DES3

Swimming Pool Drown Detection and Rescue System

FYP Progress Report

By:

LEUNG Lok Yan Loreen (CPEG & MAE), CHAN Tsz Ho (MAE),

CHENG Hoi Wai (COMP), CHONG Shing Tung (CPEG), LEE Ka Hei (MAE)

DES3

Advised By:

Prof. MA, Robin Lok Wang (MAE) & Prof. TSOI, Desmond Yau-Chat (CSE)

Submitted in partial fulfilment of the requirements of

COMP 4981 or CPEG 4901 or ENGG4901

in the

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

2021-2022

Date of Submission: 14 February 2022

Declaration of Project Nature

This is a joint department project specially endorsed by the School of Engineering.

This crossed department project is submitted to fulfil both Mechanical and Aerospace Engineering Final Year Design Project (MECH 4900) and Computer Engineering (COMP 4981 or CPEG 4901). In addition, this project is also used to fulfil the Engineering School requirement on Integrated Final Year Project - First and Second Major (ENGG4901 and ENGG4902).

Thus, the grouping included:

Students from MECH: CHAN Tsz Ho, LEE Ka Hei, LEUNG Lok Yan Loreen

Students from CPEG: CHONG Shing Tung, LEUNG Lok Yan Loreen

Co-advised By:

Prof. TSOI, Desmond Yau-Chat (Computer Science and Engineering) & Prof. MA, Robin Lok Wang (Mechanical and Aerospace Engineering)

Executive Summary

Due to the noticeable lifeguard shortage in Hong Kong, the current lifeguards are overworked, which may cause fatigue and the lifeguard's attention to drop [1] [2] [3]. This magnified the chance of drowning without being noticed immediately.

In response to drowning, some current systems are mainly developed to alert and support the lifeguard to observe the drowning people underwater. Poseidon is one of the examples that developed to the commercial level. However, these applications focus on the alert upon identifying victims who are already drowning.

On the other hand, our Swimming Pool Drown Detection and Rescue System focuses on the entire supporting architecture. In particular, we focus on identifying pre-drown [4] victim with an active rescue system. In our system, each swimmer will be given a wearable device to measure and detect any abnormal physical conditions by monitoring blood oxygen level and heartrate. When an abnormal signal is detected, the web application will alert the lifeguard and shoot a lifebuoy to the victim's location under the lifeguard's attention.

Our system would further focus on the indoor 25-meter training swimming pool setting. This venue selection not only enables a better focus of the rescue system on fundamental functionality but is also chosen for being the most common type of pool in Hong Kong’s public swimming pool selected to maximize the possible usage.

Table of Contents

1. Introduction

1.1 Background

In Hong Kong, lifeguard shortage is a yearly problem. During summer, the peak season of swimming, the frequency of insufficient lifeguards leading to the closure of swimming pools and beaches increases significantly (Local lifeguard union, 2019). In 2019, some swimming pools and beaches were closed due to a shortage of nearly half of the required number of lifeguards needed during peak season [5].

Under the pandemic, the problem of lifeguard shortage had been worsened. The keen competition between the public and private sector with the opening of Water World Ocean Park exacerbated the lack of lifeguards. According to a report by the Standard in August 2021, there were no lifeguards at half of the beaches resulting in temporary close for some of them. Due to the closures, the training for newly recruited lifeguards and revalidation for qualified lifeguards were cancelled [6]. This reflects the seriousness and the impact of lifeguard shortage.

One of the lifeguard's main responsibilities is maintaining safety. However, without enough lifeguards at the swimming pool implies there will be not enough or even no people monitoring the safety of the pool users. Among the safety issues, drowning is the most severe. As shown in Figure 1, it is noteworthy that poolside drowns took up 13% of all drowning cases, with 8% in the private sector and the other 5% in public swimming pools [7].

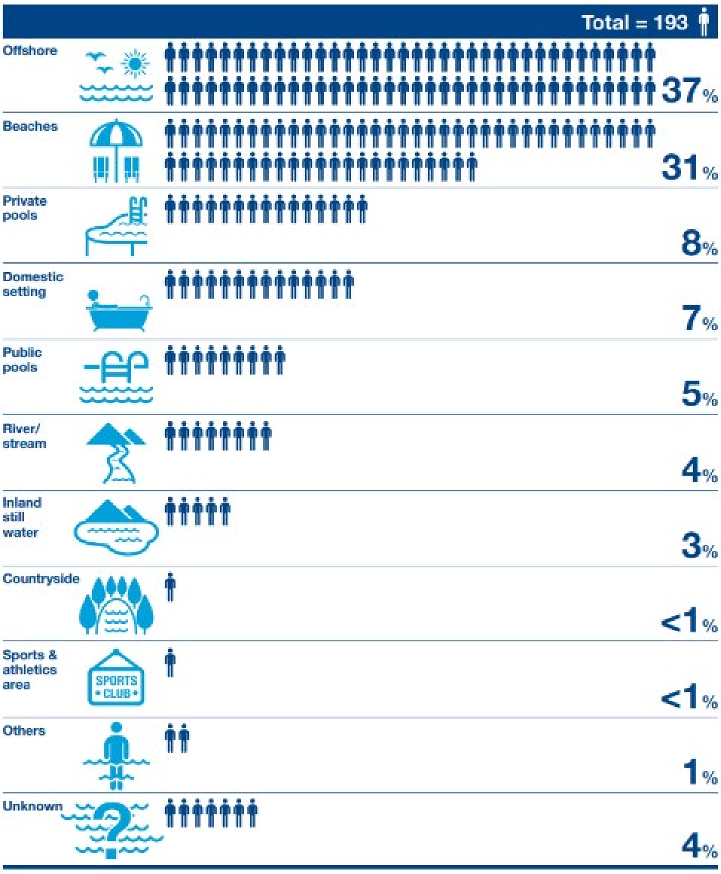


Figure 5. Drowning deaths by location, 2012-2016

Figure 1 Drowning Death by location in 2012-2016 [7]

Furthermore, even for swimming classes with a coach, there are occasionally cases of drowning for young swimmers. This increases the need for a better system. With the knowledge and concern mentioned earlier, improvement in drowning detection and rescue is imperative.

With the above considerations and case selection, to ease the lifeguard shortage pressure, a Swimming Pool Drown Detection and Rescue System is introduced to assist the lifeguard in efficiently identifying and saving the drowning victim.

Despite that there have been some developing drowning detecting systems, the current designs are mostly focused on drowning detection and warning. In our system, we would also include a follow-up action to shoot the lifebuoy to rescue the victim. With the drowning detection system, warning system, and lifebuoy throwing in one system, the signal transmission and reaction time can be further minimized. The rescue time can be shortened and become more automated. This aims to achieve a higher rescue rate since faster rescue means a higher chance of survival and fewer injuries experienced by the victim.

The drowning detection and warning sub-system is a combination of an electronic wearable device, a computer system for drowning alert trigger, a victim localization and a web application for the lifeguard to decide what to do when drowning is detected. Electronic components and computer programming will be utilized here.

The lifebuoy throwing sub-system consists of a robot that could move the mechanical parts such that it aims at the location of the drowning person and throws a lifebuoy for rescue. The throwing mechanism will be implemented by our Mechanical Engineering team[1] for the mechanical design and modification.

In the current development of our Final Year Project, we would focus our system on indoor swimming pool rescue for Olympic Pool Sized pools. In the future, the system may not be limited to indoor or outdoor swimming pools but also be used at beaches or the seaside.

[1] MECH FYDP team consists of Lok Yan Loreen LEUNG, Tsz Ho CHAN, Ka Hei LEE and Prof. Lok Wang Robin MA (Supervisor).

1.2 Objective of the Project

The aim of our project is to detect and rescue drowning victims in the pre-drown stage in a swimming pool. Although an IoT wearable device detection system will be developed to identify the victim, the lifeguard will also then be alerted through the Web application and is given a choice to decide whether to shoot or not. Once the choice to shoot has been made, the lifebuoy will be shot to the drowning victim's location to aid the victim's self-rescue capability and shorten the time needed for lifeguard rescue.

There are three parts in the system, namely (1) drown detection sub-system; (2) drown-alarming sub-system and (3) a lifebuoy throwing sub-system. Wearable device and Image processing are done in the "drown detection sub-system". Though a combination of the three systems, the rescue speed might be able to be further minimized.

With this aim in mind, we define the following project objectives:

Objective 1: To build a wearable device that identify potential drowning victims

The wearable device will be responsible for detecting the blood oxygen level and the heartbeat of the swimmer during swimming for every 30 seconds. The device will then send the signal to the web application control. Within the wearable device, some key abilities include:

1. Enhanced water resistant to 5 BAR[2] [8] of the wearable device. This is used to support the sensor to function well for swimming in a swimming pool.
2. Accurate collection of sensor's data under water and during exercises.
3. Analysis of the data collected by the sensor to determine whether the situation is abnormal or has fallen into the drowning category
4. Transmitting the drowning message information from the swimming pool to the web application control system beside the swimming guard. The distance would be approximately 50m above water and 1m depth in water.

We will also explore the possibility of adopting a camera-based approach to locate the drowning swimmers who sunk with their heads in the water for 15 seconds as a hybrid approach to be able to identify the stage 2[3] drowning victim. In this approach, the camera would be mounted underwater inside the swimming pool to capture the swimmers' actions and moments.

[2] 5 BAR water resistance means the device is capable of withstanding splashes, showering and swimming. It is not suitable for pool-side diving, sea driving or water sports like snorkelling, water-skiing.

[3] Stage 2 means the victim chokes in water, and his/her vocal cord involuntary spasm to protect one's lung. More details are given in Section 2.3.1.1 Software design.

Objective 2: To locate the drowning victim accurately with a sub-system

The location detection sub-system will be developed by in camera-based approach, where visible light detection by a camera system is used to identify the location of the possible drowning victim.

When the victim's wearable device sensor detected abnormal readings or have reported drowning, the Arduino LED device on the wearable device will be lit up for the location system to detect the light. A camera mounted on the top of the swimming pool will be used for checking the location of the emitted light from the wearable device. This camera selection and location further enhances the identification of the location. The computation of the camera-light location system would be done by the external computer system due to the large processing power.

Objective 3: To build a user-friendly application to alert and assist the lifeguard in making drown and rescue decision

The aim of building an application is to allow the lifeguard to make the final decision making for the lifebuoy throwing mechanism operation. Since the lifeguard should be a professional judgment on rescue, the final decision making would be fall back on the lifeguard.

In order to allow the lifeguard to continue the regular routine checking while being aided by the system, there are three requirements. These requirements are set based on making the application easy and require minimal attention for the lifeguard to use.

The criteria include:

1) User-friendliness for first time users. Since the lifeguard might not be professional in technology, it is suggested to have a simple and straight forward interface.
2) Responsiveness of the application. Since rescue is a timely manner, the system should have a minimal delay, so the lifebuoy throwing system can be initiated once the confirm signal is sent.
3) Clarity. The button and interface should be clear and easy to operate. Since rescue is a timely issue, it is not preferred to have the situation of wrong button pressing occur.

Node.js will be used to develop the web application to enhance compatibility. Different tablet computers and mobile devices may also use the application. This allows the flexibility of the management of the swimming pool. We will also simulate the swimming setting in the application to make the situation more manageable for the lifeguard to use.

Objective 4: To Integrate the Detection and Warning System with the Mechanical Lifebuoy Throwing System

For this objective, due to the immediate rescue aim, the signal transmission between the Detection and Warning system and the Mechanical Lifebuoy system should be fast. Thus, various signal transmission approaches will be explored, including but not limited to Wi-Fi transmission and cable transmission.

Moreover, the data transmitted from the detection and warning system, in particular the location and the basic sensor information, should be merely essential and sufficient data. This enables the focused resources on the operation of the Mechanical Lifebuoy system.

There may also be some mechanical lifebuoy throwing system data, including the current shooting stage, possible error message, and power status, transmitted back to the web application for error reporting and advanced monitoring of the Mechanical System for the lifeguard.

Objective 5: (MECH) To calculate and test the shooting distance and shooting angle

This part is responsible by the students from the MAE department.

A three-dimension shooting machine is used. A spherical polar coordinate system, as illustrated in Figure 2 is used to control the shooting distance. The magnitudes of two angles and the shooting velocity are calculated by using the data collected by the sub-system. By controlling these parameters, the life ring can be shot to different positions.

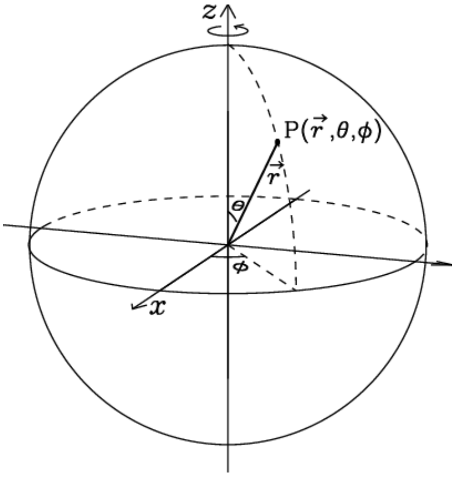


Figure 2 Figure illustrating the spherical polar coordinate system

Objective 6: (MECH) To shoot the life ring that to the potential drowning victim's location

This part is responsible by the students from the MAE department.

Air cannon and a wheel shooter are considered. For air cannon, air pressure is used to ignite the launching. For the wheel shooter, two wheels, which are driven by two motors, are used to accelerate the life ring to create a projectile motion. The shooting time is targeted to be smaller than 20 seconds.

2. Methodology

2.1 Project Progress

Our project can be divided into three sub-systems, including the wearable sub-system, image processing sub-system, and server sub-system. The tables below are our current progress on different portions of the project.

Table 1 shows the progress for the wearable sub-system. As shown from the table, the basic control and functionality are tested. The hardware construction for the wearable device is also mostly finished. The main issue lies in unstable sensor input for MAX30100. In the upcoming stage, we would focus on fully utilising the sensor's capability with Arduino Nano. At the same time, we would design a waterproof mould for enhancing the waterproof capability of the wearable device.

Status	Wearable Sub-system
Done	• Designed the wearable system with the respective functions, criteria to be considered • Decided & purchased the electronic hardware components from Taobao or other stores • Built the workable prototype with Arduino Uno, blood oxygen sensor (MAX30100) and LEDs • Researched and set the threshold for the drown detection • Programmed and tested the control of MAX30100 with Arduino to enhance the result stability • Tested the accuracy of the MAX30100 against another external medical used device • Implemented basic testing functions to test the 4 main hardware components, MAX30100, LED, Arduino Uno & Nano, speaker separately and repeatedly • Built the preliminary waterproof prototype for image testing, water testing, poolside environment testing
Doing	• Design the silicone rubber potting mould to be 3D printed for the wearable device • Improving the sensor detection for a longer duration of time to solve MAX30100 stability problem
To Do	• Improve the stability of the sensor of measuring the blood oxygen level • Build the waterproof model & use silicone to pot and form the waterproof layer • Do the waterproof test for the hardware under 0.5-1 metre depth

Table 1 Project Progress of Wearable Sub-system

Table 2 shows the progress of the image processing sub-system. We have finished corner detection and perspective transformation algorithms. We are still working on the LED detection algorithm. After finishing the LED algorithm, we will have a thorough test on the three algorithms.

Status	Image Processing Sub-system
Done	• Experimented corner detection with Harris corner detection • Experimented corner detection with hue extraction and Hough transformation • Have done perspective transformation algorithm • Experimented LED detection with template matching • Experimented LED detection with multiscale-template matching
Doing	• Fine tune parameters for LED detection • Collect testing images
To Do	• Combine results for multiscale-template matching • Test the 3 algorithms.

Table 2 Project Progress of Image Processing Sub-system

Table 3 shows the progress of the server. We have finished all technical critical development, communication channels to all sub-systems, and basic UI. Currently, we are implementing the detailed interaction of the backend server and image processing algorithm. And in the future, we will add an authentication function and improve the user interface.

Status	Server Sub-system
Done	• Wireless webcam input from a smartphone • Streaming webcam to the web client. • Designed the user interface • Finish interactive setting page • Finish monitor page UI • Finish touch to shoot function • UI Optimization for IOS (iPad mini) • Established a communication channel between the server and the robot.
Doing	• Adaptation to the image processing algorithm • Alert function
To Do	• Translation of image position to real position for float launching robot • Authentication function • UI improvement with Material UI

Table 3 Project Progress of Server Sub-system

Status	Other
Done	• Controlled the movement of the linear actuator with 2-way 24V relay through Arduino Uno in two opposite directions • Tested and controlled the planetary stepper motor movement by Arduino Mega2560 and Ramp1.6
Doing	• Have a more accurate control of the planetary stepper motor • Control the location of the top part of the linear actuator, with a given angle. The system would be able to calculate the required height
To Do	• Calibrate of the whole system

Table 4 Project Progress of Other Tasks

2.2 Design Philosophy

To maximise the ability of the system to rescue, there are a few design principles to be taken into account.

- Responsiveness
    - As an active rescue system, the response time has to be short to save time to rescue. Thus, the response and signal transmission between different systems have to be minimal.
- Modularity
    - Having the system divided into sub-systems, they can contribute individually to alert and thus support the drowned rescue decision making of the lifeguard. This also facilitates the debugging and easy maintenance of the system in the long run.
- Optimised for the Swimming Pool Environment
    - Being a computer system operated in a swimming pool, durability is very important. Since electronic components will be easily damaged upon water contact, the material selection should be carefully made.
- Minimal disturbance to the routine
    - Being a rescue system supplementary to the lifeguard observation, the system is not designed to replace lifeguard professionals. Thus, lifeguards should be not disturbed or are required to spend additional attention to observe the report from the system. The system should be mainly automated and have its own ability to alert the lifeguard instead. Additionally, the location and the installation of the system should not affect the regular operation of the swimming pool.
- Safety
    - Being used by the poolside and as a rescue system, the system aims to rescue instead of causing casualty. Thus, the insulation of electricity becomes crucial. Moreover, the system setup should not induce possible injury.

2.3 System Overview

The system consists of 4 sub-systems, the wearable device, the server, the image processing algorithm, and the launching robot. Figure 3 shows how the sub-system is integrated and how the data transfers in between. First, the wearable device will detect if the user is drowning. If so, it will emit an LED signal, and the server's camera will capture that signal. Then the server will send the webcam input to the front end and the image processing algorithm to determine the position of the LED. Once the location is known, it will send to the front end to alert the lifeguard. The lifeguard will give a shooting command or trigger automatically after the timer runs out. The command is then sent to the float launch robot to shoot the float to the destination. In the following, we describe the design and implementation of each sub-system in detail.

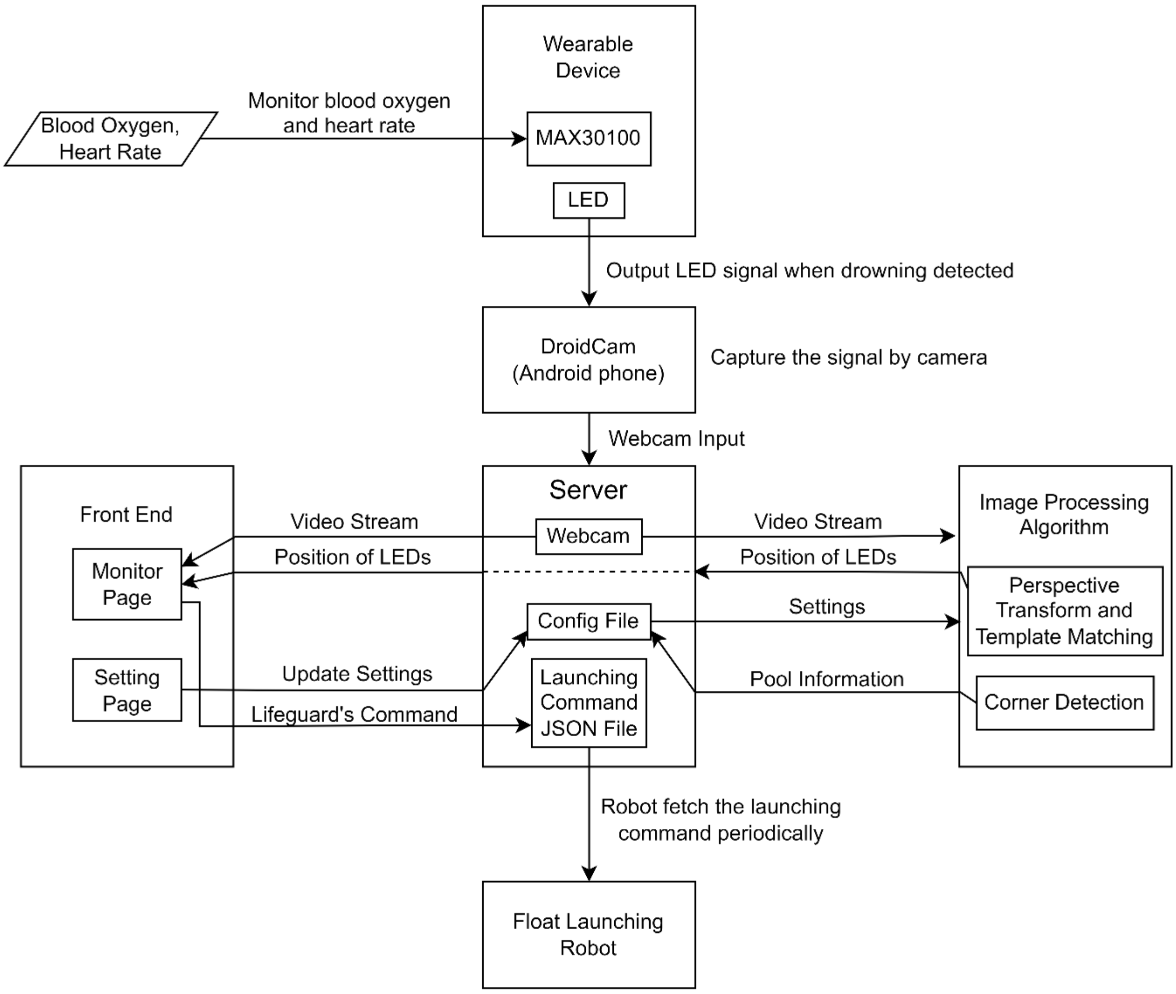


Figure 3 System architecture

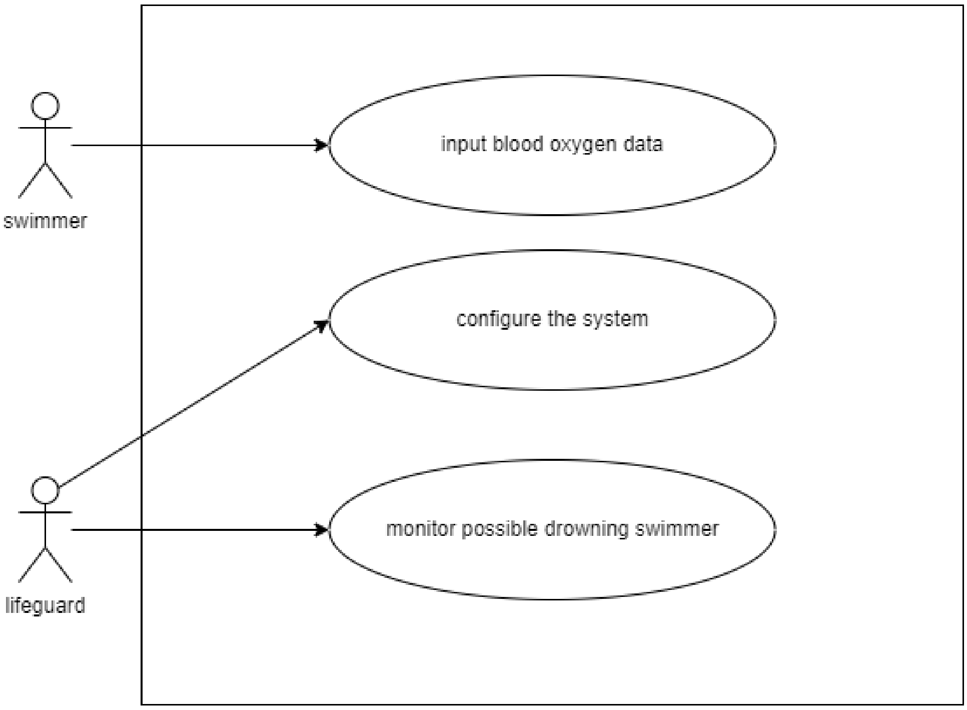


Figure 4 Use Case Diagram

2.3.1 Use Case: Input Blood Oxygen Data

This use case describes how a swimmer inputs blood oxygen data into the wearable device when he/she is in the swimming pool.

Basic Flow

1. The use case begins when the swimmer wears the device before going to the pool. To wear the device, the swimmer should fasten the strap and fit it onto the wrist.
2. The swimmer turns on the device and data will be collected automatically.
3. The swimmer turns off the device and takes it off after he/she leaves the swimming pool.
4. The use case ends.

2.3.2 Use Case: Configure the System

This use case describes how a lifeguard configures the pool size, pool corners and bot position.

Basic Flow

1. The use case begins when the lifeguard chooses to configure the system.
2. The system displays the configuration page.

{Input pool size, bot position, and count down time}

3. The lifeguard inputs the pool size, bot position and count down time and clicks the confirm button.

4. The system displays a message to notify the lifeguard that the inputs are received.

5. If the lifeguard chooses to perform automatic corner detection

    5.1. The lifeguard clicks the detect button and the four corners are detected automatically.

6. The system displays a screen for manual adjustment for the corners.

7. The lifeguard adjusts the corners manually.

8. The lifeguard confirms the configuration.

9. The system displays a message to notify the lifeguard that the inputs are received.

10. The use case ends.

Alternative Flow

A1: Invalid pool size/bot position/countdown time

At *{Input pool size, bot position, and countdown time}* if any of the inputs is not a positive integer,

1. The system displays a message to notify the lifeguard that there is an invalid input.
2. The flow of events is resumed at *{Input pool size, bot position and countdown time}*

2.3.3 Use Case: Monitor Possible Drowning Swimmer

This use case describes how a lifeguard monitors the swimming pool and shoot lifebuoy to the drowning swimmer.

Basic Flow

1. The use case begins when the lifeguard chooses to monitor the pool.

2. The system displays the swimming pool.

{Monitor the swimming pool}

3. While the lifeguard is monitoring the pool,

    {Detect drowning swimmer}

3.1. If the system detects a drowning swimmer,

3.1.1. The system marks the drowning swimmer with a yellow box.

3.2. The lifeguard clicks on a position of the pool.

3.3. The system displays a prompt window with “Confirm rescue? Yes/No”.

3.4. If the lifeguard clicks “Yes”,

3.4.1. The system shoots a lifebuoy to the target

3.4.2. The prompt window closes.

3.5. If the lifeguard clicks “No”,

3.5.1. The prompt window closes without any action taken.

Alternative Flow

A1: No response from the lifeguard

At *{Detect drowning swimmer}* if no response is received from the lifeguard after some time,

1. If the prompt window has opened,
   1.1. The prompt window turns red.
2. Else,
   2.1. The prompt window pops out in red.
3. The system displays the countdown timer in the prompt window
4. If still no response received from the lifeguard after countdown,
   4.1. The system shoots a lifebuoy to the target automatically.
5. The prompt window closes.
6. The flow of events is resumed at *{Monitor the swimming pool}.*

2.3 Wearable Sub-system

2.3.1 Design

The wearable system consists of data analysis, wearable hardware design and waterproof design. In detail, the software required is C language for Arduino; the hardware counterparts include Arduino Nano, GY-MAX30100, WS2812 5050 RGB Round Panel and 6V power supply.

As shown in Figure 5, this sub-system targets to alert the imaging processing sub-system with a victim detected with a high chance of being drowned. With the user's blood oxygen level measured, the system would decide if the swimmer was drowning then respond with a warning light signal.

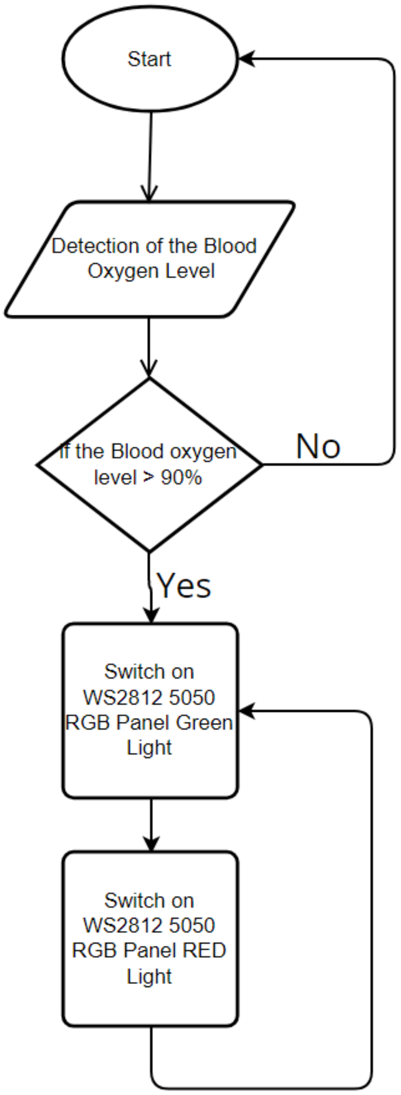

Start
Detection of the Blood Oxygen Level
f the Blood oxygen level > 90%
No
Yes
Switch on WS2812 5050 RGB Panel Green Light
Switch on WS2812 5050 RGB Panel RED Light

Figure 5 Flowchart representation of the drown-alarming sub-system

2.3.1.1 Software design

Physiological Effect of Drowning

Many may think drowning victims will scream and have rapid movement when drowning, but that is incorrect. According to the World Health Organization (WHO), the definition of drowning is "the process of experiencing respiratory impairment from submersion/immersion in liquid.", "Any submersion or immersion incident without evidence of respiratory impairment should be considered a water rescue and not a drowning." [4].

In stage 1 pre-drowning, the victim will panic and react with violent struggling while still being able to scream for help. Also, hyperventilating will occur, leading to a sudden rise in a heartbeat [9]. This stage will last for 20 to 60 seconds for adults, 10 to 60 seconds for a child [10].

In stage 2, when the victim aspirates water, it will trigger laryngospasm, which means the vocal cord will spasm to block the airway and stop water coming in to protect their lung. However, it also blocks the airway of breathing, which leads to brain hypoxia. In this stage, the victim can hardly be differentiated from a beginner swimmer as the victim is fatigued from the violent struggling and cannot yell for help due to laryngospasm [10] [11].

In stage 3, laryngospasm stops due to brain hypoxia, aspiration of water continues. Hypoxemia quickly leads to loss of consciousness and apnea. The sequence of heart rhythm deterioration is usually tachycardia, then bradycardia, pulseless electrical activity, and finally cardiac arrest [4].

PPG (Photoplethysmogram)

PPG is an optical technique used to detect volume change and oxygen change of blood in the peripheral circulation. When infrared (IR) travel through human tissue, it gets absorbed by bones, skin pigments, and blood. To measure heart rate, PPG detects the reflected IR intensity change. Since blood absorbs more IR light, when blood volume gains, the IR intensity reflected will decrease, vices versa [12]. For blood oxygen, we detect the colour of light reflected. Blood with sufficient oxygen will absorb more infrared and reflect more red light. Less oxygen will absorb more red light, making reflected light appear bluer [13].

A field that makes use of PPG is air safety. According to Air Safety Institute Safety Alert, when a pilot measured blood oxygen saturation is below 90%, his mental function begins to deteriorate, he should receive enough oxygen to function normally. Therefore, we may consider 90% of blood oxygen saturation is in a danger zone [14].

Having 90% as the threshold is further supported by the medical study conducted by University and Medical Centre in Netherlands [15]. Figure 6 shows the measured value of blood oxygen after swimming is still above 90% in real life scenario. Only few cases lie between 80-90% or even lower. With the fact that there was no drowning case reported throughout the study, this verified the validity of using 90% as our threshold value.

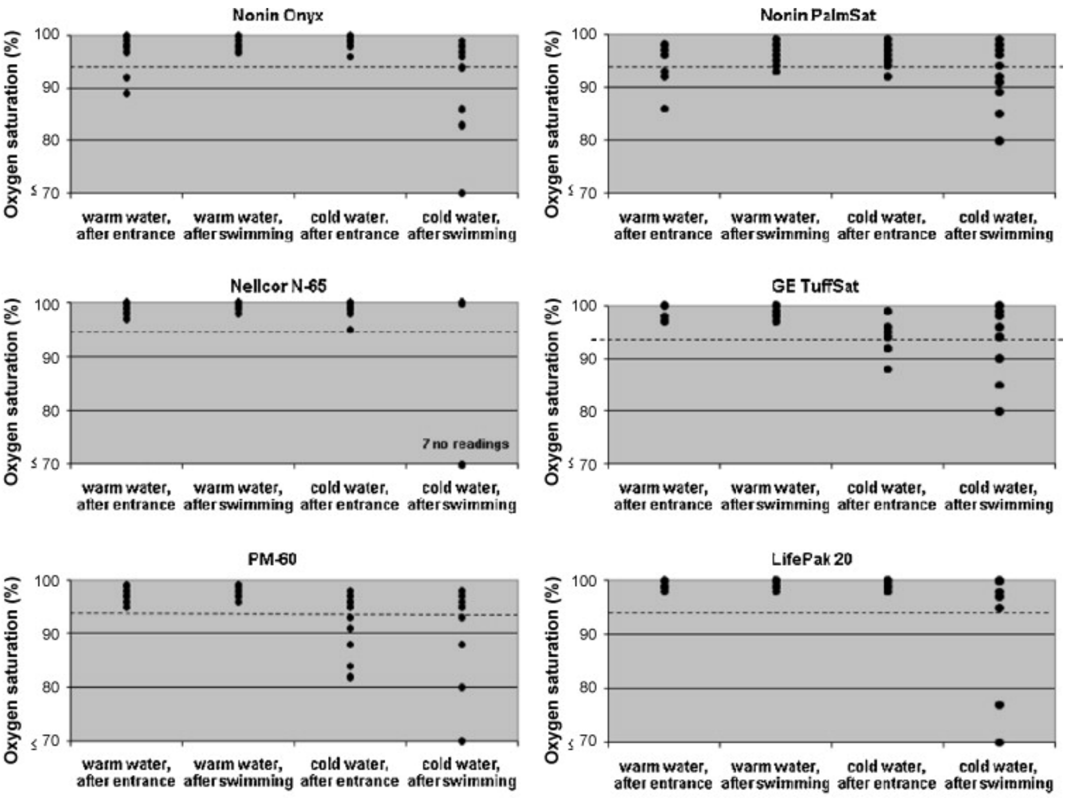


Figure 6 Line plots for oxygen saturation values (y-axis) per test situation (x-axis) from different pulse oximeters

Although PPG is a cheap and easy way to measure heart rate and oxygen levels, it has its limitation. PPG is inaccurate in tracking the PPG signals during daily routine activities and light physical exercises because of Motion Artifact (MA) caused by hand movement [16].

As a counterargument, commercial wearable health devices such as Garmin Venu and Huawei Band 6 do the all-day monitor. Take ZOLL's R Series Pulse Oximeter (SpO-2) as an example. During the no-motion condition, its accuracy is within ±2%, and during motion conditions, it is within ±3%. The error range is acceptable, especially in extreme cases like hypoxia [17].

As suggested from the "Feasibility of pulse oximetry after water immersion", the value taken from the finger is appreciated and more accurate [18]. This meant the possibility of using it on the wrist with further support from the wide usage from healthcare watches.

2.3.1.2 Hardware component

Mainly four components will be used. To minimise the dimension for ease of wearing, smaller devices are used. These include one Arduino Nano, one GY-MAX30100, four pieces of 3 Bit WS2812 5050 RGB Round Panel and a 6V power supply by two coined batteries.

Before selecting the hardware, two factors were considered: small dimension and ease of usage. As a result, Arduino Nano is chosen due to the minimal dimension and sufficient connectivity for the input and output devices. This include both the MAX30100 sensor and the WS2812 5050 RGE LED pieces. In our original design, we have also considered adding a waterproof speaker. However, due to the user experience and vulnerability added to the design, the design is later discarded.

Arduino Nano

Arduino Nano is chosen with the consideration of minimal disturbance to the routine for the swimmer, compatibility, and support to other electronic components. Comparing STM32 and Arduino Nano, Arduino Nano is smaller with less but sufficient functions. The smaller size helps building the waterproof layer since this size is easier to control. Although both are written in C language, Arduino's platform consists of more libraries and support. The support further facilitates us to collect data from the sensor ahead for more accurate calibration.

Comparing Micro-bit, a smaller microcontroller, with Arduino Nano, Micro-bit is a more basic micro-controller. The customization and programming are comparatively limited due to the fact it was designed for academic purposes for the general public.

GY-MAX30100

We have made some comparisons with another model of MAX30100. As shown in Figure 7, the green board is with resistors building deflect. The cheap cost suggested the testing application purpose for the water test, for the destructive testing.

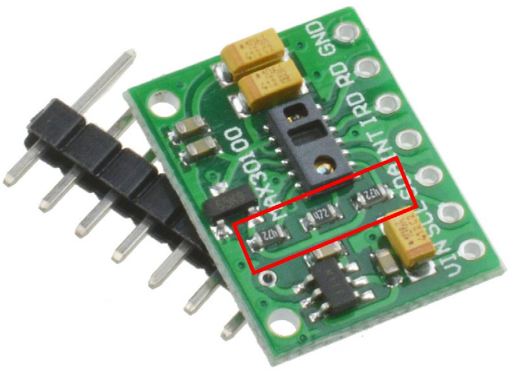


Figure 7 Green MAX30100 Circuit with design Deflect

WS2812 5050 RGB Round Panel

The RGB Round Panels (as known as LED) are used as the output, signal emission medium for the camera to capture. However, considering the warning LED being blocked by the waving of the drowning swimmer's hands, there are four LEDs installed in different directions, namely, top, bottom, left and right, placed around the wrist.

Also, we have tested with different colours, including but not limited to red, green and blue. There are some specific concerns for the colour. With the colour filter used by the image processing, various type of blue colour or colours that appears to be similar to the pool has to be excluded. With further consideration that the possibility of the image processing unit mistaken the swimmer's suit as an LED, dual colours are introduced. The LED will be blinking from red to green to enhance accuracy.

Waterproof Speaker with 4 Ohm and 3 W

Specifically, the user experience is limited by the dimension of the hardware diameter of 50mm. Compared with the normally worn locker key bracelet straps suggested in Figure 8, the longest key accepted would be 50mm, with a pointed design [19]. However, the waterproof speaker we would suggest is 57mm in diameter, with the additional circuit board, PAM8403, needing to adapt to the Arduino. It will increase the wearable device to be larger size and increase in weight. Thus, the idea of including such are later removed.

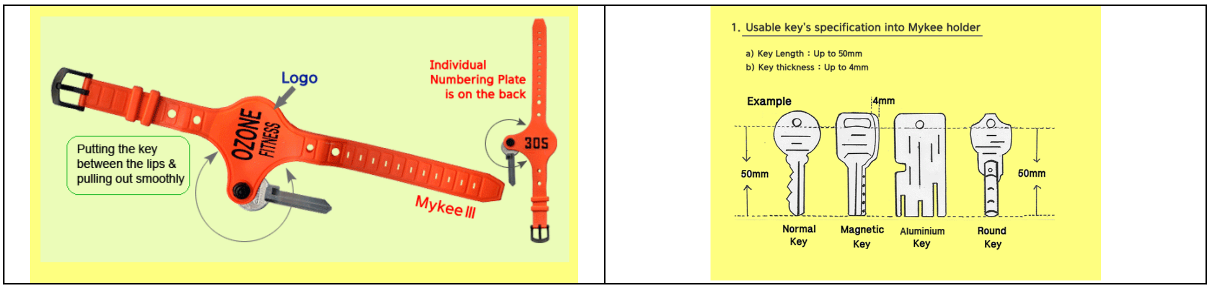

Logo
Individual Numbering Plate is on the back
Putting the key between the lips & pulling out smoothly
OZONE FITNESS
305
Mykee III
1. Usable key's specification into Mykee holder
a) Key Length : Up to 50mm
b) Key thickness : Up to 4mm
Example
4mm
50mm
50mm
Normal Key
Magnetic Key
Aluminium Key
Round Key

Figure 8 Advertisement figure shown the swimming wrist band which holds the locker key

3V coin battery

We have swapped the battery from 3.7V lithium-ion battery to 3V coin batteries (CR1220 / CR2032) for testing purpose due to easily discard feature and cost control. It is also worth notice that two batteries are needed to work in series in order to provide sufficient battery for the LED to lit up while powering the Arduino (6-9V). Since rechargeable lithium-ion battery are good for long use but it is not cost effective for vulnerable water testing stage. For our final product, 3.7V lithium-ion battery or LIR2450, 3.6 V, 120 mAh rechargeable coin battery will be used for repeated use in long term since both are capable for recharging.

Component	Arduino Nano	GY-MAX30100	WS2812 5050 RGB Round Panel	Waterproof Speaker with 4 Ohm, 3W	3V coin battery
Function	Processor	Blood Oxygen Sensor (also known as Pulse Oximeter)	Warning & Signal for Location detection	Warning	Power supply
Amount (pieces)	1	1	4	1	2
Dimension					
Width(mm)	18	14	17	57	20
Length(mm)	45	14	17	57	20
Height (mm)	5	3	3	13+2	3.2
Pin Number	32	5	3	2	2
Input Voltage					
Max. Voltage	12	5	7.5	4.8989	6
Min. Voltage	7	3.3	-0.5	3.4641	1.5

Table 5 Summary for the hardware for the wearable device on functionality and specifications

Table 5 concludes the specification for all the used hardware. This specifies the exact version and functionality of the hardware on the wearable device.

2.3.1.3 Waterproof design

A 3D model will be designed to contain the electronic elements with potting by silicone rubber, as shown in Figure 9. We will design two versions of the drawing and model to achieve an optimal waterproof of 5 bar. The first version aims to provide fundamental skills experience for potting with silicone rubber. The second design would focus more on the user experience and accuracy.

Figure 9 Electronic component potting with silicone rubber

2.3.2 Implementation

2.3.2.1 Software implementation

The software mainly includes the input signal from the MAX30100 (pulse oximeter) and the output signal to the LED. Among the two parts, libraries were used for enhancing their functionalities, respectively.

For the pulse oximeter, the library named MAX30100lib is used. This library is produced by OXullo Intersecans, which aims to provide a modular approach to calculate pulse rate and SpO2 for the Maxim-IC MAX30100. The main functions used in our programme are *pulseOxymeter->update()* of pulse oximeter and *result.Sao2*. The code segment Figure 10 below shows the update function and the averaging component. With about 10-20 results recorded per second, the average input value is taken to minimise the effect from some extreme value due to detection error between the skin surface and the sensor.

```
Pulseoxymeter_t result = pulseOxymeter->update();

 myTime = millis();

 if ( result.pulseDetected == true )
 {
  Serial.print(“Instant BPM: “);
  Serial.print( “” );
  Serial.println( result.heartBPM );
  Serial.print( “ Instant SaO2: “ );
  Serial.println( result.SaO2 );
```

```
// Averaging system
arrHbeat[arrHbeatIndex] = result.heartBPM;
arrHbeatIndex++;
if (arrHbeatIndex == arrHbeatNumOfData) {
  arrHbeatIndex = 0;
}

Serial.print(“Average value:”);
averHbeat = takeAverage(arrHbeatNumOfData, arrHbeat);
Serial.println( averHbeat );

arrSaO2[arrSao2Index] = result.SaO2;
arrSao2Index++;
if (arrSao2Index == arrSaO2NumOfData) {
  arrSao2Index = 0;
}

Serial.print(“Average value:”);
averSao2 = takeAverage(arrSaO2NumOfData, arrSaO2);
Serial.println( averSao2 );
```

Figure 10 Code Segment for the MAX30100

However, the detection for the blood oxygen sensor is not stable with the application on Arduino Nano. The behaviour with the Arduino Uno has been more promising as more successful values are taken upon close contact of the skin and the sensor as far as the wired connections are in contact. In contrast, with the same MAX30100, only with a changed Arduino Nano board, the data collected is limited within 2 minutes, then the measure will stop working. We are testing to discover whether this issue is caused by the limited RAM of the Arduino Nano Board or it might be some connection issue. With the Arduino Uno board as a reference, we think we should solve this issue within this month.

For the output signal, the LED programme has used the WS2812 library. This library mainly provides some functions for selecting a specific colour for RGB LED, while providing the signal control for customised speed. The main function used are as listed in Figure 11.

```
// set up the cRGB variable to store the fixed value, i.e. the warning colour

cRGB value1;

value1.b = 0; value1.g = 0; value1.r = 255;


// set up pin 9 as the output pin for LED1
LED1.setOutput(9);

// update the LED1 value according to the colour stated by value1
LED1.set_crgb_at(0, value1);
```

```
// update the LED1 status as previous set set_crgb_at(0, value1)
LED1.sync();

// to change the frequency of the signal by delaying the LED signal (change the time)
delay(1000);
```

Figure 11 Block included the commonly used code for WS2812 library.

The colours of the LED are also specially designed to avoid the missed up with the pool colour. In our design, we have chosen Red and Green to minimise the usage of BLUE. Also, the change of LED colour and the high frequency of colour change is added for improved recognition of the image processing component. The LED changing colours contrast with the swimmer suit; even the swimmer might have worn a red or green suit. Since the swimmer's swimming suit colour cannot change, this further reduces the chance of mistaking the suit as one of the possible values.

A customised high frequency of 10Hz (10 signals per second) to the LED colour change aids the image processing unit. The swimmers would not move as fast as the frame, say 3m per second. If we can discover two different colours changing at about the nearby spot within 1 second. In that case, this suggests a higher possibility that the image reading is the output from the wearable device instead of the swimsuit.

2.3.2.2 Hardware component

Figure 13, Figure 14 shows the combined circuit for the first version. As illustrated below, the 3 LEDs are placed around the breadboard. This design ensures the LED will be seen from a different angle. For instance, when the swimmer's hand may be waving, the back of the hand (i.e. top of the device) will face toward the water as in Figure 12. As a result, the top part of the device might not face the camera mount on the pool top and thus cannot be detected. With the tilted angle, the side LED will be shown instead. Then the camera will still be able to catch the signal from the side of the wearable device.

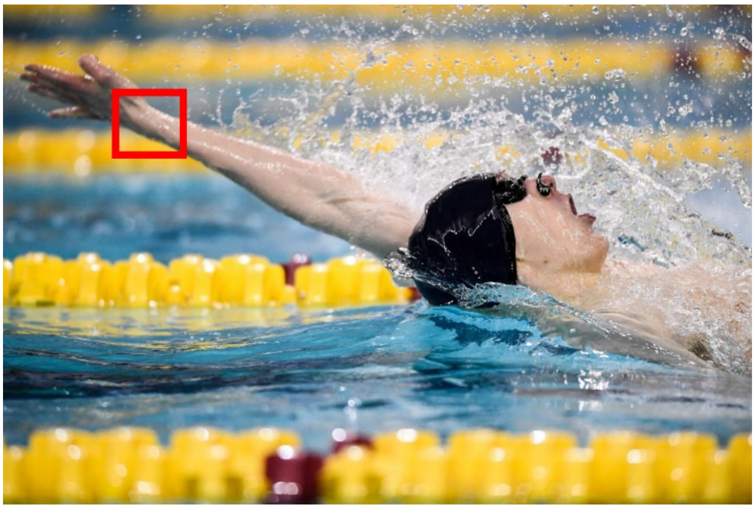


Figure 12 Swimmer having his/her back of the hand facing the pool [20]

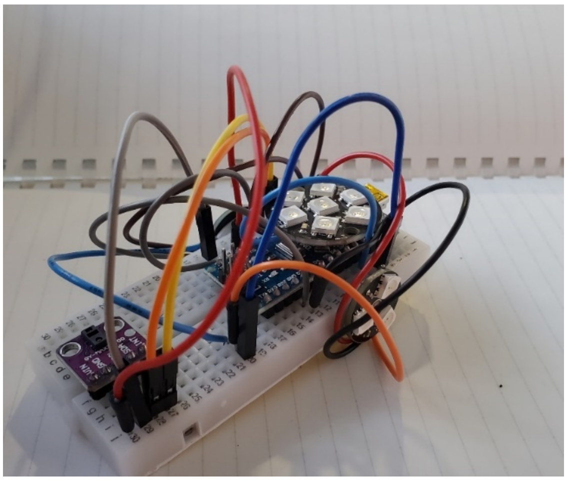


Figure 13 Prototype with the LED and the pulse oximeter

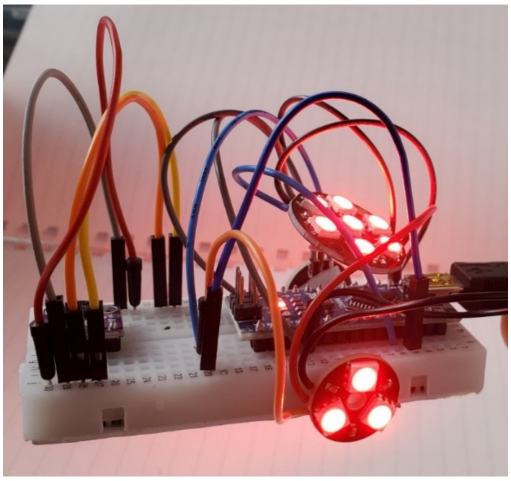


Figure 14 Prototype with the LED and the pulse oximeter with light lit

In the second version, the MAX30100 sensor is suggested to be placed at the bottom of the bracelet. This allows the sensor to have closer contact with the wrist to achieve a more accurate value. This is suggested with the working principle of the MAX30100 as illustrated in Figure 15. The infrared is emitted from the diode and reflected from the fingertip back to the photodetectors. With thin skin between the blood vessel at the wrist, setting the detection location at the wrist can reduce signal noise and infrared loss by travelling through to other body tissue.

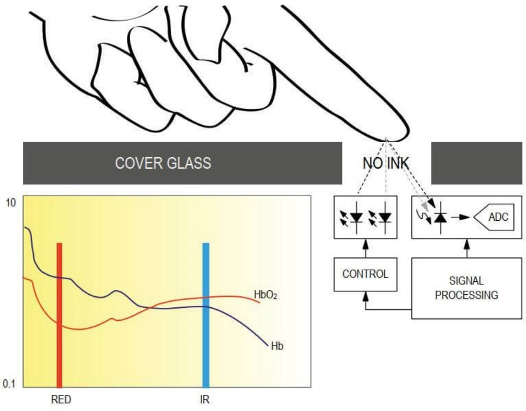


Figure 15 MAX30100 sensor system [21]

Circuit Diagram

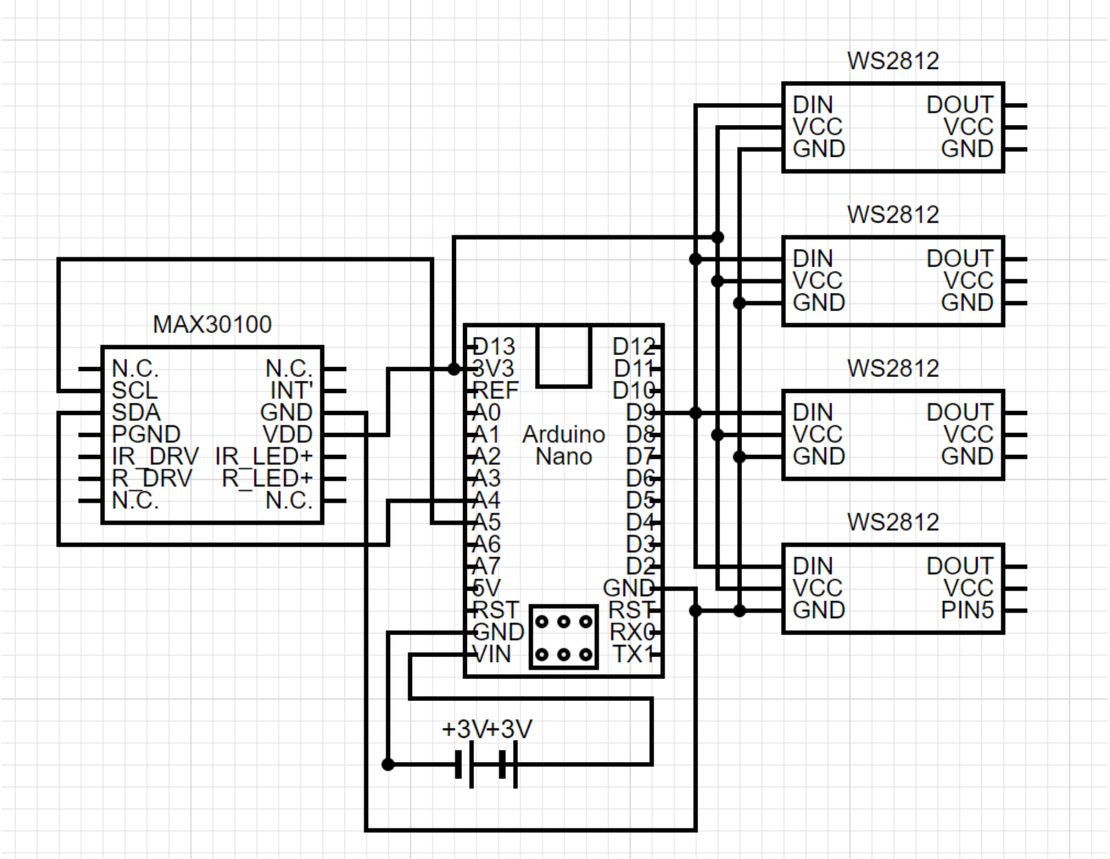


Figure 16 Circuit diagram for the hardware components

Figure 16 shows the detail circuit diagram with the 4 hardware components. In particularly, pin D9 of Arduino Nano is used as signal controlled for all four WS2812s. Pin A4 of the Arduino Nano is the SDA *and A5* is the SCL for the corresponding pins in MAX30100.

There are also some problems with the programming for the Arduino Nano.

Although Arduino Nano is ideal in size, the debugging progress is commonly known with the communication problem when reloading the programme to Arduino Nano board. The problem, as known as "avrdude", happens almost every time. Thus, it is highly recommended to test the functionality of the programme on Arduino Uno and the hardware before transferring the circuit design to Arduino Nano. This supports modularity and speeds up the debugging progress.

The main problem we encountered was the *avrdude* issue. After multiple testing and research, there are some possible solutions for the *avrdude: stk500_getsync()*. Firstly, the processor should be using "ATmega328P (Old Bootloader)". Secondly, different programmers have also been tested to bypass such an error. "Arduino ISP" is tested to be functioning for this Arduino Nano Chip.

Furthermore, to avoid any error when loading the programme, it is essential to remove all wires connected to the Arduino Nano Board, and then load the programme to the board.

2.3.2.3 Waterproof design

To achieve a waterproof of 5 bar, we would apply a dual-layer of protection for the electronic circuit to ensure its' functionality even under water.

We have tried a different approach. The first approach is placing the model inside a small lock-n-lock box, as illustrated in Figure 17. However, after water testing, the container we used was not waterproof enough, so another box was chosen to try. The second rectangular box had better performance in waterproof. Additional details are discussed in the Section 2.3.3.2: Water testing.

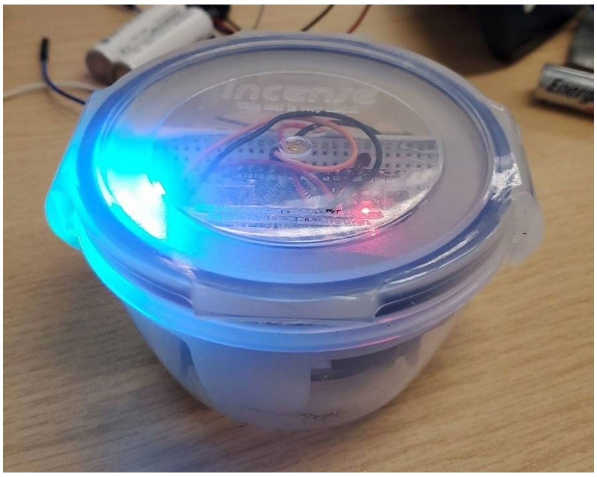


Figure 17 Wearable device placed in the first lock-n-lock box

There are two versions of mould designs for the circuit to better control the potting technology and effect of potting on our circuit.

Version 1

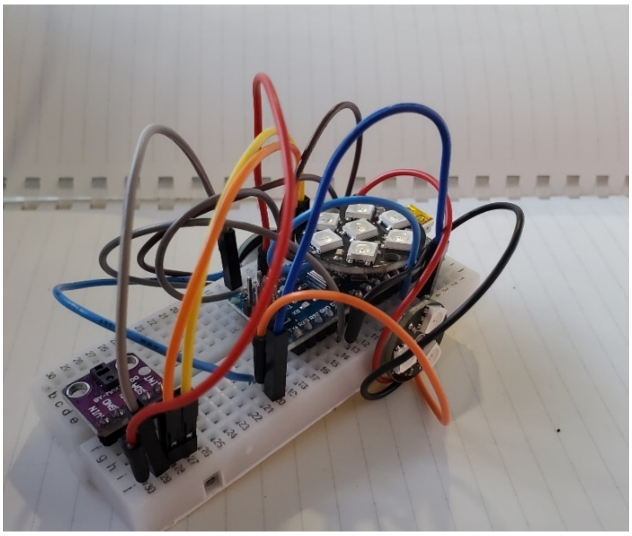


Figure 18 The electronic component to be placed inside the potting

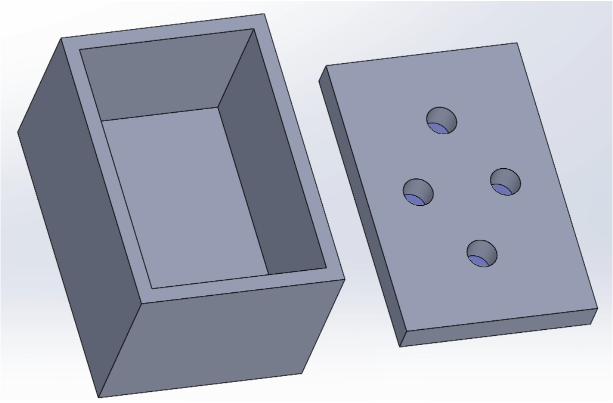


Figure 19 SolidWorks drawing for the mould of the potting container

Figure 18 above shows the electronic component as aforementioned from Section 2.3.2.2 Hardware Component. The circuit will be placed into the 3D printed mould box, as illustrated in Figure 19, for the potting container. As the first version, there will be some hole to speed up the solidification of the silicone rubber. Some possible problems are suggested for this progress. For instance, the silicone rubber may block out the electronic connection between the breadboard and the wires. If this is the issue, the wires will be soldered together on a hole circuit board. If time permits, this will be further improved to be soldered with a custom-made PCB board.

Version 2

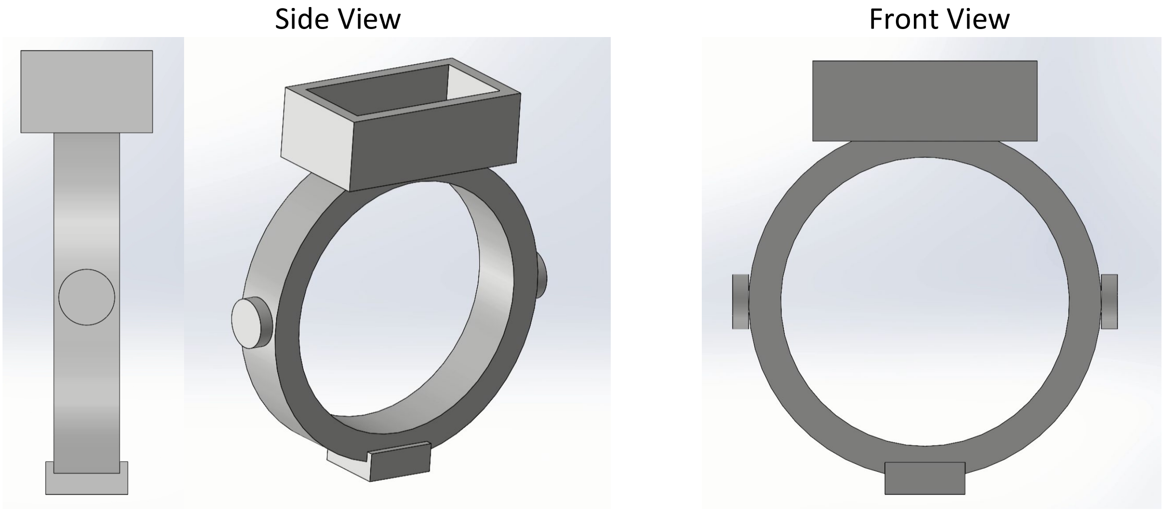


Figure 20 SolidWorks drawing for the mould of the potting container

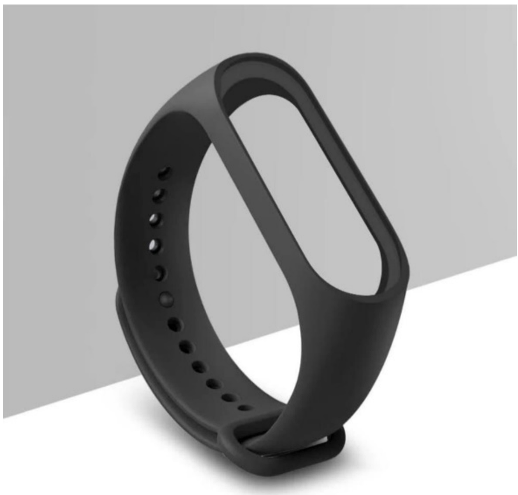


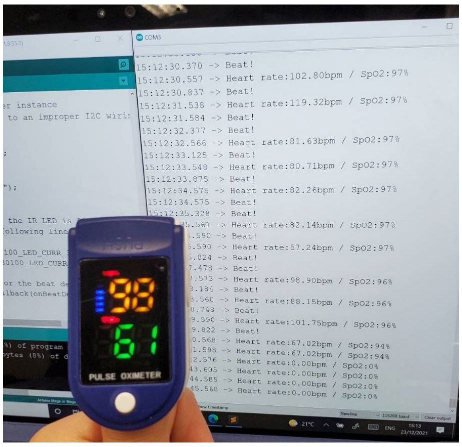
Figure 21 Waterproof bracelet for attaching the electronic parts [22]

As illustrated in Figure 20, the wearable device would be wrapped around the circular ring. The circular ring in the middle would be the bracelet. The detailed appearance of the bracelet can be referred to the Figure 21. Small cylinders are the left and right LEDs that go around the bracelet to wrap around to face a different direction. The top part is the potting area for the Arduino Nano, while the bottom part is mainly potting for the MAX30100.

2.3.3 Testing

2.3.3.1 MAX30100 accuracy testing

Testing is done towards the accuracy of the pulse oximeter. Compared the value with the external device, in Figure 22, the external record data was 98%, the MAX30100 was also arriving a data of 94-97%. This shows that the values are about ± 5% difference. This concludes that the result from the MAX30100 is quite consistent.



```
er instance
 to an improper I2C wirir
;
");
 the IR LED is
following line
100_LED_CURR_
0100_LED_CURR
or the beat de
llback(onBeatDe
%) of program
ytes (8%) of d
```

```
15:12:30.370 -> Beat!
15:12:30.557 -> Heart rate:102.80bpm / SpO2:97%
15:12:30.837 -> Beat!
15:12:31.538 -> Heart rate:119.32bpm / SpO2:97%
15:12:31.584 -> Beat!
15:12:32.377 -> Beat!
15:12:32.566 -> Heart rate:81.63bpm / SpO2:97%
15:12:33.125 -> Beat!
15:12:33.548 -> Heart rate:80.71bpm / SpO2:97%
15:12:33.875 -> Beat!
15:12:34.575 -> Heart rate:82.26bpm / SpO2:97%
15:12:34.575 -> Beat!
15:12:35.328 -> Beat!
      35.561 -> Heart rate:82.14bpm / SpO2:97%
        .590 -> Beat!
        .590 -> Heart rate:57.24bpm / SpO2:97%
        .824 -> Beat!
       7.478 -> Beat!
       7.573 -> Heart rate:98.90bpm / SpO2:96%
       3.184 -> Beat!
       8.560 -> Heart rate:88.15bpm / SpO2:96%
       8.748 -> Beat!
       9.590 -> Heart rate:101.75bpm / SpO2:96%
       9.822 -> Beat!
       0.568 -> Heart rate:67.02bpm / SpO2:94%
       1.598 -> Heart rate:67.02bpm / SpO2:94%
      2.576 -> Heart rate:0.00bpm / SpO2:0%
      43.605 -> Heart rate:0.00bpm / SpO2:0%
      44.585 -> Heart rate:0.00bpm / SpO2:0%
      45.568 -> Heart rate:0.00bpm / SpO2:0%
```

Figure 22 Record of the external pulse oximeter and MAX30100

2.3.3.2 Water testing

The first waterproof test is conducted with a water basin. The testing method is as follows:

1. Placing two pieces of dry tissue into the lock-n-lock.

2. Place the lock-n-lock under the water.
3. Place the lunch box into a water basin that has sufficient water to cover the entire lock-n-lock.
4. Rotate the box for 3 rounds to ensure no side is slipping the water in.
5. Check if the water gets wet or not.
6. If not, the test will be repeated by putting a plastic bag with two pieces of dry tissue inside.

After passing the two attempts of water-proof testing, the wearable device will be used for further testing with the image processing sub-system.

1st Water Testing

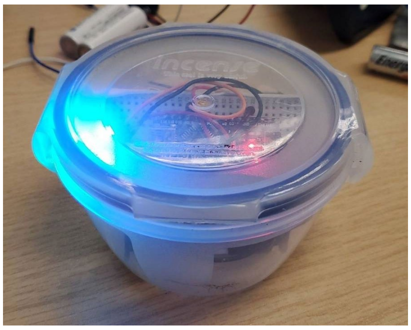


Figure 23 Wearable device in the circular lock-n-lock

The first test is conducted by the circular lock-n-lock shown in Figure 23. However, after the first attempt, bubbles are coming out from the top of the lock-n-lock. This indicates that it is not an air seal and not a water seal. With this observation, the tissue got wet as shown in Figure 24.

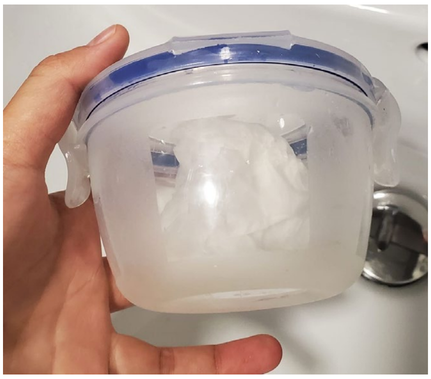


Figure 24 Test result for the circular lock-n-lock

2nd Water testing

The second water testing was also done by a lock-n-lock box. This time, the tissue was dry for both the first and the re-test. Thus, the box was further brought into the water pool outside LT-J for water testing. Placing the lock-n-lock under 50cm water with weight and rotation, the tissue inside was still maintained in a dry condition. Thus, the test was passed. As shown in

Figure 25, some metal screws and nuts are placed to increase the weight to sink the lock-n-lock to create some pressure to counteract the buoyancy. The entire electronic component was then placed into the box and started our image testing. Further testing details are included in Section 2.4.2.

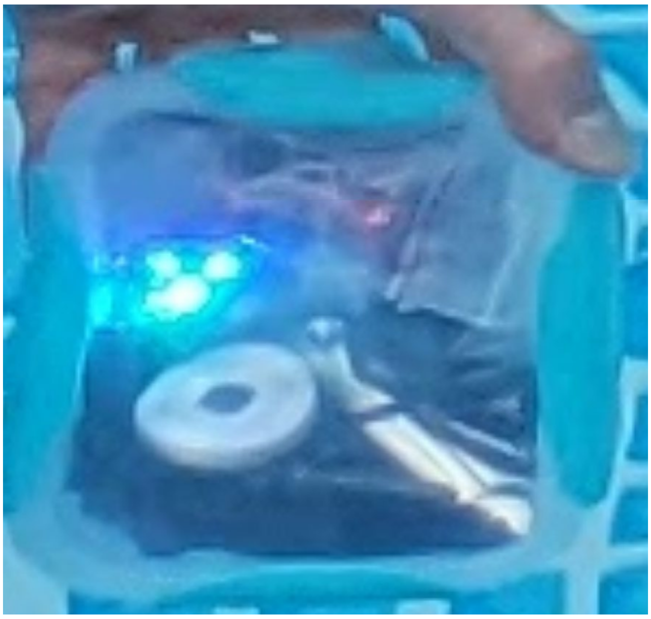


Figure 25 2nd Water test for the wearable device

2.4 Image Processing Sub-system

2.4.1 Design

The sub-system consists of 3 algorithms, namely pool corner detection, perspective transformation and LED detection algorithm.

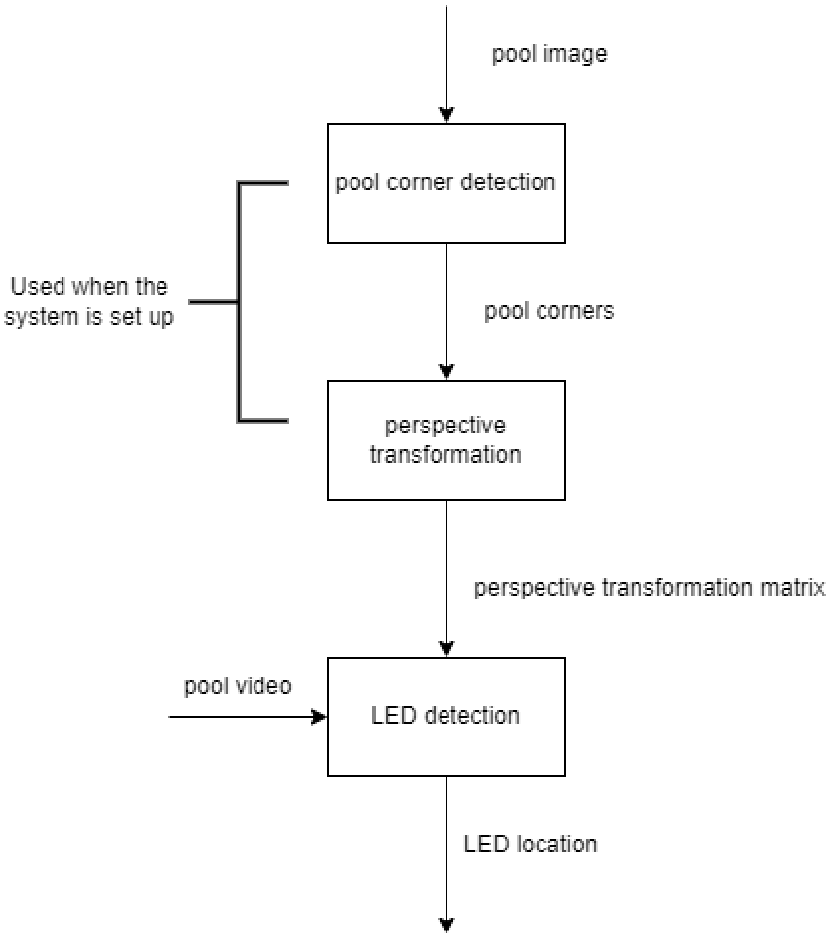

pool image
pool corner detection
Used when the system is set up
pool corners
perspective transformation
perspective transformation matrix
pool video
LED detection
LED location

Figure 26 Flow of the three algorithms

The pool corner detection algorithm takes the pool image as input and outputs the 4 corners of the pool. The output is a 4 * 2 list of lists of 32-bit float numbers. The four lists are the four corners of the pool in the order of top left, top right, bottom right and bottom left. Each list has two elements, which are the x and y coordinate respectively.

The perspective transformation algorithm takes the four corners of the pool as input and outputs a perspective transformation matrix. The matrix will then be applied to the pool video, as shown in Figure 26, to perform perspective transform to locate accurate coordinates.

The LED detection algorithm takes the perspective transformation matrix and the pool video as inputs and outputs the coordinates of the LED light as a tuple of 32-bit float numbers. The coordinates will be relative to the top left corner of the pool, i.e. (0, 0) is top left, (width – 1, height – 1) is bottom right.

Both the corner detection and the perspective transformation algorithms are used once for setting up the system.

2.4.2 Implementation

In the following section, we will talk about the implementation of the algorithms with Python, OpenCV (cv) and Numpy (np) libraries.

2.4.2.1 Pool Corner Detection

We have tried two approaches to detect the corners: Harris corner detection and Hough transformation.

Harris Corner Detection

```
pool_corner_detection_with_harris(pool_image):
1       denoised_image <- gaussianBlur(pool_image)
2       grey <- convertToGrey(pool_image)
3       corners <- cornerHarris(grey)
4       final_corners <-extractTop4Corners(corners)
5       final_corners <- Order(final_corners)
6       Return final_corners
```

Figure 27 Pseudo code of the pool corner detection algorithm using Harris corner detection

As shown in Figure 27, the algorithm starts by denoising the image with a gaussian filter. We use the GaussianBlur function provided by OpenCV with a kernel size of 11 * 11. The image is then converted to grey scale using cv.cvtColor.

After pre-processing the image, we apply cv.cornerHarris on the image. This function applies the Harris corner detector to the image. A 2 * 2 covariance matrix M is calculated for each pixel (x, y) over a block of neighbourhood. Then a characteristic score is calculated for each pixel (x, y) using the following formula:

$$dst(x,\, y) = \det M - k(trace(M))^2$$

where trace(M) is the sum of the eigenvalues of M [23].

Line 5 extracts the top 4 coordinates that have the highest score. Those are believed to be the corners of the pool. The function returns the 4 coordinates after ordering them into the top left, top right, bottom right, bottom left order.

However, this approach was not successful. Since there are many objects in the scene which may have more significant corners than the pool. The algorithm is always misled and gets those irrelevant corners. Therefore, we rejected this approach and tried another method.

Corner Detection with Hough Transformation

In the last section we mentioned that the surrounding objects mislead the algorithm to produce unexpected results. For this approach, we would like to ignore all other objects except the pool. We believe the pool has a blue colour within a range of hue values. Therefore, we would filter the image to extract the blue area out. The large blue area in the image is believed to be the pool. Since the pool is a rectangle, it should look like a quadrilateral on the image. Then we use Hough Transformation to find out the four sides of the pool. The four intersection points, which are the four corners, will then be calculated using the four lines.

```
pool_corner_detection_with_hough(pool_image):
1       denoised_image <- gaussianBlur(pool_image)
2       hsv <- ConvertToHSV(pool_image)
3       upper_limit <- 160
4       lower_limit <- 100
5       masked_image <- filter(hsv, lower_limit, upper_limit)
6       masked_image <- erode(masked_image)
7       edge <- canny(masked_image)
8       lines <- hough(edge)
9       strong_lines <- findStrongLines(lines)
10      corners <- findIntersects(strong_lines)
11      final_corners <- Order(final_corners)
12      Return corners
```

Figure 28 Pseudo code of the pool corner detection algorithm using Hough Transformation

As shown in Figure 28, it starts by denoising the image and converting it to a HSV image. Then it masks out the blue pixels with a hue value between 100 and 160 in Line 5. The values (100 and 160) are obtained by investigating the hue value of several swimming pool pictures with Paint. After that, the masked image is eroded using cv.erode with a kernel size of (5, 5) and 10 iterations to remove the tiny blue clusters in the image. In Line 7, an edged image is obtained using cv.Canny with 300 and 500 as the two thresholds and aperture size of (5, 5).

The edged image is then passed to the cv.HoughLines function to obtain the lines. The lines are expressed in the Polar system [24], i.e.

$$r = x \ \ cos\theta + y \ \ sin\theta$$

The result of the function will be a list of r and θ . However, not every line will be taken into the result. The function only considers those lines which have intersections larger than a threshold. For our algorithm, we set the distance resolution to 1 pixel, angle resolution to 1 degree and threshold to 70.

The lines the algorithm obtain include a lot of similar lines, i.e., lines which have similar r or θ . We want to find out the four most distinct lines from these lines. In Line9, it loops over all the lines until it finds the four lines with distinct r and θ and stores them in a list.

In Line10, it loops over the strong lines list and calculates the intersection points of each pair of lines. It is done by solving a linear matrix equation. We use np.linalg.solve to solve the equation. Then, it returns the result after ordering them.

This approach finds out the 4 corners of the pool successfully. However, if there is another big blue object in the image, the sky, for example, the result will probably be disturbed.

Therefore, when the user uses the corner detection function, he/she should avoid large blue objects in the scene. Still, we will use this approach in our project.

2.4.2.2 Perspective Transformation

This algorithm takes advantage of functions given by OpenCV to perform perspective transformation. Given the four points of the pool in the image (src), our algorithm finds out the four corresponding corners in the warped image (dst). A perspective transformation matrix is then calculated based on these eight points [25]. The matrix should suffice the relation:

$$(t_i x_i{'} \; t_i y_i{'} \; t_i)^T = matrix \cdot (x_i \; y_i \; 1)^T$$

where

$$dst(i) = (x_i' \, , \, y_i'), \; src(i) = (x_i \, , \, y_i), \; for \, i = 0, \, 1, \, 2, \, 3$$

```
perspective_transform(tl, tr, br, bl):
1    width_1 <- sqrt((bl[0] - br[0]) ^ 2 + (bl[1] - br[1]) ^ 2)
2    width_2 <- sqrt((tl[0] - tr[0]) ^ 2 + (tl[1] - tr[1]) ^ 2)
3    width <- max(width_1, width_2)
4
5    height_1 <- sqrt((tl[0] - bl[0]) ^ 2 + (tl[1] - bl[1]) ^ 2)
6    height_2 <- sqrt((tr[0] - br[0]) ^ 2 + (tr[1] - br[1]) ^ 2)
7    height <- max(height_1, height_2)
8
9    pts <- [tl, tr, br, bl]
10   dst <- [[0, 0], [width – 1, 0], [width –1 , height – 1], [0, height – 1]]
11   transform_matrix <- getPerspectiveTransform(pts, dst)
12   Return transform_matrix
```

Figure 29 Pseudo code of the perspective transformation algorithm

In Line 1 and 2 of Figure 29, the distances between the bottom left and bottom right corner, and top left and top right corner are calculated. The longer one will be taken as the width of the image after perspective transformation, as shown in Line 3. Line 5 to 7 serve a similar purpose, which calculates the height of the perspective transformed image.

Then, two lists that store the four corners of the input image and the perspective transformed image are declared in Line 9 and 10.

In Line 11, we use cv.getPerspectiveTransform to obtain the perspective transformation matrix. The matrix is solved using Gaussian elimination with the optimal pivot chosen. Then, it returns the matrix.

2.4.2.3 LED Detection

We have tried two approaches to detect the LED light: template matching and multiscale-template matching.

LED Detection with Template Matching

We locate the LED with template matching. Template matching is a technique to locate an object in an image like a given template. The template is slid over the whole image to calculate a similarity score at each pixel. The location with the best similarity score will be recognized as the object location [26].

```
led_detection_with_template_matching(image, template):
1    score <- matchTemplate(image, template)
2    best_loc <- getBestScoreLoc(score)
3    Return best_loc
```

Figure 30 Pseudo code of the LED detection algorithm with template matching

Line 1 of Figure 30 uses cv.matchTemplate to calculate a similarity score at each pixel of the image. We use the normalized square difference as a similarity score. Thus, the location with the best score should have the smallest similarity score. In Line 2, we get the best score location, where the score is the least. It then returns the best score location.

This simple approach works best when the object in the image is of similar size to the template. However, if the object is farther away, it is smaller, which makes it does not match the size of the template. As a result, it may not detect the object accurately. Therefore, we tried the following approach.

LED Detection with Multiscale-Template Matching

To detect the object at different distances, i.e., the different sizes on the image, we scale the template for different sizes and perform template matching for each scale. This is known as multiscale-template matching.

```
led_detection_with_multiscale_template_matching(image, template):
scale_factor <- 0.2, step <- (1.0 - 0.2)/ 20
While scale_factor <= 1.0
  temp_image <- image
  scaled_template <- resize(template, scale_factor)
  score <- matchTemplate(image, scaled_template)
  best_loc <- getBestScoreLoc(score)
  rect_bottom_right <- (best_loc[0] + scaled_template.width, best_loc[1] + scaled_template. height)
  drawRectangle(temp_image, best_loc, rect_bottom_right)
  displayImage(temp_image)
  scale_factor <- scale_factor + step
End
```

Figure 31 Pseudo code of the LED detection algorithm with multiscale-template matching

This approach works the same way as the previous one, except there is a loop to increment the scale_factor, resize the template with that factor and perform template matching with the scaled template, according to Figure 31. This algorithm will loop from 0.2 to 1.0 for 20 iterations. We are still fine-tuning these numbers.

This approach can detect objects with different sizes on the image successfully. We will continue to fine-tune this approach. Things we are working on include combining the results of each iteration into one location. We will set a threshold to differentiate successful detections and unsuccessful detections. Close locations will be combined to one location. We may also explore ways to detect blinking LED light.

2.4.2 Testing

For corner detection and perspective transformation algorithms, we tested them with swimming pool images. The swimming pools are rectangular, and the four corners are visible in the image. The pool is filled with water which appears blue in the image. Figure 32 is an example of the testing data.

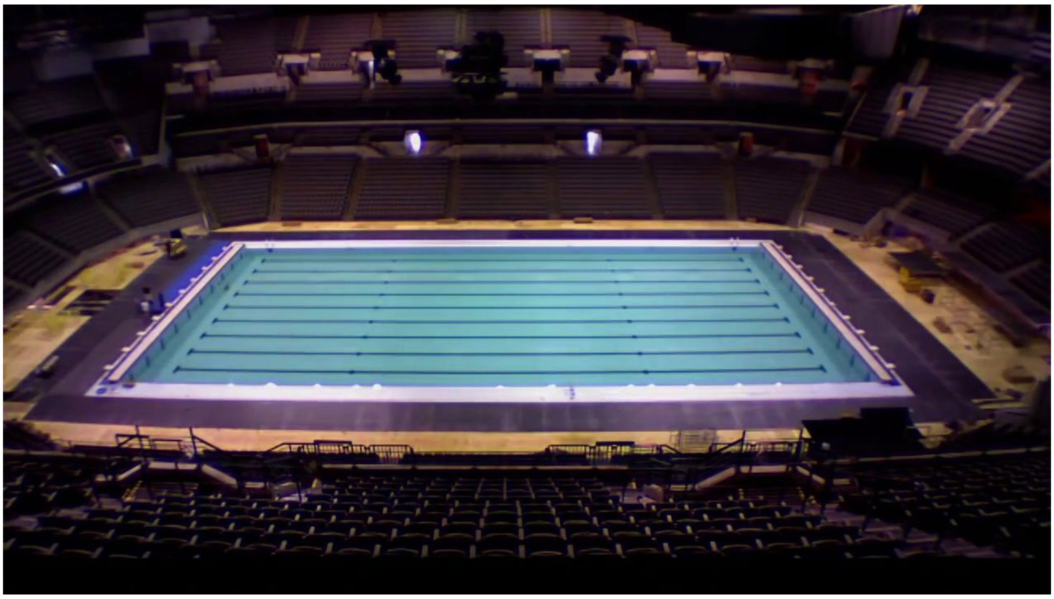


Figure 32 Example of testing data for corner detection and perspective transformation

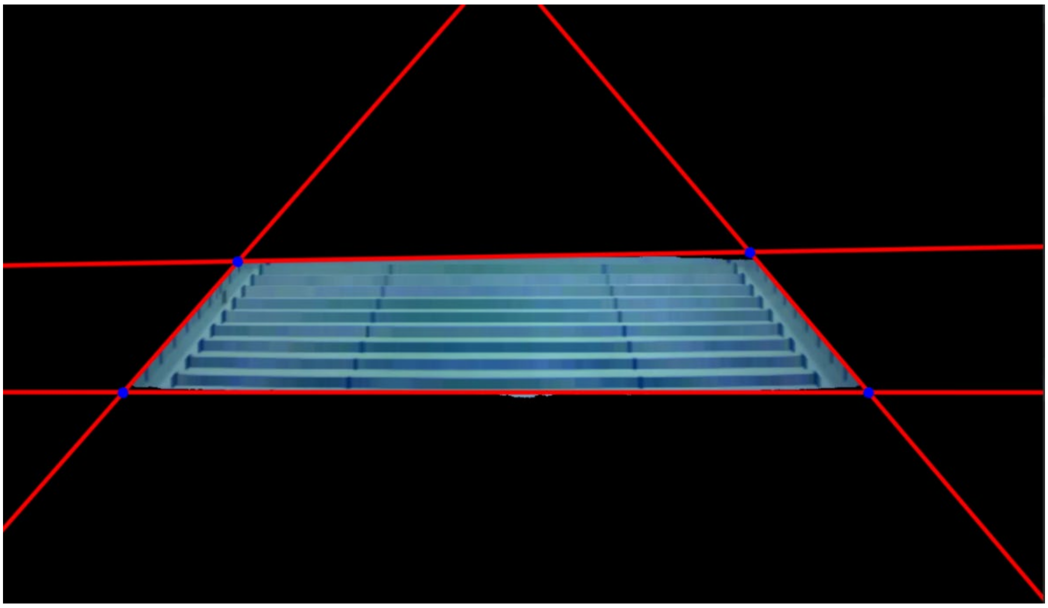


Figure 33 Image after corner detection

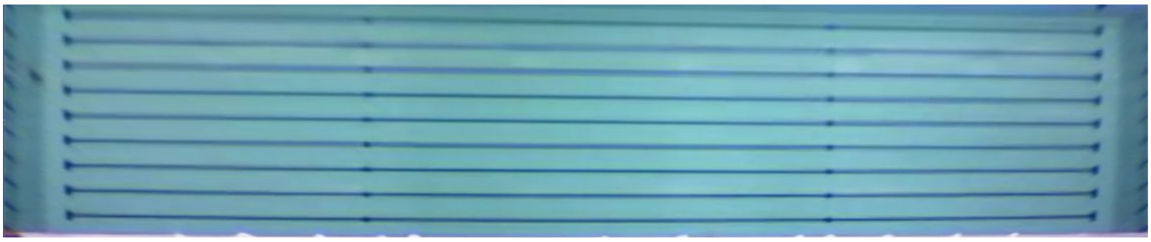


Figure 34 Image after perspective transformation

Figure 33 is the image after corner detection. The four red lines are the Hough lines calculated by the Hough transformation. The four blue dots are the intersections of the four lines. Figure 34 is the image after perspective transformation. The obtained image is a “top-down” view of the pool in the original image.

The algorithms work fine with this image, yet we still need more images to verify the accuracy of the algorithms. Since there are not so many valid (rectangular swimming pool with four corners visible) images, we would take more pictures once we could access a standard swimming pool.

For the LED detection algorithm, we have tested it with images of the LED light at different distances. Since using images is easier to debug the program, we tested the algorithm using images instead of videos. The following Figure 35, Figure 36, Figure 37, Figure 38 are some examples.

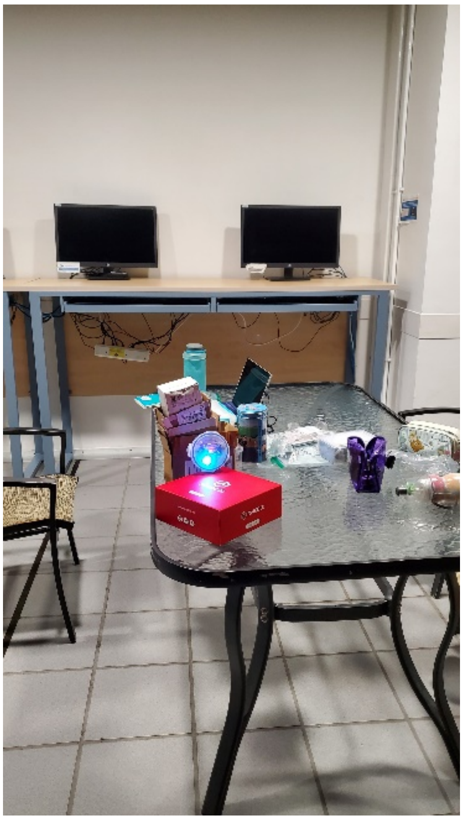


Figure 35 LED at 1m away

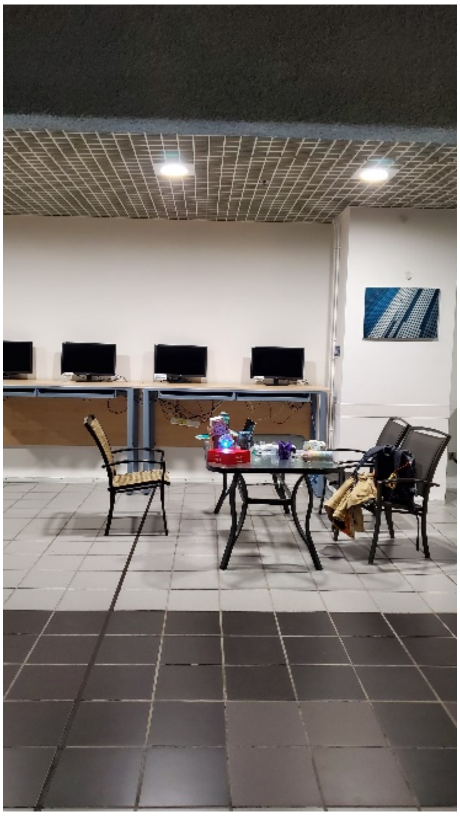


Figure 36 LED at 3m away

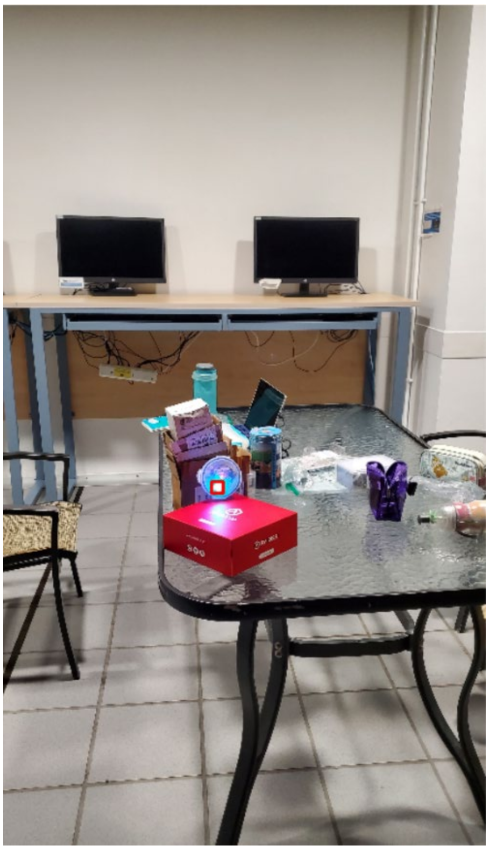


Figure 37 Detected LED at 1m away

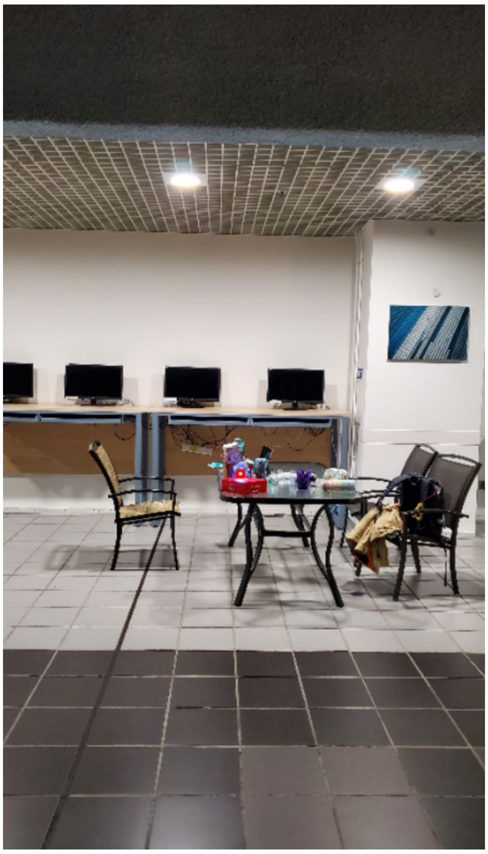


Figure 38 Detected LED at 3m away

Figure 35 and Figure 36 show the LED at 1m and 3m respectively while Figure 37 and Figure 38 show the detected LED at the corresponding distance. The detect LED is marked with a red rectangle as shown in Figure 37 and Figure 38.

Under the pandemic, all swimming pools were shut down. Yet, we could still access a small pool (3m * 2m) in HKUST. We also used that pool to test the algorithm. Below are some example images.

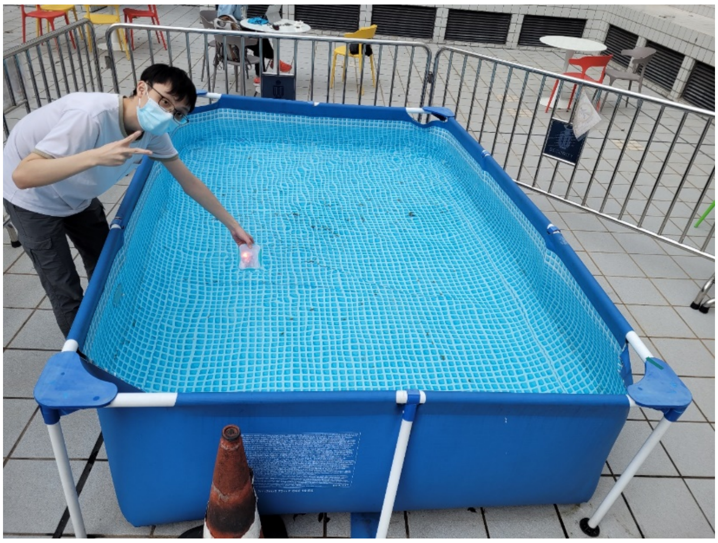


Figure 39 Red LED above the water surface at 1m

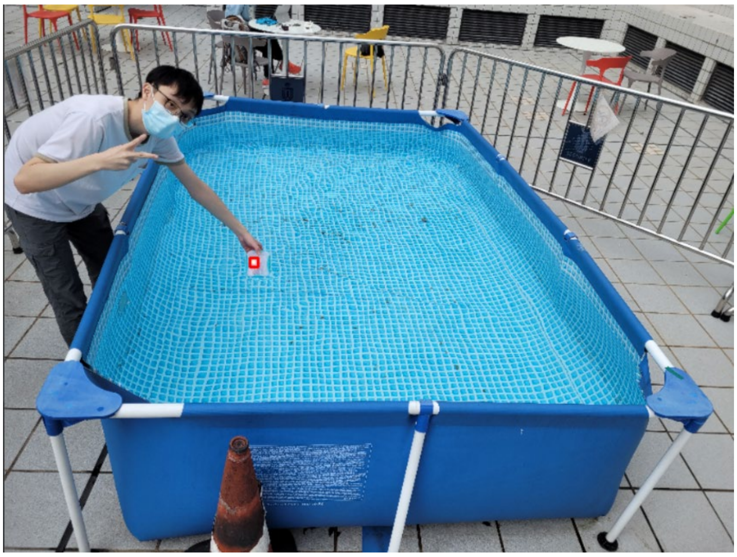


Figure 40 Detected red LED above the water surface at 1m

Figure 39 and Figure 40 show the LED and detected LED above the water surface at 1m away from the bottom pool side respectively. The detected LED is located with a red rectangle in Figure 40. Unlike the previous images, we used red instead of blue LED since we want to avoid the similar colour of the pool.

The algorithm works fine when the LED light is close to the camera. Yet, we do not know how well it performs when farther away in the pool. We need images that have the LED farther away in a standard swimming pool to check the accuracy of the algorithm. The images will be the LED light placed at different distances (1m, 2m, … 25m) and both beneath and above water surface images will be taken.

2.5 Server Sub-system

This system is responsible for integrating sub-systems and providing an interface for the lifeguard.

2.5.1 Design

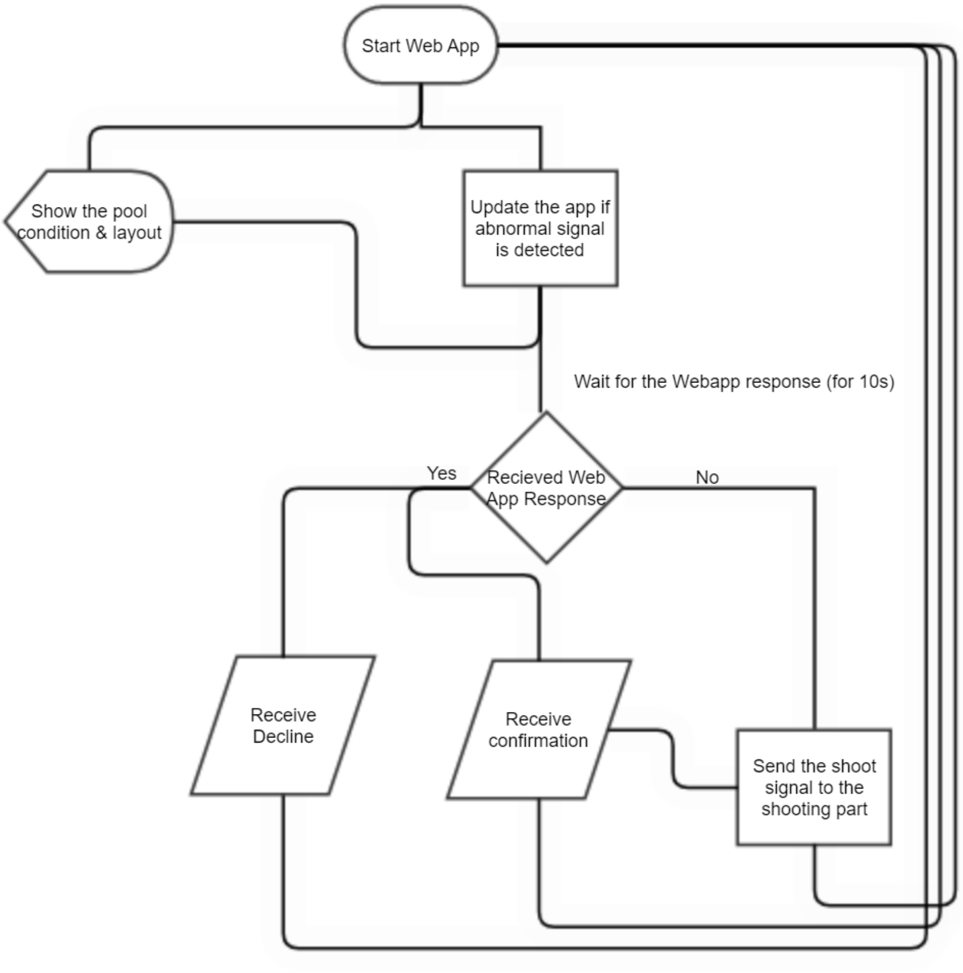

Start Web App
Show the pool condition & layout
Update the app if abnormal signal is detected
Wait for the Webapp response (for 10s)
Yes
Recieved Web App Response
No
Receive Decline
Receive confirmation
Send the shoot signal to the shooting part

Figure 41 Flowchart representation of the drown-alarming sub-system

Figure 41 illustrates the flow of how the web application will function. Upon receiving the warning signal, the application will be updated with the victim's location and displayed on the screen. An alarm signal will also be sent out to catch the lifeguard attention.

Then, there will be a 10-second countdown for the lifeguard to confirm whether there is a drowning detection. This would allow the lifeguard to make the professional decision. If the lifeguard has confirmed the drown condition or no response is received within 10 seconds, the Lifebuoy throwing sub-system will follow up to throw the lifebuoy.

Moreover, there would also be an input system for the lifeguard to enter the location for the drowning case. This allows the lifeguard sitting on the lifeguard tower to be able to control the Lifebuoy Mechanism in case the detection system missed any drown victim, but the lifeguard is not able to reach the victim due to not being able to leave the lifeguard tower.

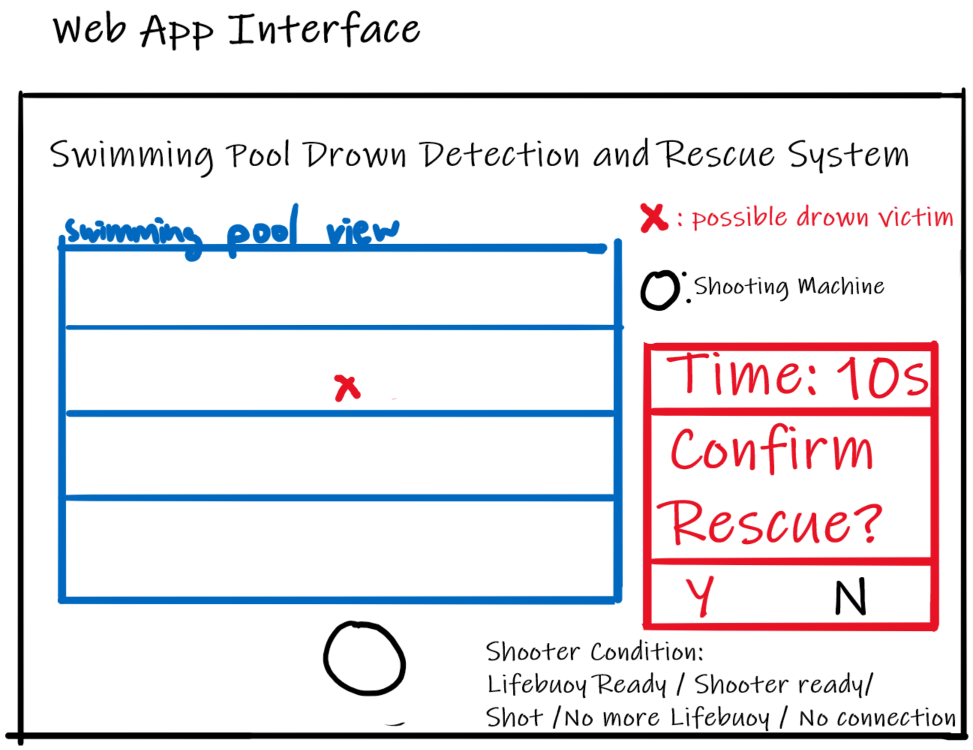

Web App Interface
Swimming Pool Drown Detection and Rescue System
swimming pool view
X : possible drown victim
O: Shooting Machine
Time: 10s
Confirm Rescue?
Y
N
Shooter Condition:
Lifebuoy Ready / Shooter ready/
Shot /No more Lifebuoy / No connection

Figure 42 Preliminary UI Interface for the web application

For the web application, due to the design factor of being easy to use and not causing an additional burden for the lifeguard, a simplistic approach is taken. As shown in Figure 42 only the pool view, the location of the victim and the shooter stage (prepared, no lifebuoy, shoot) will be shown.

The monitoring webpage can be accessed by smartphone, tablet, or laptop, but touch screen devices will be the main service platform. The user interface will be designed as touch-screen-friendly as it has a low learning slope and is intuitive to interact with.

The website will contain two webpages to serve different purposes:

- Monitor Page: The footage captured will display here with a notation of alert. The lifeguard can answer the prompt to stop the false alarm (depending on the setting) and touch the screen to do override shooting.
- Setting Page: Set the pool size, pool edge, and config the alarm procedure.

2.5.2 Implementation

In the implementation, the system can further break apart into the server itself and the connection between different sub-systems, which are 1) the server, 2) webcam, 3) adaptation to image processing algorithm, 4) web App, 5) communication to the robot as shown in Figure 43. The following will explain their design and implementation in detail.

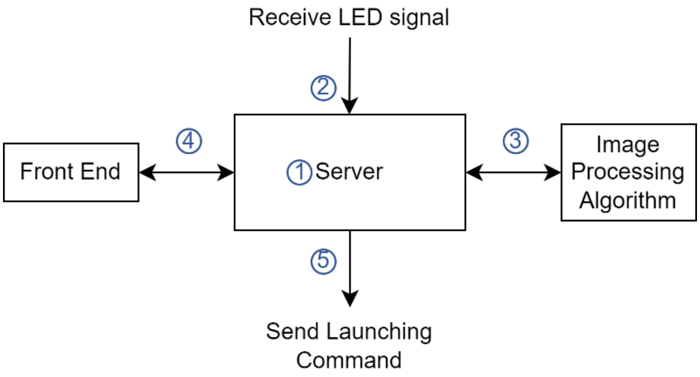


Figure 43 Simplified server architecture

1. The Server

The server is running on a VM from HKUST. The operating system is Ubuntu 20.04.1 LTS. Since the software currently used is available on a multi-platform, we also set up an identical server on the Windows 10 platform for testing purposes. There is a minor issue running on Windows 10. The npm package used for capturing the webcam into an image file is slow. It can be fixed by using another package, but we keep the package for consistency of the code between the Linux platform and the Windows platform for development. Therefore, it will take not much further development to run the server on different platforms to adapt to already present swimming pool equipment.

The backend logic runs by nodeJS, and we use different packages to implement functions and adapt to sub-systems. Mainly we use the following packages: “node-webcam” for webcam capturing, “child_process” for Python integration, “express” for web-related logic, and “socket.io” for streaming webcam to the client.

2. Webcam

We use an Android app DroidCam for wireless webcam input to the server. It provides 640x480 resolution for the free version and 1920×1080 resolution for the paid version DroidCamX Pro, and it is available on both Linux and Windows. We choose this method as our webcam input because it is portable, easy to set up, low cost, and easier to find suitable mounts for a smartphone. With that said, the whole system also supports any webcam input, such as an IP camera, a wired DSRL camera for high-quality video, or OBS virtual camera as the adhesive for older systems.

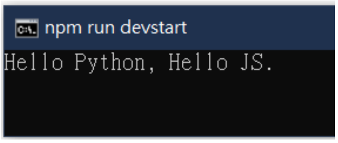
3. Adaptation to the image processing algorithm

Since the server uses node.js, the image processing algorithm uses Python. We need to bridge the two languages to allow the server to use the algorithm. This function is under development, and we are negotiating the data and format to be transferred between the two systems. A Python read-only config file will be used to store the settings from the user, which will include the pool size, user-defined corner, robot origin, and timer setting. The implementation is by using the "child_process" package. It provides node.js a way to call a Python program with arguments and receive the output by flushing system stdout. Figure 44 shows the basic testing on calling the Python and receiving the output. First, we call a Python program using the "child_process" package in node js, then in Python, we print "Hello Python, Hello JS" and flush the stdout. "child_process" will catch this event and print it to the terminal.

```
//code on node js
const spawn = require("child_process").spawn;
const pythonProcess = spawn('python',["./public/test.py"]);
//stdout event handler
pythonProcess.stdout.on('data', (data) => {
      console.log(data.toString());
});
```

```
#code on python
print("Hello Python, Hello JS.")
sys.stdout.flush()
```

Output from terminal



```
Hello Python, Hello JS.
```

Figure 44 Code of Adaptation to the image processing algorithm

4. Web App

Back end

The backend is handled by nodeJS package "Express" and uses "EJS" as the template engine to provide website services. Express is a web backend framework of node.js that features routing function, logic handling, and generating front-end HTML by EJS. EJS is like PHP, EJS is a template that can take in parameters and generate a customized webpage when a request is received.

The webcam streaming functionality is implemented by the "socket.io" package. First, the client and server will establish a socket.io connection. Socket.io is an instance of WebSocket, used for real-time data transfer in the HTTP service. Then, "node-webcam" captures a frame from webcam input and returns base64 data. When node-webcam captures a frame, it will send it out to the client by socket.io.

Front End

For the front end, we use HTML, CSS, JavaScript, SVG, and jQuery to develop. Here we will describe the streaming function, stream display function, UI design, and iOS optimization.

Webcam streaming function

Before the data comes from the server, it displays an animated gif indicating that the stream is loading. Figure 45 shows the implementation of webcam streaming on front end. When new image data comes in from the socket connection, JavaScript will update the current image with the new image data. We also obtain the size of the image as it will be used when the user inputs the shooting position, but the size is usually constant in our use case, therefore we comment out the function to reduce calculation.

```
const socket = io();//Start socket connection
const image = document.getElementById("webcam");//The image node
socket.on("image",(data)=>{//When new data comes in, this will run
            if(data){
            //getImgSize(data);
            image.setAttribute("href",data)
      }
});
```

Figure 45 Code of webcam update on client side

Stream Display

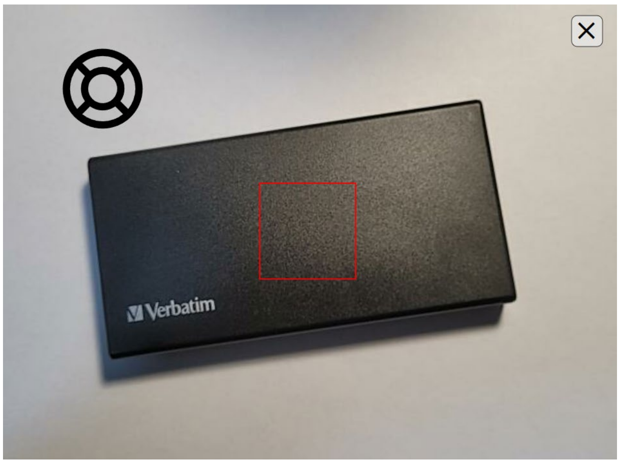
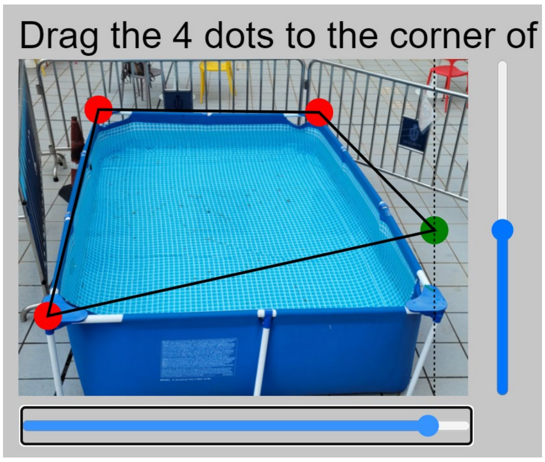


Figure 46 Monitor page



Drag the 4 dots to the corner of

Figure 47 Cropped setting page

This part presents the image data received from the backend. We use an SVG image node to hold the data. Every time the client receives valid image data from socket.io, it will update the "href" attribute of the image node, therefore updating the latest image from the webcam. The reason for using SVG image node instead of HTML "<img>" tag, is because SVG allows us to draw different shapes on top of the image without obstacles to the position input from the user. In Figure 46, the monitor page demonstrates drawing a detection box and float throwing SVG animation. Another advantage of using SVG is the animation of SVG is handled by the browser, therefore it is smoother and will not consume the resource of the single-threaded JavaScript engine. In Figure 47, the setting page demonstrates the combined use of SVG and JavaScript. JavaScript draws auxiliary lines in SVG to assist the user positions the dots precisely.

UI design

The UI is optimized for 4:3 ratio large screen devices, for example, most iPad devices. It is because most cameras are capturing in a 4:3 ratio, using a 4:3 screen can maximize the use of both camera and the screen. As shown in Figure 48, the screen is fully utilized on iPad mini. But since we applied responsive web design, Figure 49 demonstrate it is also flexible to other screen ratios, and the gap will fill with the background colour.

The main screen is minimalistic that only includes the video stream and a universal button that can navigate between the monitoring page and the setting page, this helps the lifeguard focus on monitoring the swimming pool. When drowning is detected, a color-changing rectangle from yellow to red will show on the position indicating how long the victim has been drowning. If the timer is nearly running out, an alert will pop up and ask the lifeguard for a shooting command.

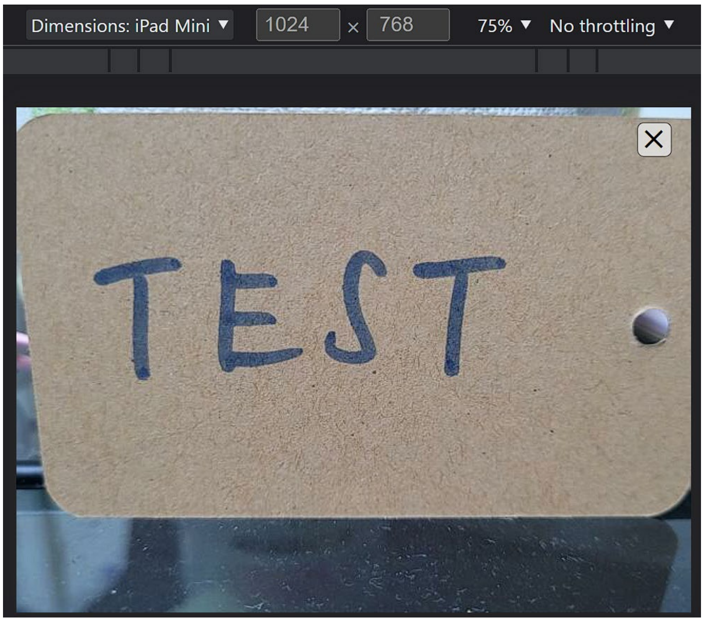

Dimensions: iPad Mini ▾ 1024 × 768 75% ▾ No throttling ▾

Figure 48 Display of monitor page on iPad mini using Chrome DevTools

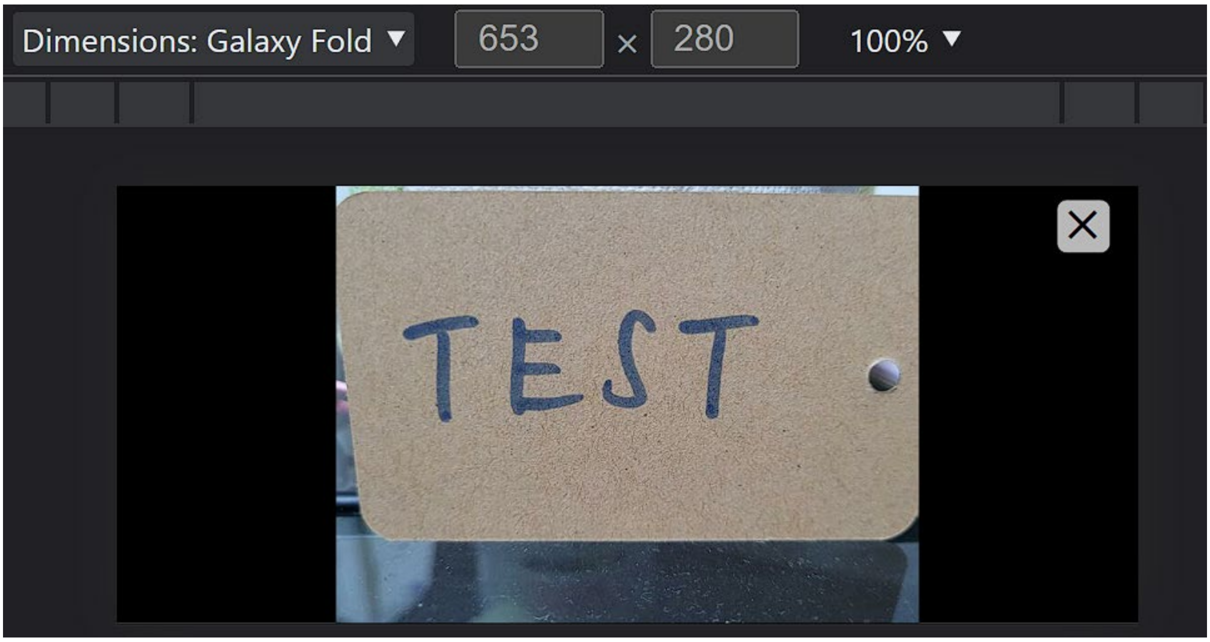
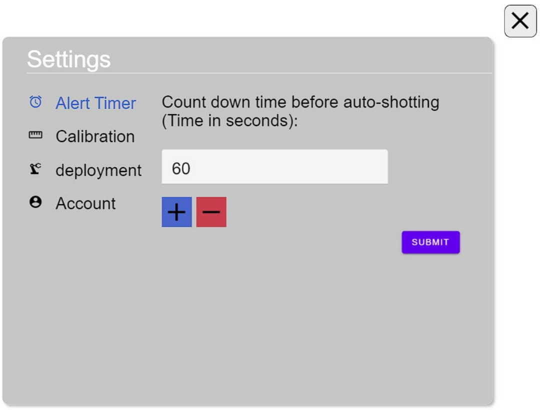


Figure 49 Display of monitor page on Galaxy Fold using Chrome DevTools

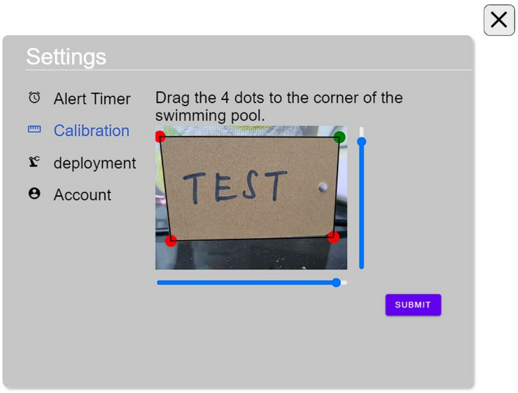
The setting page is optimized for large-screen mobile devices. In Figure 50 large, coloured buttons are used to indicate the options, which are easy to click on and intuitive. And we use horizontal and vertical range input for users to enter the X-Y of the position to avoid fingers blocking their view. When the users is adjusting the input, JavaScript draws the auxiliary lines to help users align to the pool. If the corners align correctly, the line will frame the swimming pool seamlessly as shown in Figure 51.



Count down time before auto-shotting (Time in seconds):

60

Figure 50 Setting page – Alert Timer



Drag the 4 dots to the corner of the swimming pool.

Figure 51 Setting page – Calibration

iOS optimization

First, we maximize the usage of the screen. Usually, browsers on iOS like Safari and Chrome have a navigation bar at the top that cannot be removed by the webpage, which results in a black bar around the screen. Therefore, we use a specific browser "Full-Screen Web Browser App" to avoid the navigation bar.

Second, we eliminate the 300ms button delay on iOS. To avoid delay, we identify the user-agent right after the page loaded. If it is a mobile device, we insert the "ontouchstart" event handler, else we insert the "onclick" event handler.

Third, we disable the double-click zoom feature on mobile devices by setting the meta element. Since we will click multiple times on a button on the setting page, it will trigger the double-click zoom function, which is inconvenient for the user.

Different with design

1. The monitor page is further simplified, to provide maxim usage of the screen, and minimum distraction to the lifeguard.
2. The front-end design was intended to use React for UI development. After further findings on React, we find that the benefit of React is adding program logic to generating UI, therefore reducing repetitive codes. But our application only consists of a few pages, it is excessive to use React and will cost more loading time to load React library.

5. Communication with robot

In the communication with the robot, we use the internet as the channel and JSON as the data transfer format. After the user clicks on the swimming pool and sends a shooting command, nodeJS will run the first part of the code in Figure 52. It will calculate the projectile motion and obtain the shooting angle and velocity, then those into a JSON file accessible through the internet. The second part of Figure 52 demonstrates how the robot receives the command. The robot will fetch the JSON file through the network, decode the JSON file and turn the JSON into usable variables.

```
//code on nodejs
var shootingPara = projectileMotion(req.query.x-200,req.query.y);
var shootingData = {
      angleH: shootingPara[0]*57.2958,
      angleV: shootingPara[1]*57.2958,
      velocity: shootingPara[2],
      shoot: req.query.shoot,
      note: req.query.note
};

fs.writeFileSync(path.resolve('./public', 'shootingData.json'),
JSON.stringify(shootingData));
```

```
//code on robot
//Getting JSON
HTTPClient http;
http.begin("http://theAddressOfTheServer/test.json");
http.GET();
String JSON = http.getString();
```

```
http.end();
//Parse JSON
StaticJsonDocument<192> doc;
DeserializationError error = deserializeJson(doc, JSON);
//the error handling is cut out for better demonstration.
//Got the variables
int x = doc["x"];
int y = doc["y"];
bool shoot = doc["shoot"];
const char* note = doc["note"];
```

Figure 52 Code for robot communication

2.5.3 Testing

The user test concerns responsiveness, user-friendliness and clarity. We will time the webcam to client delay and the whole process from the guard clicking on the screen to the bot shooting the float to test the responsiveness. And we will conduct a user test by inviting five lifeguards to evaluate the user-friendliness and clarity.

2.6 Evaluation

The fundamental purpose of this project is to develop a drown detection and rescue system. Thus, the ultimate testing would be sending a random abnormal heat beat externally to trigger the progress. Then test the progress time from the signal detected to the complete inflation of the lifebuoy to be within 1 minute. The choice of 1 minute is with reference to the golden rescue time for drowning of 2 minutes [27]. We could also invite the general public to move the victim and change to a different location in the pool to test the calculation for the changed location. Additionally, we would also invite some additional suggestions and advice from 2 lifeguards during the progress to ensure some safety measure and swimming pool related matters are considered.

3. Project Planning

3.1 Project Work Plan

Year	Date	Task	Personal In Charge
2021	12 September	Internal Proposal Deadline	Loreen
	30 September	Collection for Detection Sensor Data	Don
	15 October	Functionality Testing for the Detection System	Loreen
	22 October	**Deadline for Individual Ethics Essay**	Loreen, Don, Edwood
	29 October	**Deadline for Monthly Report**	Don
	30 October	Water Testing for the Detection System	Loreen
	30 October	Testing for the Positioning System	Edwood
	15 November	Functionality Testing for Web App	Don
	26 November	**Deadline for Monthly Report**	Edwood
	30 November	Functionality Testing for the Positioning System	Edwood
	30 December	Full Testing in Swimming Pool for Detection System	Loreen
2022	**12 January**	**Deadline for Monthly Report**	Loreen
	30 January	Seek User Opinion from Lifeguards for Web Application	Don
	15 February	Combined Testing for the Detection System & Positioning System	Edwood
	28 February	Combined Testing for the Positioning System & Lifebuoy Throwing System	Loreen
	1 February	Internal Deadline for Progress Report	Loreen
	14 February	**Deadline for Progress Report**	Edwood
	6 April	Internal Deadline for Final Report	Loreen
	20 April	**Deadline for Final Report & Self-Assessment Report**	Don
	1 May	Internal Deadline for Video	Edwood
	11 May	**Deadline for Video**	Don

Table 6 Project Work Plan

With a new groupmate CHENG Hoi Wai joined at the start of October 2021, the person in charge section is updated.

3.2 Human Resources

Name	Skills / Ability
LEUNG, Lok Yan Loreen	Stronger in have an overview in a problem, Have communication skills for different parties, Understand both skills in Computer Engineering and Mechanical Engineering, Interested in testing data collection and programming 🕐 Can help in project management 🕐 Can coordinate between different departments and resources 🕐 Can integrate the mechanical and computer system 🕐 Responsible for the drown detection system
CHAN Tsz Ho	Stronger in language skills, Rich in hands on experience on Mechanical parts 🕐 Can proofread and become second reader for Computer Engineering Documents 🕐 Can direct and guide the progress of prototype building and testing
CHENG Hoi Wai	Stronger in programming skills and software documentation Experienced in learning new programming platform, especially image processing 🕐 Responsible for the image processing sub-system 🕐 Can help in software architecture documentation
CHONG Shing Tung	Rich experience in electronic components and web programming, Swimming 🕐 Can initiate the electronic control and design the web application 🕐 Can support on-scene testing
LEE Ka Hei	Experienced at design and drawing, Thinks with a careful and detailed mind Patient 🕐 Responsible for the robot design and solid work drawing for Mechanical and gives suggestion for Computer UI design 🕐 Can give second option for the user experience 🕐 Responsible for the calibration and Mechanical Debugging

Table 7 Our skills and ability

With a new groupmate CHENG Hoi Wai joined at the start of October 2021, the Human Resources Section is updated.

3.3 Project Monitoring

We will be monitoring the project using the RACI method (Responsible, Accountable, Consulted, Informed). With a new groupmate CHENG Hoi Wai (Edwood) joined at the start of October 2021, the responsibility is slightly modified and redistributed. Since some of the design and research are done before October, some entries for CHENG Hoi Wai are labelled as X.

Overall Version

Task/Name	LEUNG, Lok Yan Loreen	CHAN Tsz Ho	CHENG Hoi Wai	CHONG Shing Tung	LEE Ka Hei
Design and Research					
Communication Between Different Parties	R	A	I	I	C
Analyse on Different Topic	I	R	X	A	C
Defining Topic	C	R	I	I	A
Feasibility Study	R	C	I	A	I
Integration with CPEG/Mech Portion	A	I	C	R	C
Finalization					
Testing, Debugging, and Tunning	I	A	R	C	C
UI Improvements	C	I	C	R	A
Documentation					
Writing Proposal	A	C	X	R	I
Monthly Report	R	I	A	C	I
Progress Report	C	C	R	A	I
Writing Final Report	R	C	A	C	C
Preparing Presentation	C	I	R	A	C
Making Video Trailer	I	C	R	A	I

Table 8 Project Monitoring Table for the overall project (Both CPEG and MECH)

CPEG Version

Task/Name	LEUNG, Lok Yan Loreen	CHAN Tsz Ho	CHENG Hoi Wai	CHONG Shing Tung	LEE Ka Hei
Design and Research					
Study on ROS	A	C	X	R	I
Analyze on Different Approaches	R	C	X	A	I
Feasibility Study	R	C	X	A	I
Literature Review	A	C	X	R	I
Implementation					
Testing Electronic Component	R	C	C	A	I
Decide Detection Process of Drowning	R	I	C	A	C
Implement Detection System	R	C	A	C	I
Testing Positioning System	A	I	R	C	C
Implement Positioning System	C	I	R	A	C
Integration with Mech Portion	A	I	C	R	C
Web App Development	C	I	A	R	C
Finalization					
Implement Calibration System	R	I	C	A	C
Testing, Debugging, and Tunning	A	I	R	C	C
UI Improvements	C	I	A	R	C
Documentation					

Writing Proposal	A	C	X	R	I
Monthly Report	R	I	A	C	I
Progress Report	C	C	R	A	I
Writing Final Report	R	C	A	C	I
Preparing Presentation	C	C	R	A	I
Making Video Trailer	I	C	R	A	C

Table 9 Project Monitoring Table for the CPEG component of the project

MECH Version

Task/Name	LEUNG, Lok Yan Loreen	CHAN Tsz Ho	CHENG Hoi Wai	CHONG Shing Tung	LEE Ka Hei
Design and Research					
Feasibility Study	C	A	I	I	R
Built varies robot models	C	R	I	I	A
Literature Review	C	R	I	I	A
Analysis of Different Launching Approaches	R	C	I	I	A
Implementation					
Launcher 3D Modelling	C	A	I	I	R
Acquiring Launcher Parts	C	R	I	I	A
Implement first Launcher	C	R	I	I	A
Testing Different Launching Approaches	R	C	I	I	A
Acquiring Pan Tilt Mechanism Parts	R	A	I	I	C
Testing Pan Tilt Mechanism	R	A	I	I	C
Integration with CPEG Portion	R	A	C	C	I
Finalization					
Testing, Debugging, and Tunning	A	R	C	I	C
Documentation					
Project Planning	R	C	I	I	A
Progress Report	A	R	I	I	C
Final Report	A	R	I	I	C
Making Video Trailer	C	R	I	I	A
Oral Presentation	R	C	I	I	A

Table 10 Project Monitoring Table for the MECH component of the project

3.4 Gannt Chart

Below are the two Gannt Charts for both Computer and the Mechanical Engineering part.

CPEG version

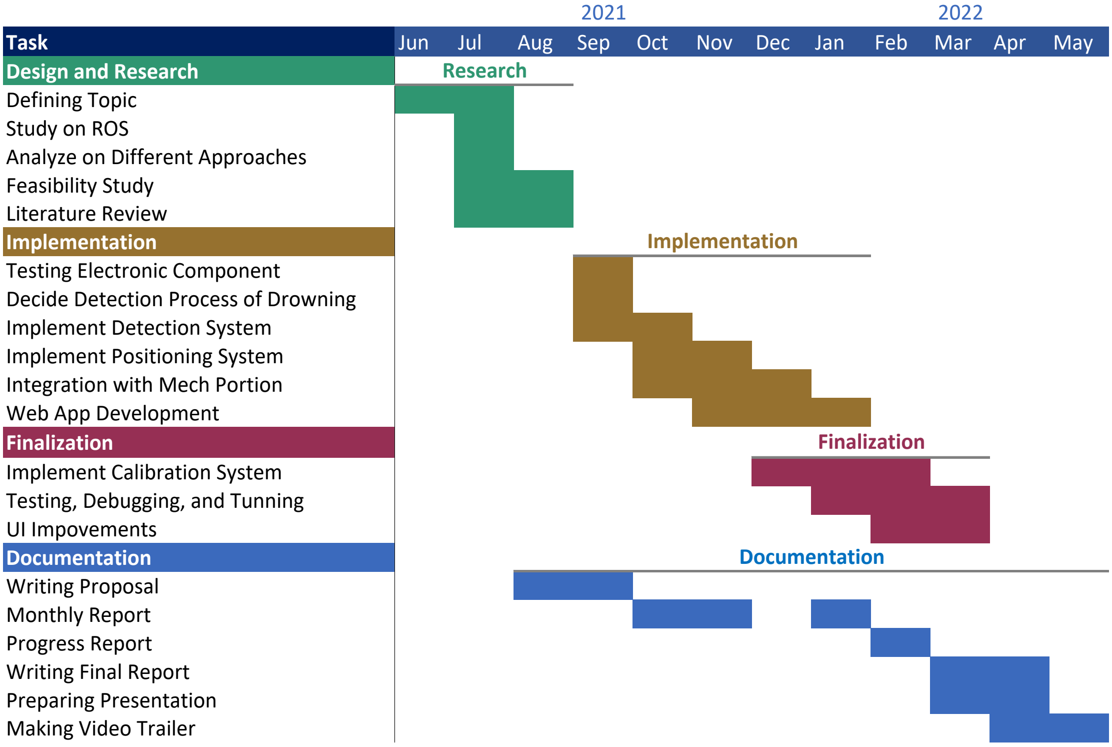


Table 11 Gannt Chart for the CPEG component of the project

MECH version

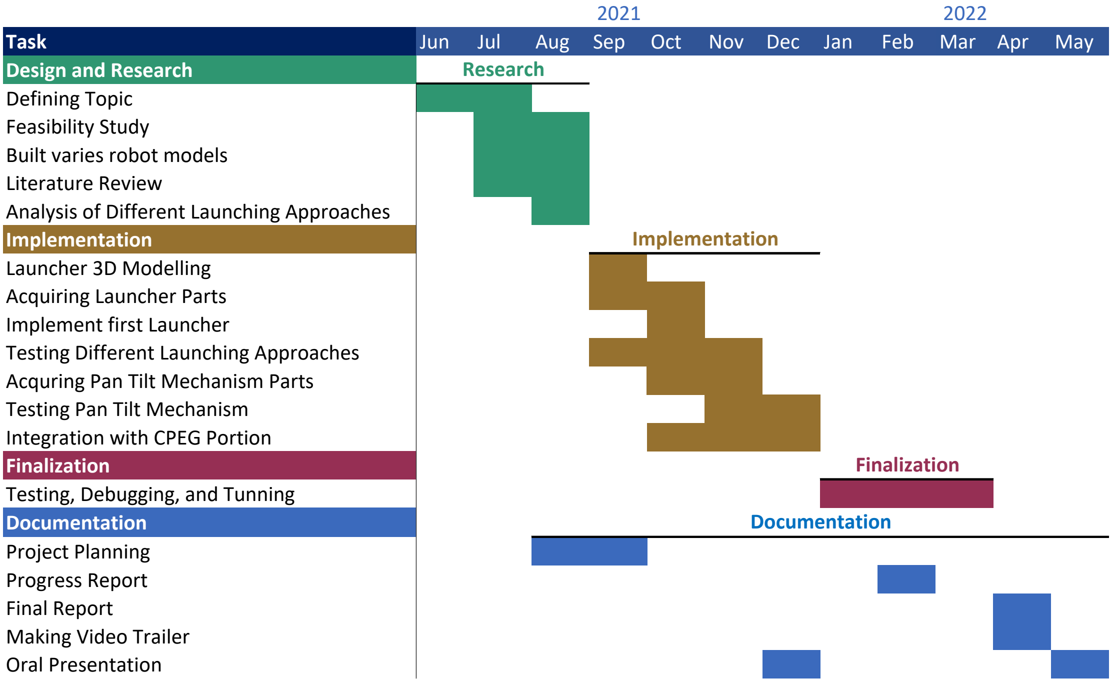


Table 12 Gannt Chart for the MECH component of the project

4. Required Hardware & Software

4.1 Hardware

Hardware	Usage	Price (HKD)	Quantity
Arduino Nano	Process data in the wearable	29	1
MAX30100	Monitor heartrate and blood oxygen level of user	19	1
Small Lithium ion Battery	Power the whole wearable device	40	1
LEDs	Positioning and alert nearby swimmers	7	4
Smart Phone	Webcam	1000	1
HKUST Server	Calculate position and host the webpage	Free	1

Table 13 List of details for the required hardware

4.2 Software

Software	Version	Usage	Price (HKD)
Arduino IDE	1.8.16	Program the Arduino	Free
Node.js	14.17.6 LTS	Server of Webpage	Free
DroidCam	6.15	Webcam Input	Free
Full Screen Web Browser App	2.1	Fullscreen web browser for iOS	Free
Python	3.10.0	Image processing algorithm	Free

Table 14 List of details for the required software

5. References

[1] “【泳灘遇溺】工會發聲明指救生員人手嚴重不足 憂泳灘遇溺事件陸續有來 ([Drowning at the Beach] The union issued a statement saying that lifeguards are seriously understaffed.It is worried more drowning incidents to be coming),” 香港經濟日報 HKET, 06 Jun 2021. [Online]. Available: https://topick.hket.com/article/2975671/%E3%80%90%E6%B3%B3%E7%81%98%E9%81%87%E6%BA%BA%E3%80%91%E5%B7%A5%E6%9C%83%E7%99%BC%E8%81%B2%E6%98%8E%E6%8C%87%E6%95%91%E7%94%9F%E5%93%A1%E4%BA%BA%E6%89%8B%E5%9A%B4%E9%87%8D%E4%B8%8D%E8%B6%B3%E3%80%80%E6%86%82%E6%B3%B. [Accessed 16 Sep 2021].

[2] “奪命泳池再揭救生員煲手機 (The life-threatening pool reveals that the lifeguard is phubbing.),” 東方日報 Oriental Daily News, 4 May 2010. [Online]. Available: http://orientaldaily.on.cc/cnt/news/20100504/00176_035.html. [Accessed 16 Sep 2021].

[3] “探射燈：救生員缺人康文署密冚 盲點睇漏遇溺難救(Spotlight: LCSD is troubled by the lack of lifeguard. It is difficult to rescue the victim in blind spots.),” 東方日報 Oriental Daily News, 10 Jun 2019. [Online]. Available: https://orientaldaily.on.cc/cnt/news/20190619/mobile/odn-20190619-0619_00176_091.html. [Accessed 16 Sep 2021].

[4] D. Szpilman and J. J. Bierens, “Drowning,” *New England Journal of Medicine,* vol. 366, no. 22, pp. 2102-2110, 2012.

[5] K. Ho, “Hong Kong's lifeguard shortage could seriously impact your summer plans,” SCMP, 16 Apr 2019. [Online]. Available: https://www.scmp.com/yp/discover/news/hong-kong/article/3062096/hong-kongs-lifeguard-shortage-could-seriously-impact. [Accessed 14 sep 2021].

[6] “No lifeguard at half of all public beaches,” The Standard, 25 Aug 2021. [Online]. Available: https://www.thestandard.com.hk/breaking-news-print/179370/No-lifeguard-at-half-of-all-public-beaches. [Accessed 14 9 2021].

[7] “HONG KONG DROWNING REPORT,” the Non-Communicable Disease Division, HKSAR, 2019.

[8] “Water Resistance On Watches Guide,” The Watch& Clock Shop, [Online]. Available: https://www.watcho.co.uk/watches/information/water-resistance-on-watches-guide.html. [Accessed 14 Sep 2021].

[9] M. R. MS, M. Ali, P. E. N, N. E. G, S. Ali and K. M.Y, “An early drowning detection system for internet of things (Iot) applications,” *TELKOMNIKA (Telecommunication Computing Electronics and Control),* vol. 16, no. 4, p. 1870, 2018.

[10] “What Does a Real Drowning Victim Look Like?,” FOX NASHVILLE, 29 7 2014. [Online]. Available: https://youtu.be/ErtQKtTTGxs. [Accessed 14 Sep 2021].

[11] L. Noggin, “What Really Happens To Your Body When You Drown?,” 9 Mon 2017. [Online]. Available: https://youtu.be/CMxHQ5B7dZ8. [Accessed 14 Sep 2021].

[12] S. Cheriyedath, “Photoplethysmography (PPG),” News Medical, 27 Feb 2019. [Online]. Available: https://www.news-medical.net/health/Photoplethysmography-(PPG).aspx. [Accessed 14 Sep 2021].

[13] “Tests to measure your oxygen levels,” British Lung Foundation, Jan 2020. [Online]. Available: https://www.blf.org.uk/support-for-you/breathing-tests/tests-measure-oxygen-levels. [Accessed 14 Sep 2021].

[14] “Air Safety Institute Safety Alert,” AOPA, [Online]. Available: https://download.aopa.org/asf/Air-Safety-Institute-Safety-Alert_Hypoxia.pdf. [Accessed 14 Sep 2021].

[15] L. J. Montenij, W. de Vries, L. Schwarte and J. J. Bierens, “Feasibility of pulse oximetry in the initial prehospital management of victims of drowning: A preliminary study,” *Resuscitation,* vol. 82, no. 9, pp. 1235-1238, 2011.

[16] M. Ghamari, “A review on wearable photoplethysmography sensors and their potential future applications in health care,” *International Journal of Biosensors & Bioelectronics,* vol. 4, no. 4, 2018.

[17] “ZOLL R Serires Pulse Oximetry (SpO2),” ZOLL, Jan 2018. [Online]. Available: https://www.zoll.com/-/media/public-site/products/r-series-defibrillators/9650-0901-01-sf_f.ashx. [Accessed 14 Sep 2021].

[18] L. Holbery-Morgan, J. Carew, C. Angel and N. Simpson, “Feasibility of pulse oximetry after water immersion,” *Resuscitation Plus,* vol. 7, p. 100147, 2021.

[19] “Mykee Locker Key Holder,” [Online]. Available: http://www.mykee.com/english/html/emykee_main.html. [Accessed 14 Feb 2022].

[20] “Boys' swimming story lines, metro swimmers and divers to watch,” 11 Dec 2017. [Online]. Available: https://www.mnswimminghub.com/news_article/show/865090. [Accessed 14 Feb 2022].

[21] “MAX30100 Pulse Oximeter and Heart-Rate Sensor IC for Wearable Health,” 24 Sep 2014. [Online]. Available:

https://datasheets.maximintegrated.com/en/ds/MAX30100.pdf. [Accessed 2022 Feb 14].

[22] “Strap band for Xiaomi Mi Band 3/4/5/6,” AliExpress, [Online]. Available: https://tr.aliexpress.com/item/1005002460119063.html?pdp_ext_f=%7B%22ship_from%22:%22%22,%22sku_id%22:%2212000020731413412%22%7D&&scm=1007.37427.249190.0&scm_id=1007.37427.249190.0&scm-url=1007.37427.249190.0&pvid=f3b1da52-b527-4fff-bd4b-885895169986&utpa. [Accessed 13 2 2021].

[23] “Harris corner detection,” OpenCV, [Online]. Available: https://docs.opencv.org/4.x/dc/d0d/tutorial_py_features_harris.html. [Accessed 14 Feb 2022].

[24] “Hough Line Transform,” OpenCV, [Online]. Available: https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html. [Accessed 14 Feb 2022].

[25] “Geometric Image Transformations,” OpenCV, [Online]. Available: https://docs.opencv.org/4.x/da/d54/group__imgproc__transform.html#ga20f62aa3235d869c9956436c870893ae. [Accessed 14 Feb 2022].

[26] “Template Matching,” OpenCV, [Online]. Available: https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html. [Accessed 14 Feb 2022].

[27] K. Watson, “Drowning Facts and Safety Precautions,” Healthline Media, 24 Mar 2020. [Online]. Available: https://www.healthline.com/health/how-long-does-it-take-to-drown#drowning-stages. [Accessed 14 Sep 2021].

[28] A. H. Marblestone and Z. B. M., “Physical principles for scalable neural recording,” *Frontiers in Computational Neuroscience,* 2013.

[29] “How to ensure heart rate is accurate on Samsung Galaxy Watch Active?,” Samsung, 12 Oct 2020. [Online]. Available: https://www.samsung.com/sg/support/mobile-devices/how-to-ensure-heart-rate-is-accurate-on-samsung-galaxy-watch-active/. [Accessed 14 Sep 2021].

[30] F. Dehbashi and N. Ahmed, “SwimTrack: Drowning Detection using RFID,” *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos on - SIGCOMM Posters and Demos '19,* 2019.

[31] O. X. Qian, “Automated underwater lifeguard,” 2019. [Online]. Available: https://ongxiangqian.com/portfolio. [Accessed 14 Sep 2021].

[32] W. T. Fok, “Drowning detection with AI,” The University of Hong Kong - KE Newsletter, 2020. [Online]. Available: https://www.ke.hku.hk/story/drowning-detection-with-ai. [Accessed 14 Sep 2021].

[33] W. T. Fok, “HKU AI Drowning Detection System,” 15 Oct 2020. [Online]. Available: https://youtu.be/nFtpUgEQvvI. [Accessed 14 Sep 2021].

[34] “A drowning detection system for swimming pools,” SwimEye, 15 Nov 2020. [Online]. Available: https://swimeye.com/. [Accessed 14 Sep 2021].

[35] RichD, “Tennis Ball Thrower,” 11 Feb 2019. [Online]. Available: https://grabcad.com/library/tennis-ball-thrower-1. [Accessed 14 Sep 2021].

[36] “Arduino Nano,” Arduino.cc, [Online]. Available: https://store.arduino.cc/products/arduino-nano. [Accessed 14 Sep 2021].

[37] “small li-polymer battery 501015 3.7v 20mah 50mah lithium ion battery,” Alibaba, [Online]. Available: https://www.alibaba.com/product-detail/small-li-polymer-battery-501015-3_62138465279.html. [Accessed 14 Sep 2021].

[38] “434MHz RF Transceiver Module (UART)-100m,” Cytron, [Online]. Available: https://www.cytron.io/c-wireless-devices/c-rf-module/p-434mhz-rf-transceiver-module-uart-100m. [Accessed 14 Sep 2021].

[39] “Arduino MAX30100 | Pulse Oximeter Heart-Rate,” Taiwan Intelligent Sensor Technology Co., Ltd., [Online]. Available: https://www.taiwansensor.com.tw/product/arduino-max30100-%E8%84%88%E6%90%8F%E8%A1%80%E6%B0%A7%E5%84%80-%E5%BF%83%E7%8E%87pulse-oximeter-heart-rate-%E6%84%9F%E6%B8%AC%E5%99%A8%E6%A8%A1%E7%B5%84/. [Accessed 14 Sep 2021].

[40] “NeoPixel LEDs,” Compoents101, 3 Oct 2018. [Online]. Available: https://components101.com/displays/neopixel-led-strip. [Accessed 14 Sep 2021].

[41] “Dev47Apps,” Dev47Apps, [Online]. Available: https://www.dev47apps.com/. [Accessed 14 Sep 2021].

[42] “Double for loop performance | Speed Test | Java vs Javascript vs Python | Tech Tips,” Flabo Education, 17 4 2020. [Online]. Available: https://youtu.be/QrFq8d8ngzQ. [Accessed 14 9 2021].

[43] S. Izle, “Node.js advantages: What do you need to know before starting the project?,” ProCoders, 03 Aug 2021. [Online]. Available: https://procoders.tech/blog/advantages-of-using-node-js/. [Accessed 14 Sep 2021].

[44] H. D. “Best Frontend Frameworks of 2021 for Web Development,” SIMFORM, 5 Jan 2021. [Online]. Available: https://www.simform.com/blog/best-frontend-frameworks/. [Accessed 14 Sep 2021].

[45] “Optimizing Performance,” React, [Online]. Available: https://reactjs.org/docs/optimizing-performance.html. [Accessed 16 Sep 2021].

[46] “ISL will Donate bulkhead to city of Naples to divide pool into 25 Meter Course,” SwimSwam, 19 Sep 2019. [Online]. Available: https://swimswam.com/isl-will-donate-bulkhead-to-city-of-naples-to-divide-pool-into-25-meter-course/. [Accessed 15 Sep 2021].

[47] R. Pelayo, “How to use the Max30100 as Arduino heart rate sensor,” Microcontroller Tutorials, 23 Jul 2021. [Online]. Available: https://www.teachmemicro.com/max30100-arduino-heart-rate-sensor/#google_vignette. [Accessed 16 Sep 2021].

6. Minutes

6.1 Minutes of 9th project Meeting

Date: 10 June 2021 (Thur)

Time: 17:30 – 19:00

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

- Research on SLAM, ROS, Printed PCB.
- Terrain the robot will operate on.
- Ask Mech Prof for advice on the terrain adaptability of the robot.
- Try all peripherals before summer break ends.

Discussion:

1. General stuff to be aware
    a. We have two reports to submit (MECH and CPEG)
    b. Think of how to test our robot
    c. Setup “complete stages” of the project, to ensure to Project is “complete”
    d. UI is a crucial element
    e. The project should Demo Oriented
2. Idea on robot
    a. The Robot should be overridable, as we cannot promise the bot to operate the way we want.
    b. Which Sensor we may use: Thermal Camera, Gyro, GPS, Mic, LTE module
    c. Form of Robot: Portable? 4-feet? Drone? Hybrid (4-feet + drone)?
    d. Control of Robot: autonomous/with a human assist/Human control with an electronic assist(advice)
    e. What to do when it encounters patient: set up a communication channel to outside, Bring patient back to the main route.
3. We set up 4 Temporary “complete stages” for our Project

6.2 Minutes of 2nd Project Meeting

Date: 1 July 2021 (Thur)

Time: 22:00 – 00:30

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

- Research on GPS accuracy
- Ask Robin for a sample of FYP

Discussion:

1. Focus: The patients who cannot call for help, mostly solo and without preparation
2. Idea:
   a. Keep Track of the people, send out when needed
   b. A wearable device, keep track of heart rate, etc
   c. Trigger from outside to obtain patient's GPS location (PLBS?)
   d. A drone that able to fly inside the forest
   e. Signal Approach: a drone that detects the common phone's communication frequency
3. Soft Robot This Unstoppable Robot Could Save Your Life
4. How to Find People who are not in range of signal
   a. 天災時代替基地台的無人機，可供方圓 10 公里手機同時使用 (A Drone that provides Mobile Phone Signal for phones within the range of 10KM)
   b. 中華電信與雷虎科技合作展出「空中基地台系統」(Chunghwa Telecom Unmanned Air Systems; CHTUAS)

Useful Links:

- MOBILE NETWORK COVERAGE OF HIKING TRAILS IN COUNTRY PARKS
- Use fall detection with Apple Watch
- Hong Kong Frequency Allocation Chart
- WHY DRONES ARE THE FUTURE OF SEARCH AND RESCUE

6.3 Minutes of 3rd Project Meeting

Date: 7 June 2021 (Wed)

Time: 22:00 – 00:30

Present: Dr. Desmond Tsoi, Kris, Loreen, Fionn, Jackal, Don

To-do:

1. Try out different Wheel
2. Try out ROS
3. Feasibility study on Drone
4. Assemble and try belt drive in different terrain /w ROS

Discussion:

1. Idea
    a. A drone saves people from heat stroke
        i. Scenario:
            1. Heatstroke blackspots usually do not have many trees that can block the sunlight, which is suitable for the drone to operate
            2. According to AFCD and hobbies website, there is a place surrounded by 3-4 blackspots within 10km, set up a charging station and keep petrol around those blackspots
        ii. Stage 1: use a thermal sensor to identify patients, take advantage of the drone's propeller to fan the patient, and use a mechanical part to spray water to help heat dissipation.
        iii. Use an AirTag-ish cheap BLE, distribute at the entrance of the hiking trail, if the Tag is not returned within 2-days, we send out the drone to search for the patient
            1. Advice: use the deposit to increase the return rate, add a stamp on the tag, the user can return my putting the tag post-box
    b. Land-Base robot
        i. Scenario: Following the rescue team, Petrol 24/7

ii. Usage:

1. Rescuer uses a phone to order the bot to search for an area
2. Petrol blackspot with schedule
3. If it meets the patient, build a communication channel to the rescuer, block sunlight, spray water, measure body temperature, heart rate

iii. Appearance

1. WALL-E like: https://youtu.be/ISznqY3kESI

c. Limitation in outdoor: signal usually weak as it will not bounce back

2. Kris gives us some detail of using InnoLab
3. Note on Project

a. It is ok to change the project if it is justifiable

i. e.g., What benefits that is better than original

b. The report is a storytelling of the report: the journey of exploration, what option you encounter and why you did not choose them, compare their pros and cons

6.4 Minutes of 4th Project Meeting

Date: 31 June 2021 (Thur)

Time: 17:30 – 19:00

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

1. Ask Mech Prof for approval of the swimming pool robot
2. Research on how to detect drowning, and how to save a drowning person
3. Ready some questions for the swimming pool manager

Discussion:

1. Huge “human-eating” firefighting bot
    a. Hi-power Motor may be expensive, take advantage of gear ration
    b. Fire-proof material, we need to ensure the patient save inside the robot
    c. Can it substitute a firefighter? Maybe it can bring the patient out of the area, to free the firefighter to keep saving people.
    d. Scale is huge and difficulty is high, we might not have enough time, 1:10 scale demo?
2. Swimming Pool Robot:
    a. Privacy problem (If we use computer vision approach), CCTV does exist in public swimming, but it is confidential, we may use a low-resolution camera/Infrared, but does infrared work with water?
    b. Wearable Device approach, measure position, blood oxygen, and heart rate to detect drowning
    c. Underwater robot approach, it is easier to reach the patient. Corrosion problem? Does not necessary to solve
    d. Key Points: Is it fast enough? (30m to reach) Can it save a person?
3. Surveillance heat stroke
    a. We can have multi bot petrol together
4. If we encounter a situation where we do not have enough budget, we need to justify the user scenario

6.5 Minutes of 5th Project Meeting

Date: 5 Aug 2021 (Thur)

Time: 17:30 – 19:00

Present: Prof. Ma, Loreen, Fionn, Jackal, Don

To-do:

1. Gantt Chart
2. CS: Wearable Device, detect drawn, Locating
3. MECH: Release Mechanism
4. Ask Robin to get an FYP sample

Discussion:

1. Shooting approach is preferred than Spidercam approach, but we need to consider the safety issue as it might hit other swimmers.
2. Wrist band is feasible solution.
3. We need to consider the effect of water and movement toward the heart rate sensor.

6.6 Minutes of 6th Project Meeting

Date: 19 Aug 2021 (Thur)

Time: 17:30 – 19:00

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

1. Research and settle a detection method.
2. Draft report before/on 1/9, prove read till last mins.
3. Buy electronic parts and test.
4. Explore more papers about drowning detection, compare their feasibility, cost, and accuracy.
5. Ask Prof. Tim Woo for the question about the control system.

Discussion:

I. Resolution with Robin: 1. wrist band is recommended 2. A NO for spider cam approach, use shooting approach.

II. Backup System: We may give the decision part to the lifeguard to avoid error, Make it configurable for shooting

III. Calibration System: camera angle? Wind direction? we need a system for calibration to adapt to different swimming pools.

IV. IoT communication method: visible Light, Radio wave, we should test one of it first.

V. Desmond gives the detail of writing the FYP report and Presentation

6.7 Minutes of 7th Project Meeting

Date: 25 Aug 2021 (Thur)

Time: 12:00 – 15:00

Present: Loreen, Fionn, Jackal, Don

To-do:

I. Buy heartbeat and Blood oxygen sensor.

II. Try the exiting image approach on GitHub (Drown detection).

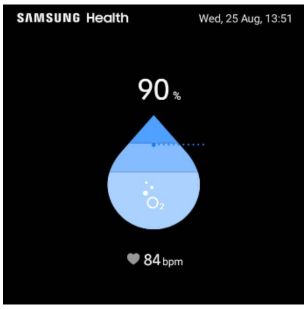
III. Begin in writing the proposal.

IV. Think of how our system communicates with the robot.

V. Ask for advice from Desmond to decide should we buy a commercial watch as a control data.

Discussion:

1. Tennis ball shooting approach and Air cannon shooting approach.
2. We can measure the drop of blood oxygen when we hold our breath, so our approach is promising.
3. We may make a complete seal to avoid water's effect on the blood Oxygen sensors. And use a watch clasp mechanism to ensure it fits.
4. We draft the flowchart and Gantt chart of our project.



6.8 Minutes of 8th Project Meeting

Date: 26 Aug 2021 (Thur)

Time: 11:00 – 01:00

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

1. Read the Indoor Light positioning paper.
2. Buy max30100.
3. Read FYP sample “Vision and Graphic”
4. Ask Robin for the progress report and if Mech need a proposal

Discussion:

1. Desmond gives us detailed advice on writing the proposal. (From structure, to topic, so, literially, a lot)
2. Calibration System: Make a noddle that is equipped with our positioning system that shape and weight similar to our float, so it can measure position, making the calibration process easier.

6.9 Minutes of 9th Project Meeting

Date: 2 Sep 2021 (Thur)

Time: 22:30 – 00:00

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

1. Read the FYP sample “Vision and Graphic”
2. Ask if Prof. Robin have any Mech progress report sample, and if mech need a proposal report

Discussion:

1. The detail, requirement and advice in writing a proposal

6.10 Minutes of 10th Project Meeting

Date: 4 Oct 2021 (Tue)

Time: 20:00 – 22:30

Present: Dr. Desmond Tsoi, Loreen, Fionn, Jackal, Don

To-do:

1. Email to communication tutor to arrange a meeting for individual essay
2. Read sample of individual essay
3. Submit the 1st intermediate report
4. Inform mech FYP group of about the new FYP member

Discussion:

1. Update progress of both CPEG and MECH
    i. Request VM server from FYP
    ii. Tried setup the server using raspberry pi
        1. Advice: separate workload of server (run on client)
2. Distributed the task to do on the project.
    i. CPEG
        1. Take camera data, communicate with robot, UI, config interface (Don)
        2. Drowing detection (Loreen)
        3. Location system, image processing (TBC)
    ii. MECH
        1. Air compressing machine, Tilting
3. Arranged the meeting of mech and cpeg.

6.11 Minutes of 11th Project Meeting

Date: 7 Oct 2021 (Wed)

Time: 20:00 – 22:00

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

1. Don: web app, design the appreance of the website
2. Edwood: try template matching, homography transformation and radio distortion.
3. Loreen: Testing on MAX30100
4. Mech: ongoing test on valve

Discussion:

i. Update progress

1. Borrowed VM server
2. Corner Detection
3. Test the Arduino, LED and buzzer
4. Brough rubber band and float

6.12 Minutes of 12th Project Meeting

Date: 27 Oct 2021 (Wed)

Time: 21:30 – 23:00

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

1. Setup VM server, and make a basic control interface for manipulating the robot
2. waterproof & program (data storage) [buy 2 cheap buzzers]
3. corner detection [harris corner detection]

Discussion:

1. Report Progress
   a. Web interface design by figma
   b. Planning of waterproof setting
   c. Leaning OpenCV

6.13 Minutes of 13th Project Meeting

Date: 3 Nov 2021 (Fir)

Time: 20:30 – 22:30

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

Update progress:

1. Loreen:
   1. trying to verify data with a real SpO2 meter
   2. export MAX30100 data record in spreadsheet for analysis
2. Don:
   1. made a simple demo for communication with robot by web app
3. Edwood:
   1. trail on corner detection (Harris corner detection) w/ blur, RGB channel etc

6.14 Minutes of 14th Project Meeting

Date: 17 Nov 2021 (Wed)

Time: 20:30 – 21:30

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

- Waterproof design
- Purchase a waterproof speaker
- Filter out blue screen and research on edge approach
- MECH: find new parts from the trail

Update progres:

1. Bought a commercial SpO2 tseting for control testing
2. MECH: 1st trail on 3D printing, and find it's limitation

6.15 Minutes of 15th Project Meeting

Date: 1 Dec 2021 (Wed)

Time: 20:00 – 22:00

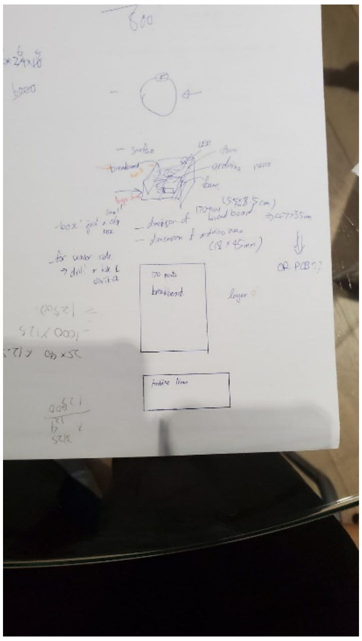
Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

1. UI, stream the video input to client, origin setting (config) of the robot
2. Start assemble the waterproof box
3. Buy the water-proof speaker, arduino nano and 170point breadboard
4. Extract the edge and corner

Update progress:

1. Projectile motion algorithm
2. Made the whole working mechanic from input(touching the screen) to sending command to robot
3. Waterproof planning
4. More study on corner/edge detection solution
5. MECH: ref to 18 meeting



Discussion:

We will need more time, and will avaliable after 10/12 (due to exam)

6.16 Minutes of 16th Project Meeting

Date: 21 Jan 2022 (Fir)

Time: 14:00 □ 16:00

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

1. Test led refraction (Loreen)
2. build small sample (Loreen)
3. test different components (Loreen)
4. Create lifebuoy animation (Don)
5. Fix zoom in/out bug (Don)
6. Check latency (Edwood)
7. Zoom template (Edwood)
8. Test if object does not appear (Edwood)

Update progress:

1. Done 4 corner setting page
2. Done size and robot setting page
3. Done alert time setting page
4. Calculated refraction of the led light -> don’t have to worry total internal refraction
5. Done multiscale template matching

Discussion:

issue about LED detection:

1) Refraction index too big -> difficult to detect the LED

Suggestion:

2) Underwater camera?

3) Change to when hear “help”, shoot buoy -> no because can’t locate that person

6.17 Minutes of 17th Project Meeting

Date: 10 Feb 2022 (Thur)

Time: 20:00 – 23:00

Present: Dr. Desmond Tsoi, Don, Edwood, Fionn, Jackal, Loreen

To-do:

- Change the old stuff (fix figure problem)
- Read fyp report (visually impair and guita)
- add stick man (use case diagram) , may ref zinc's final report page 20. usually start with a verb
- Finish draft2 on 12

Discussion:

We discuss the detail about progress report.

Advice:
1. Make a progress table, may put at the end (after explaination)
2. Consistent terminology
3. Address sec reader's comment
-Detailed division of labor.
-Individual sub-system testing is mentioned.
-Figures cannot express the clear purpose.
4. No need literature review, we can put it in appendix to play safe.
5. Add stick man (use case diagram)

Question to ask:
- Do we need the change the old stuff?

change the old stuff (address figures clearness)

- What is user case diag

How do different users interact with the system
it is a method to analyze the user requirement
We have two users: the client and the lifeguard, and we can also mention the mech part
note: Edwood have studied it b4