AI extended modules depend highly on the robot's software, networking and positioning systems. The extended module needs to maintain communication with the robot as all the inputs needed come from the robot. Other than that, the robot's sensors need to recalibrate every time before starting the navigation and the detection. Therefore, running both extensions parallel increases the system's complexity and burdens the central processor heavily. As a result, the robot's performance may drop if there is no consideration in managing the various messaging and data streaming.

4.11 Chapter Summary

This chapter presents and discusses the performance of the robot structure, software system, the networking and message-passing topology, and RTK positioning quality. The results show the water surface robot is ready for remote connection and driving. An operator can drive the robot to sampling locations and get water samples. The operator can also drive the robot around for garbage collection. The robot off-board processing of the open robotics design is tested through two case studies. Case study 1 shows an implementation to automate route generation for sampling tasks. Case study 2 shows an implementation to automate floating garbage detection and tracking. The robot system performs well through both tests. The following chapter closes the dissertation and suggests recommendations for future work.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

The dissertation hypothesizes that an open robotic system design and off-board AI processing can reduce the water surface robot developmental cost for lake sanitation and sampling. Exploring this hypothesis leads to several contributions from this study. The first is designing and developing a low-cost robotic system that can glide on the water surface. The mechanical aspects of the robot, such as the materials, electronics and power supply, have to be carefully considered under a limited budget. They must be cheap yet durable and non-susceptible to the strain and stress of a lake condition. A robot with a water collection mechanism and netting to trap garbage has been developed.

Cost-wise, the materials and equipment for the final robot prototype amounted to less than RM10,000, when only the final design is considered. The design and development went through several prototypes before deciding on the one described in this dissertation. Compared to the commercialized water surface vehicles, the cost to develop the robot system presented in this dissertation is at least 90% lower. The commercialized ones are superior as an off-the-shelf solution; however, they usually handle single task. For the cost consumed, the water surface robot in this dissertation served two functions: to sample water, and to perform garbage detection and tracking.

Managing the sensing technology and data processing framework leads to the study exploring modular, distributed and cross-platformed software design. The study proposed the Raspberry Pi 3b+ as the central processor to the low-cost robot. Automation often requires imaging, and the Raspberry Pi is superior to its peers in vision processing. Many other microcontrollers are in the market; however, the Raspberry Pi also stood out for its I/O devices capacity. A new PCF8574 API has been designed and published in this study.

The API supports anyone using the PCF8574 as an IO expansion board for the Raspberry Pi. Without the PCF8574, the robot is limited to the number of devices it can connect.

The robot's APIs also considers robot networking connection, video broadcasting, VPN server and network security. A message-passing topology design and development following the Raspberry Pi as the central controller is contributed. The messaging include exchanges between the Raspberry Pi with IMU, I/O extender, I2C and UART protocols over the TCP connection. A positioning system complements the outdoor robot. Although GPS services are free, the study explores rover-based receivers' utility for accuracy and stability. The RTK positioning system is set up for this purpose. Interestingly, the RTK works well on the robot achieving an average of <50 centimetres errors. The errors are good enough for robot localization, considering the challenging lake condition.

The onboard computational resources on a low-cost robot have never been encouraging. For this reason, low-cost robots rarely consider automation. In this dissertation, off-board processing is proposed leveraging the distributed, modular and cross-platform software support. The communication API allows live video streaming so an external deep learning module can run garbage detection in real-time. The RTK constantly updates the robot's position so an external module can recommend GA for path planning and SLAM for autonomous navigation.

The off-board computation is unique and allows other researchers to deploy advanced algorithms without restrictions on computing resources. Anyone without knowledge of robotic and electronic could run the robot and apply any algorithm for testing. Only the results need to be communicated back onto the robot for real-time actions. In summary, the design and development of the water surface robot is a success. The experiment results

have been encouraging. The Android app is easy to use; a non-expert can drive the robot remotely to collect water samples or collect floating garbage. For advanced users, connecting the robot to a workstation allows more data for analytics.

As Malaysia experience rapid urbanization and population growth, water pollution levels have also increased, which affect the water quality of lakes. The problem requires improvement in water quality monitoring and mitigation measures. Hopefully, the proposed open design robotic system with AI technologies can improve lake cleaning and sampling methods, which are currently conducted manually.

5.2 Future work

There are many directions in which this dissertation can take form. The first is upgrading to faster protocols for streaming video. The current streaming is sufficient, with a minimum of 2Mbps internet speed. However, a higher bandwidth such as 5Mbps can allow HD quality streaming. The deep learning algorithm can show better detection of small objects with higher resolution images or live streams. While the case study 2 algorithm shows high accuracy with single object detection, the algorithm has not considered group detection. Detecting a group of floating plastic bottles or containers remains a gap. Another gap is detecting dry leaves and some animals like ducks or geese, which often appears at local lakes.

There are also other object detection algorithms to try; Fast R-CNN, Faster R-CNN, Histogram of Oriented Gradients (HOG), Region-based Convolutional Neural Network (R-CNN), Region-based Fully Convolutional Network (R-FCN), Single Shot Detector (SSD) and the Spatial Pyramid Pooling (SPP-net). The YOLO-v3 proof to be lightweight, which suits the context of this study. Other detection algorithms, however, may offer

higher accuracy and faster detection. Classifying garbage according to types is has an advantage too.

The current water surface robot design faces 50% accuracy of garbage detection when turning. There are two reasons why the issue arises. One, the camera missed the garbage as angle rotation changes robot point of view drastically. Two, the internet connection could drop causing a delayed inferencing. Moving forward, these issues can be tackled by exploring multi-camera and edge computing on the robot. More cameras can cover wider angle thus when the robot is turning, one of the cameras is bound to capture the garbage within its frame. With edge computing, the robot does not have to rely on cloud-based inferencing. The trained model can infer garbage on board. Nevertheless, the tradeoff here would be cost as edge devices are more expensive than Raspberry Pi.

A possible direction is to combine case study 1 and case study 2 into a single project. The Raspberry Pi most likely gets overburdened to run both extensions simultaneously. Without managing the data flow and communication protocols, the Raspberry Pi can get overwhelmed and quickly corrupts the message exchanges among devices onboard and off-board. A new module or use case needs to be considered. Figure 5.1 shows a model to streamline message exchange between the Raspberry Pi, sensors, and other I/O devices.

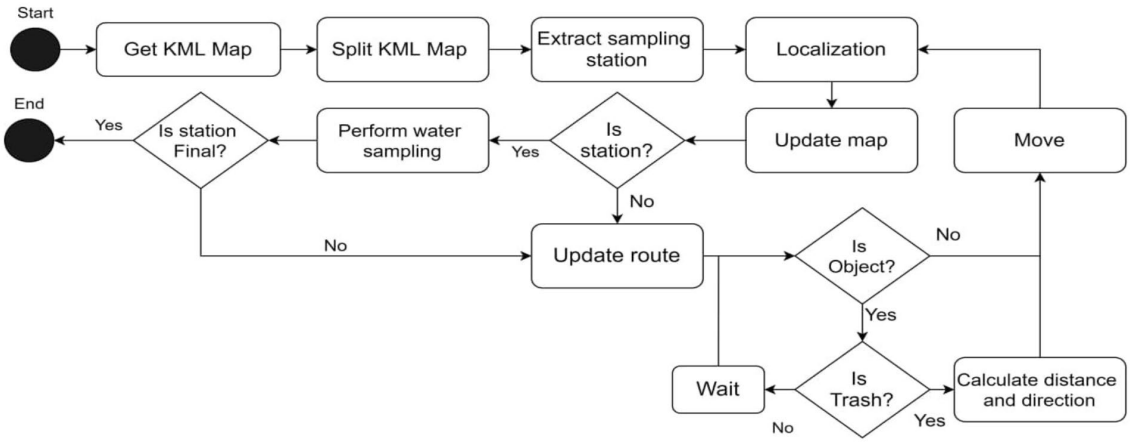

Figure 5.1: A water surface robot system for water sanitation and sampling

The model proposes that the robot localizes and updates its position on a map. Before navigating to the next sampling station, the robot runs the deep learning model to detect any nearby garbage. The robot can make a slight detour and collect nearby garbage en route to the next sampling station. The interchange between localizing and map updating and garbage detection continues until the robot reaches a sampling station. The 8-second interval where the robot drives after receiving the go command can be used to refresh sensors and clear the cache/buffers of the Raspberry Pi. This way, the central processor receives the most recent position update and detection outcome for the next cycle.

Lastly, there are gaps regarding the robot mechanical structure to explore when the existing architecture is considered. The water community has an interest in column sampling, where samples are taken at different depths. This dissertation proposed a mechanism to sample at surface level and 1-meter depth. In future, upgrading the hose reel design can address column sampling. Adding plastic hinges or stoppers at the robot's front opening can support garbage trapping. Keeping the garbage trapped allows the robot to perform a backwards movement, which can be necessary if the model in Figure 5.1 is followed.

REFERENCES

Abrams, M. (2018, May 16). *Remote Robot Cleans Trash from Water*. The American Society of Mechanical Engineers. https://www.asme.org/topics-resources/content/remote-robot-cleans-trash-water

Adeniyi, O. M., Azimov, U., & Burluka, A. (2018). Algae biofuel: current status and future applications. *Renewable and sustainable energy reviews*, *90*, 316-335.

Agrawal, P. & Bhattacharya, B. (2013). Aquatic multi-robot system for lake cleaning. In *Nature-Inspired Mobile Robotics* (pp. 171-178). World Scientific.

Al-Badaii, F., Shuhaimi-Othman, M., & Gasim, M. B. (2013). Water quality assessment of the Semenyih river, Selangor, Malaysia. *Journal of chemistry*, *2013*.

Allen, E., Browne, J., Hynes, S., & Murphy, J. D. (2013). The potential of algae blooms to produce renewable gaseous fuel. *Waste management*, *33*(11), 2425-2433.

Avio, C. G., Gorbi, S., & Regoli, F. (2017). Plastics and microplastics in the oceans: from emerging pollutants to emerged threat. *Marine environmental research*, *128*, 2-11.

Azizul, Z. H. (2019). Internal discussion with SPAN and NAHRIM on challenges of lake sampling in Malaysia. *Unpublished manuscript.*

Ballance, R., & Bartram, J. (2002). *Water quality monitoring: a practical guide to the design and implementation of freshwater quality studies and monitoring programmes*. CRC Press.

Barry, R. G. (2019). The world hydrological cycle. In *Introduction to Physical Hydrology* (pp. 8-26). Routledge.

Beloev, I. H. (2016). A review on current and emerging application possibilities for unmanned aerial vehicles. *Acta technologica agriculturae*, *19*(3), 70-76.

Blue Growth (2018, Nov 5). *An autonomous catamaran to remove floating plastic debris in ports and harbours*. German Research Center for Artificial Intelligence GmbH - Robotics Innovation Center. https://robotik.dfki-bremen.de/en/research/projects/ wasteshark.html

Cable, R. N., Beletsky, D., Beletsky, R., Wigginton, K., Locke, B. W., & Duhaime, M. B. (2017). Distribution and modeled transport of plastic pollution in the Great Lakes, the world's largest freshwater resource. *Frontiers in Environmental Science*, *5*, 45.

Clearpathrobotics (2016, Dec 19). *HERON UNMANNED SURFACE VESSEL*. https://clearpathrobotics.com/heron-unmanned-surface-vessel/

Dahlan, M. Q., Kadir, H. A., Isa, K., Ambar, R., Arshad, M. R., & Noh, M. M. (2019). Development of surface cleaning robot for shallow water. In *Proceedings of the 10th National Technical Seminar on Underwater System Technology 2018* (pp. 45-54). Springer, Singapore.

Dunbabin, M., Grinham, A., & Udy, J. (2009, December). An autonomous surface vehicle for water quality monitoring. In *Australasian conference on robotics and automation (ACRA)* (pp. 2-4). Citeseer.

Galileo GNSS. (2018, Apr 22). The path to high GNSS accuracy. https://galileognss.eu/the-path-to-high-gnss-accuracy/

Gasim, M. B., Toriman, M. E., Muftah, S., Barggig, A., Aziz, N. A. A., Azaman, F., ... & Muhamad, H. (2015). Water quality degradation of Cempaka Lake, Bangi, Selangor, Malaysia as an impact of excessive E. coli and nutrient concentrations. *Malaysian Journal of Analytical Sciences*, *19*(6), 1391-1404.

GPS. (2020a, Mar 11). GPS Accuracy. National Coordination Office for Space-Based Positioning, Navigation, and Timing. https://www.gps.gov/systems/gps/ performance/accuracy/

GPS. (2020b, Mar 11). Other Global Navigation Satellite Systems (GNSS). The National Coordination Office for Space-Based Positioning, Navigation, and Timing. https://www.gps.gov/systems/gps/performance/accuracy/

GREENBAY. (2018, Oct 20). *USV-SM800 Unmanned Cleaning boat*. Wuhan Greenbay Marine Technology Co. http://www.whgbay.com/en/Product_center/Furniture_lighting /lamp_series/186.html#

IADYS. (2019, Feb 23). *An innovative robotic solution for collecting marine waste*. Interactive Autonomous Dynamic System. https://www.iadys.com/en/jellyfishbot-2/

Isa, S. H., Ramlee, M. N. A., Lola, M. S., Ikhwanuddin, M., Azra, M. N., Abdullah, M. T., ... & Ibrahim, Y. (2020). A system dynamics model for analysing the eco-aquaculture system of integrated aquaculture park in Malaysia with policy recommendations. *Environment, Development and Sustainability*, 1-23.

Jayawant, A., & Sakpal, A. (2018). Aqua Skimmer for Trash Collection. *International Journal of Applied Engineering Research*, *13*(5), 5-8.

Jung, Sungwook, Hoon Cho, Donghoon Kim, Kyukwang Kim, Jong-In Han, and Hyun Myung. 2017. 'Development of algal bloom removal system using unmanned aerial vehicle and surface vehicle', IEEE Access, 5: 22166-76.

Kong, S., Fang, X., Chen, X., Wu, Z., & Yu, J. (2019). A NSGA-II-based calibration algorithm for underwater binocular vision measurement system. *IEEE Transactions on Instrumentation and Measurement*, *69*(3), 794-803.

Kong, S., Tian, M., Qiu, C., Wu, Z., & Yu, J. (2020). IWSCR: An intelligent water surface cleaner robot for collecting floating garbage. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Linchant, J., Lisein, J., Semeki, J., Lejeune, P., & Vermeulen, C. (2015). Are unmanned aircraft systems (UAS s) the future of wildlife monitoring? A review of accomplishments and challenges. *Mammal Review*, *45*(4), 239-252.

Madani, B. M. (2019). *Autonomous Vehicles Delivery Systems: Analyzing Vehicle Routing Problems with a Moving Depot* (Doctoral dissertation).

Manjanna, S., Li, A. Q., Smith, R. N., Rekleitis, I., & Dudek, G. (2018, May). Heterogeneous multi-robot system for exploration and strategic water sampling. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 4873-4880). IEEE.

NAHRIM. 2009. Study on the Status of Eutrophication of Lakes in Malaysia. NAHRIM, Seri Kembangan.

Nishar, A., Richards, S., Breen, D., Robertson, J., & Breen, B. (2016). Thermal infrared imaging of geothermal environments and by an unmanned aerial vehicle (UAV): A case study of the Wairakei–Tauhara geothermal field, Taupo, New Zealand. *Renewable Energy*, 86, 1256-1264.

Prempraneerach, P., & Kulvanit, P. (2010). Autonomous Robot Boat for Water Sampling and Environmental Data Acquisition Tasks. In *Proceedings of 2010 International Conference on Test and Measurement (ICTM2010).*

Rafique, S. M. S. M., & Langde, A. (2017). Design and fabrication of river cleaning machine. *IJSART, 3*(11), 8-18.

Rahmawati, E., Sucahyo, I., Asnawi, A., Faris, M., Taqwim, M. A., & Mahendra, D. (2019, December). A water surface cleaning robot. In *Journal of Physics: Conference Series* (Vol. 1417, No. 1, p. 012006). IOP Publishing.

Rajavel, S. E., & Shantosh, O. (2019). Bionic Person to Scoop Waste on Water Bodies. *International Journal of Emerging Technology and Innovative Engineering, 5*(7).

Riaz, Z., Pervez, A., Ahmer, M., & Iqbal, J. (2010, June). A fully autonomous indoor mobile robot using SLAM. In *2010 International Conference on Information and Emerging Technologies* (pp. 1-6). IEEE.

Ruangpayoongsak, N., Sumroengrit, J., & Leanglum, M. (2017, October). A floating waste scooper robot on water surface. In *2017 17th International Conference on Control, Automation and Systems (ICCAS)* (pp. 1543-1548). IEEE.

Saramanus, S., & Boonyaroj, V. (2019). The Application of Remote-Controlled Assisted Surface Water Sampling. In *Applied Mechanics and Materials* (Vol. 891, pp. 149-153). Trans Tech Publications Ltd.

Sharip, Z., & Suratman, S. (2017). Formulating Specific Water Quality Criteria for Lakes: A Malaysian Perspective. In *Water Quality*. InTech.

Sharip, Z., Zaki, A. T., Shapai, M. A., Suratman, S., & Shaaban, A. J. (2014). Lakes of Malaysia: Water quality, eutrophication and management. *Lakes & Reservoirs: Research & Management*, *19*(2), 130-141.

Sinha, A., Bhardwaj, P., Vaibhav, B., & Mohommad, N. (2014, February). Research and development of Ro-boat: an autonomous river cleaning robot. In *Intelligent Robots and Computer Vision XXXI: Algorithms and Techniques* (Vol. 9025, p. 90250Q). International Society for Optics and Photonics.

Sinha, E., Michalak, A. M., & Balaji, V. (2017). Eutrophication will increase during the 21st century as a result of precipitation changes. *Science*, *357*(6349), 405-408.

Soumya, H., & Preeti, B. G. (2018). Pond Cleaning Robot. *International Research Journal of engineering and technology*, *5*(10), 2395-0056.

Steccanella, L., Bloisi, D. D., Castellini, A., & Farinelli, A. (2020). Waterline and obstacle detection in images from low-cost autonomous boats for environmental monitoring. *Robotics and Autonomous Systems*, *124*, 103346.

Su, C., Dongxing, W., Tiansong, L., Weichong, R., & Yachao, Z. (2009, October). An autonomous ship for cleaning the garbage floating on a lake. In *2009 Second International Conference on Intelligent Computation Technology and Automation* (Vol. 3, pp. 471-474). IEEE.

Sumroengrit, J., & Ruangpayoongsak, N. (2017). Economic Floating Waste Detectionfor Surface Cleaning Robots. In *MATEC Web of Conferences* (Vol. 95, p. 08001). EDP Sciences.

Thakur, A. K., Chaudhary, D. N., Alam, M., & Sharma, K. (2019). Designing and Fabrication of Smart Water Skimmer. *International Journal of Scientific Research and Review.* 7(3), 3358-3361.

Turner, I. L., Harley, M. D., & Drummond, C. D. (2016). UAVs for coastal surveying. *Coastal Engineering*, *114*, 19-24

Valada, A., Velagapudi, P., Kannan, B., Tomaszewski, C., Kantor, G., & Scerri, P. (2014). Development of a low cost multi-robot autonomous marine surface platform. In *Field and service robotics* (pp. 643-658). Springer, Berlin, Heidelberg.

van Beusekom, J. E. (2018). Eutrophication. In *Handbook on Marine Environment Protection* (pp. 429-445). Springer, Cham.

Wang, Z., & Liu, Y. (2010). Pose control of a lake surface cleaning robot using backstepping and polar coordinates. *Advanced Robotics*, *24*(4), 537-557.