Content

1. Introduction

a. System Overview

One of the most significant factors affecting the quality of lifeguard work at the beach is the ability to monitor and operate when an emergency safety event happens and respond quickly.

These days the means available to LG for this monitoring purpose are binoculars and his eyes, using these alone, in some cases, can lead to a situation where the LG is unable to distinguish between an emergency that requires actual intervention, or a situation where verbal alert is enough.

In addition to all of the above, investigating the sequence of events is complex or impossible in many cases but it is surely a necessary task in order to optimize and improve future cases and for evidence. That difficulty comes from the problem to relay and depends on the testimony of the LG or eyewitnesses who are not always precise in the details.

DroneGuard is a LG tool. It will allow them to monitor, navigate and debrief(Figure 2.3) events on the beach. The LG will be able to control the system in an easy and intuitive way that does not require prior knowledge, in order to navigate the drone to a specific area and talk directly to specific bathers.

In addition, the DroneGuard system will also store data and videos of the events to enable future investigation when necessary.

b. Purpose

The main purpose of the system is - to provide lifeguards with a tool that enables them to monitor the water area and the beach, detect any problems, zoom in to swimmers that are in danger or potential danger, and communicate with the relevant swimmers **without** any pre-knowledge about flying drones.

The system will allow the lifeguards to investigate the videos of the incident in order to understand what happened and learn how to deal with emergency or potential dangers better.

The system will allow the lifeguards to patrol the beach and communicate with people using a microphone and speaker.

In addition to the main goal we have sub-goals regarding the functionality of the product:

- The system will have an intuitive interface for non-technical users
- The system will enable clear communication with the swimmer.
- Real-time display of the drone's view and the ability to move to a specific location by touching a specific area on the screen by the LG (touch & go).
- In order to perform and implement the drone movement from point A to B, we will calculate the difference between these two points and the transformation we should output to the LG's screen with the minimum deviation possible.

From the lifeguard side -

- The system will allow the lifeguard to control the drone using a GUI interface, without the need to know how to fly a drone.
- The system will provide the lifeguard a way to communicate with bathers in a more efficient and personal manner.
- The system will function as means of recording and monitoring emergency events for the purposes of interrogation.

From the bathers side -

- The system will allow bathers to communicate with lifeguards using the microphone on the drone.

c. Scope

Our system scope will include three main sections, Monitoring, Navigation and Post Mortem.

In the monitoring section, the scope of our system is to be able to monitor an initial predefined area using the drone and the ability to zoom in and out of specific locations within the pre-defined monitoring area.
The Monitoring will be done by flying the drone to a specific point the LG wants to examine by touching that area on the touchscreen.

The second section is Navigation. This section relates to the actions that the drone should perform, such as moving in any direction, descending in order to zoom in a specific area requested by the LG (under a predefined limit), and more. All of that should be fluent in terms of transformations from the real-world coordinates to the pixels on the screen display and back.

The third section relates to Post Mortem, which is the ability to interrogate a recorded event if needed. The system will allow the LG to access all of the drone videos and the ability to comment or delete them.

d. Definitions and Acronyms

- **Drone** - Unmanned aircraft. Drone is a flying robot that can be remotely controlled or fly autonomously using software-controlled flight plans
- **Base (Docking Station)** - This is a predefined location that will be located next to the life guards cabin. This location is used for the drone to take-off and landing.
- **Observation point** - Observation point is a predefined location which will be the default location for observing the entire monitoring area and from there the lifeguard will be able to zoom in and move to specific locations.
- **Flight-Commands** - After the lifeguard presses on a specific point on the screen, the Translation algorithm will calculate how far the drone should fly to each direction (for example: 10 meters east, 30 meters north) and this flight command will be sent to the drone.
- **Monitor -** One of the three main purposes of our project, Monitoring refers to when the drone zooms in and moves closer to bathers for second look and communication purposes.
- **Navigate -** One of the three main purposes of our project. The navigation will be managed by an algorithm that translates the point touched on the screen to real life coordinates by calculating the pixels coordinates of the touched point with the real life coordinates and height received from the drone to a new real life coordinates and then send flight commands to the drone.
- **Debrief Events (Post Mortem)**- One of the three main purposes of our project. (Figure 2.3)
  When a team carefully deconstructs and analyzes a previous event. A thorough event debrief will help to identify what went right, what went wrong, and what could be better next time.

- **Water area** - The water area inside the predefined area.
- **Beach area** - The Beach area inside the predefined area.
- **Predefined area** - This is the lifeguards responsibility area and the drone flight borders.
- **Potential danger** - This term refers to any case where the drone descends and zooms in and gets closer in order to check if there is a potential danger.
- **Emergency** - This term is used for an event where bathers' lives are in danger.
- **Latency -** The amount of time it takes from the moment the lifeguard touched the screen until the drone started moving.
- **Flight Commands** - A set of commands that can be sent to the drone in order to navigate him from point A to B e.g. takeoff, move right/left, rotate, etc.
- **Touch & Go -** Touch & Go is the method we use in order to navigate the drone by simply touching a point on the screen.

- **GUI -** (Acronyms) Graphical User Interface is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicators, instead of text-based user interfaces.
- **API -** (Acronyms) Application Programming Interface is a connection. between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software.
- **RP -** (Acronyms) Raspberry Pi is a small single-board computer.
- **LG -** (Acronyms) LifeGuard.
- **OP -** (Acronyms) Observation Point.

e. Constraints

- Drone - The project depends on the existence of a drone which will provide a better image of the beach area to the lifeguards.
- Web App - We must have a web app which will allow the end-user to control the drone and all of its functionalities for better and faster assistance.
- Open API - We need to assure that the chosen drone will allow us to use its API for our needs.
- The lifeguard cabin requires an internet connection in order to save the data to the cloud DB.
- We should limit the drone to fly in a predefined area so that the connection with it stays stable.

2. System Architecture –

a. Architectural Description and Design: Roles, Activities and Data

DroneGuard system consists of three main entities:

1. **Drone** - attached with RP to which a camera, GPS, and audio devices are connected
2. **Lifeguard** - has a tablet and can control and navigate the drone.
3. **Investigators** - have access to the DB with the drone footage and can debrief when necessary.

The RP transmits a live video stream from the camera (which is connected to it) to the tablet, responsible for:

- Transformation from pixels of the image to real world coordinates and vice versa.
- Sending the flight commands to the drone.
- Receiving and sending the audio.

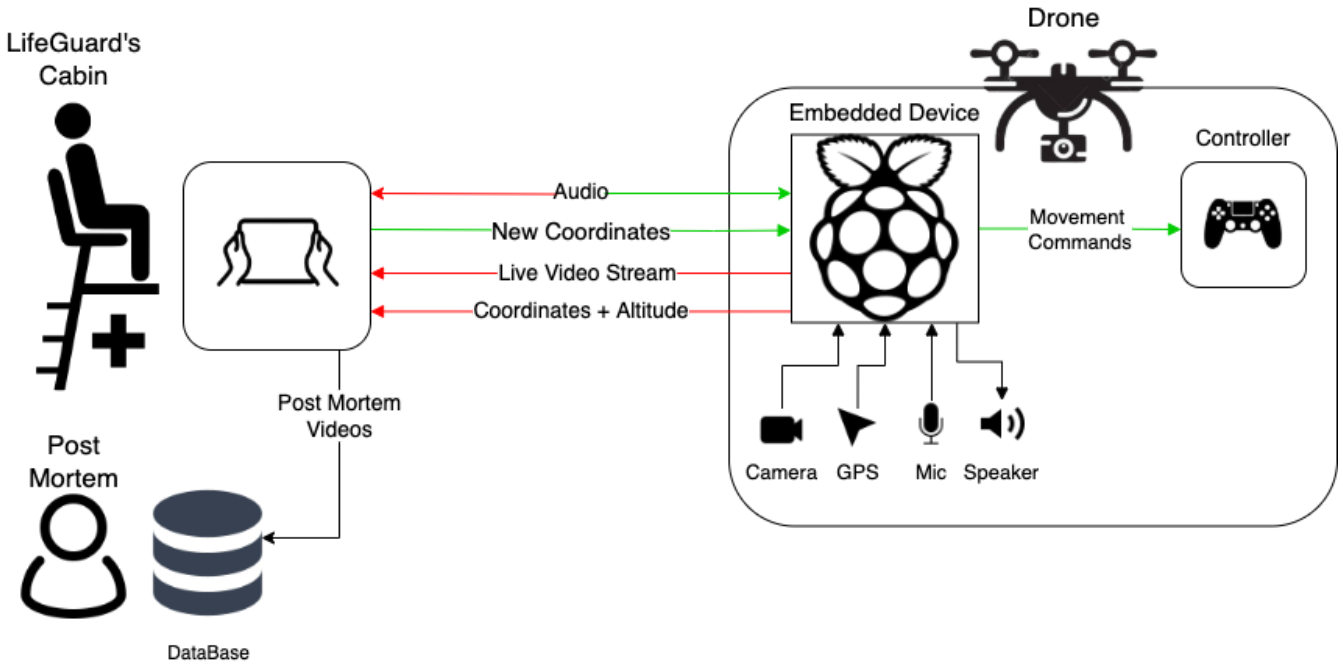


Figure 2.1: System architecture.

b. The Life Cycle of the System

i. Navigation & Monitoring

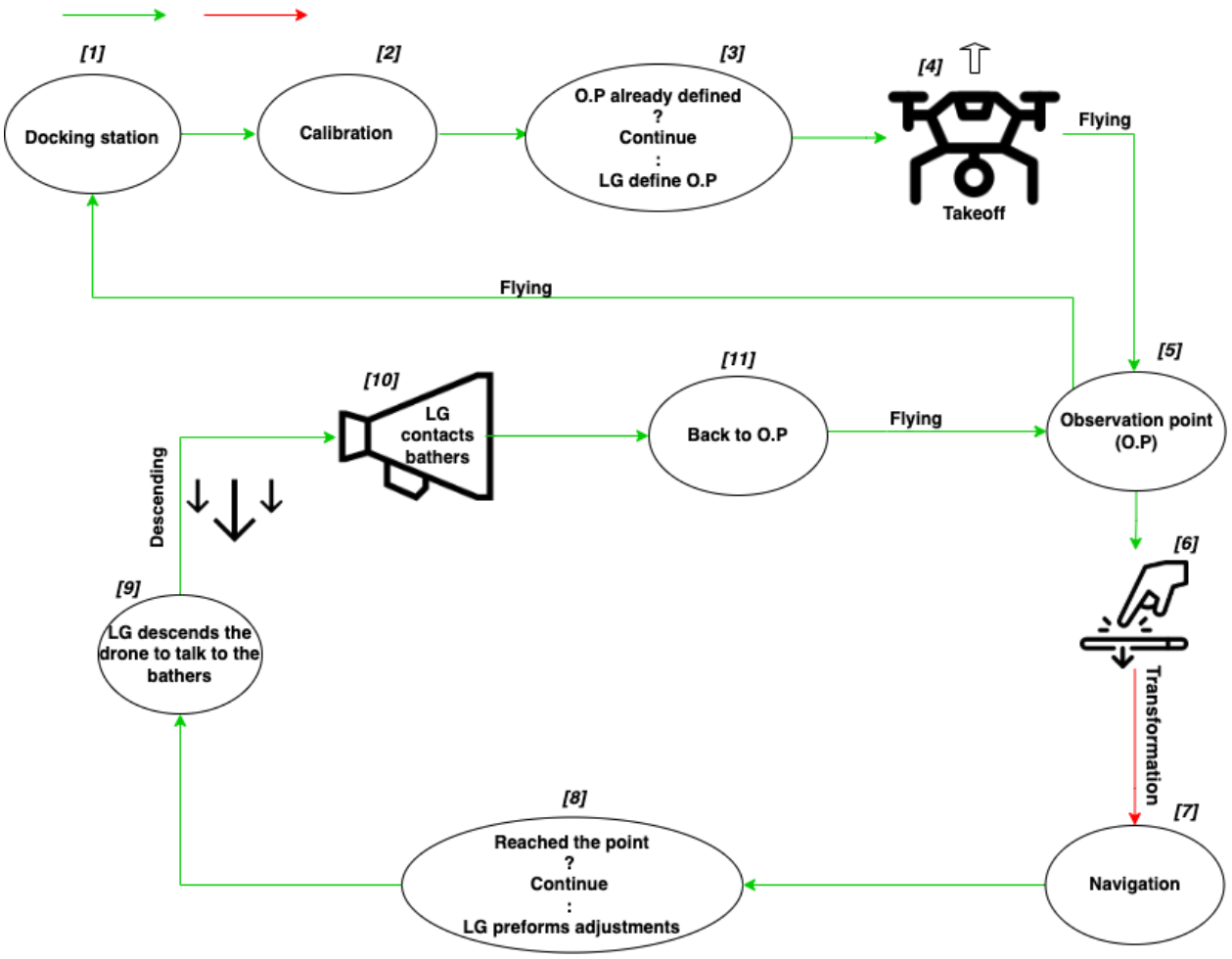


Figure 2.2: Life cycle of the system

Navigation & Monitoring flow steps:

1. Initial drone location.
2. Initial system calibration.
3. If O.P has already defined the LG can take off the drone, else the system will force him to define one using a map.
4. Drone takes off from the docking station by the LG command.
5. Drone arrives at the observation point and stays there.
6. LG watches the beach, touches the point on the screen in order to move there. For this action to take place, we need to calculate the transformation from the real-world coordinates to display pixels and vice versa. (See details regarding transformations calculation at Algorithmic description).
7. The transformations get calculated ⇒ the drone navigates itself to the desired point.
8. LG decides whether the drone is in the correct position or it needs some adjustments.
9. When reaching the point in the LG ability to descend.
10. LG decided to open audio devices to communicate with the swimmer.
11. Drone flies back to the OP, from there the LG decides whether to navigate the drone back to base or to check another point.

ii. Debriefing

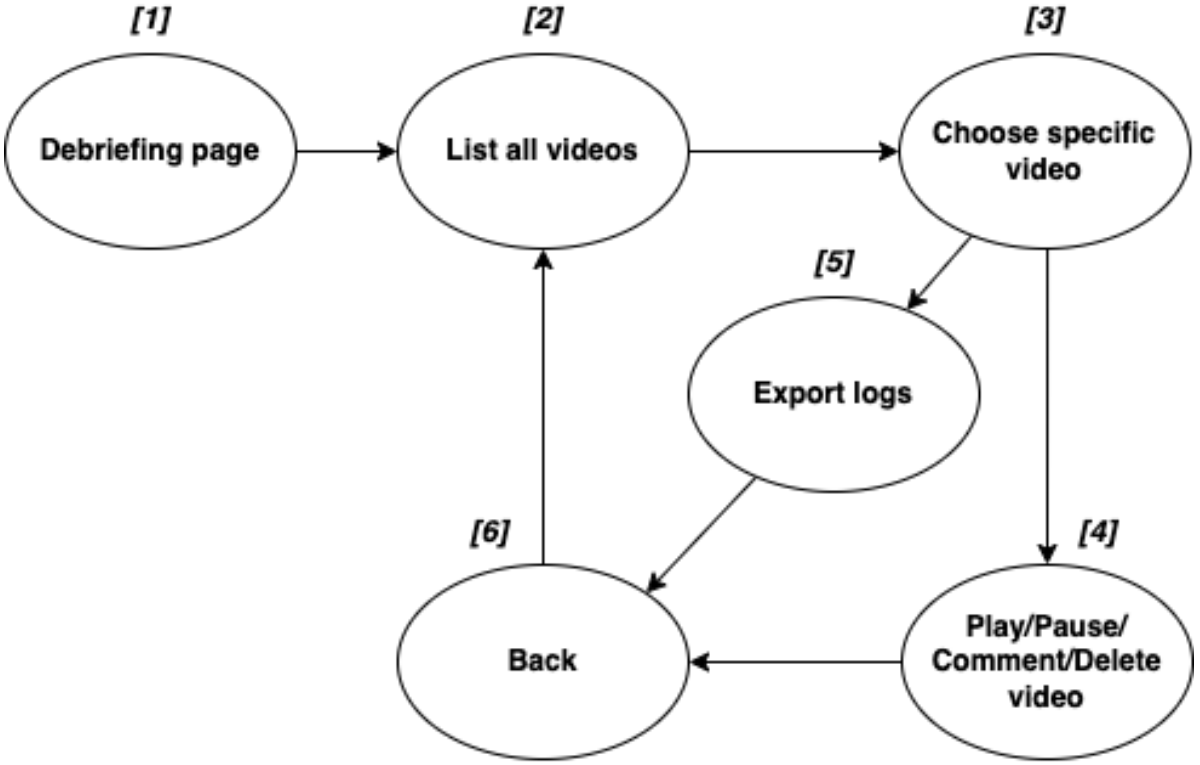


Figure 2.3: Debriefing life cycle

Debriefing flow steps:

1. Debriefing system landing page.
2. Scroll between all recordings.
3. Click specific recording.
4. User options are to play/pause the recording, comment on it and delete.
5. Users can either export drone logs containing data such as drone height, battery level, GPS coordinates, temperature, etc from the specific recording.
6. Go back in order to watch all the recordings.

c. Transformations explained

In order to deal with our main issue, the transformation from pixels to the real world and vice versa have to deal with:

The ability to convert the image obtained from the camera and display it on a screen in a pixel view, so that indicated on the screen can be converted back to….

? so that the relationship between the dimensions of the real world and the view in pixels is maintained.

In this section, we describe the relationships between the real and virtual views and how to convert from one to the other.

Technical concepts:

- **Focal Length** - the distance between the lens and the image sensor when the subject is in focus, usually stated in millimeters.
  determines the optical feature of a camera lens.
  Our camera Focal length is 3.29mm.
- **Field of view (FOV)** - the maximum area of a sample that a camera can image. It is related to two things, the focal length of the lens and the sensor size.
  Objects that are outside the FOV range are not documented in the photo.
  Our camera FOV is 72.4 degrees.
- **Sensor Image Area** - determines how many pixels are captured in the camera lens.
  Our camera's Sensor Image Area is 3.63 X 2.72mm.
- **Drone Orientation -** We will keep the drone orientation facing North. This way will also assist us because when our drone is statically facing north we can always assume that [**North = Forward**, **East = Right**, **South = Backwards**, **West = Left**].
  We get the drone's current direction from a compass component that is connected to the RP on top of the drone so that we can always rotate it northwards by default. While the drone

Navigation Discussion:
In order for the navigation to work as expected when the LG moves the drone right, left forward, or backward, we must set its default orientation northward. To do so, we make sure that after every touch on the screen by the LG the drone sets its orientation north before it executes its next navigation command.
By the time of the reset northward the LG is restricted from touching the tablet until the drone faces north. Therefore we let the LG be concern-free when it comes to compass directions.

The following are the markings of the variables for the purpose of the calculations:

Virtual world parameters(**pixels**) -

- L_S - tablet screen length in pixels (pixels length)
- W_S - tablet screen width in pixels (pixels width)

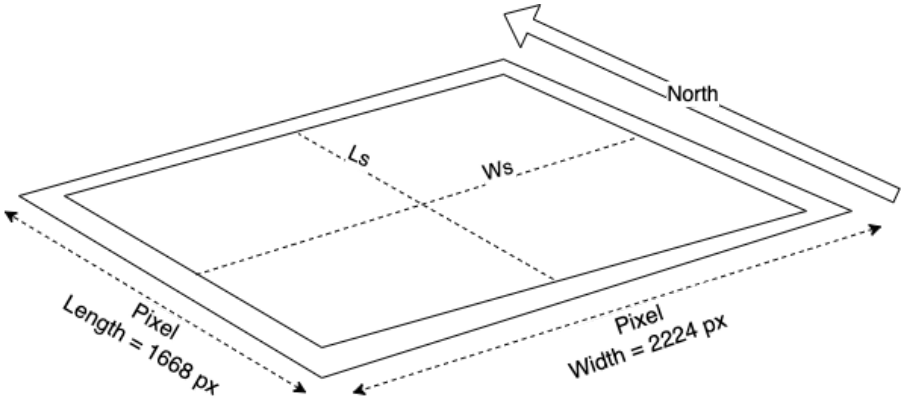


Figure 5.7: virtual world diagram(screen).

Real world parameters(meters) -

- **h** - Drone height (from surface),
  We get drone height from the built-in barometer.
- W_R - Distance (real width)
- L_R - Distance (real length)

Camera's parameters -

- **f** - focal length
- Sen_L - Sensor image area (length)
- Sen_W - Sensor image area (width)
- **α** - Horizontal lens degree
- - Vertical lens degree

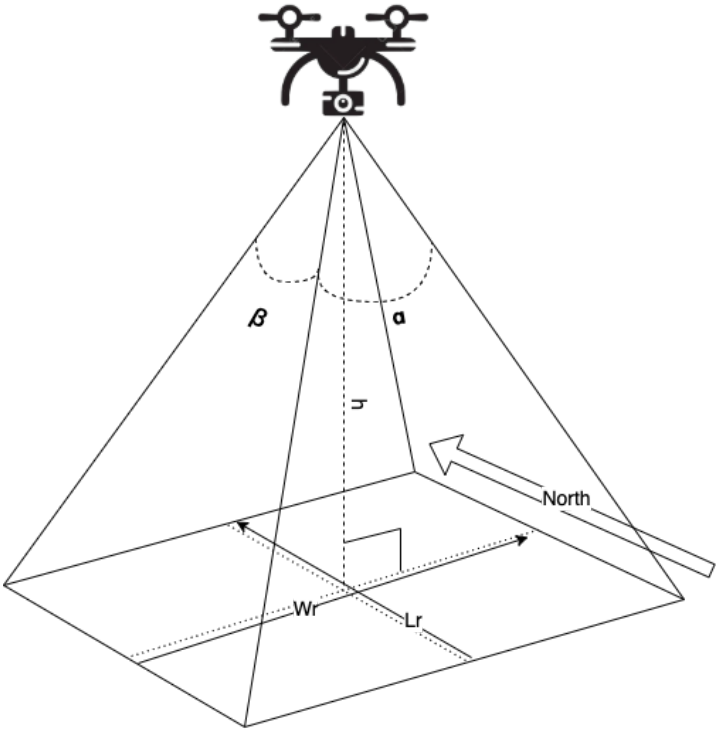


Figure 5.8: real world diagram.

Real to Virtual transformation Calculations :

- First We will calculate our α & β lens degrees from our basic given camera's parameters(f, Senw,Senl).

  Width: $\alpha = 2 \cdot arctan\left(\frac{Senw}{2 \cdot f}\right) \Rightarrow 2 \cdot arctan\left(\frac{3.68}{3.6 \cdot 2}\right) \approx 54.14°$

  Length: $\beta = 2 \cdot arctan\left(\frac{Senl}{2 \cdot f}\right) \Rightarrow 2 \cdot arctan\left(\frac{2.76}{3.6 \cdot 2}\right) \approx 41.94°$

- We use the α & β degrees and the drone height (h) in order to calculate the size of both Width and Length captured **real** dimensions in meters (real world width - W_R & real world length - L_R):

  W_R - Real World Width: $tan\left(\frac{\alpha}{2}\right) = \frac{W_R}{2h} \Rightarrow W_R = 2h \cdot tan\left(\frac{\alpha}{2}\right)$

  L_R - Real World Length: $tan\left(\frac{\beta}{2}\right) = \frac{L_R}{2h} \Rightarrow L_R = 2h \cdot tan\left(\frac{\beta}{2}\right)$

  Example -

  $W_R = 2 \cdot 10 \cdot tan\left(\frac{54.14}{2}\right)$ = 10m~

  $L_R = 2 \cdot 10 \cdot tan\left(\frac{41.94}{2}\right)$ = 7.5m~

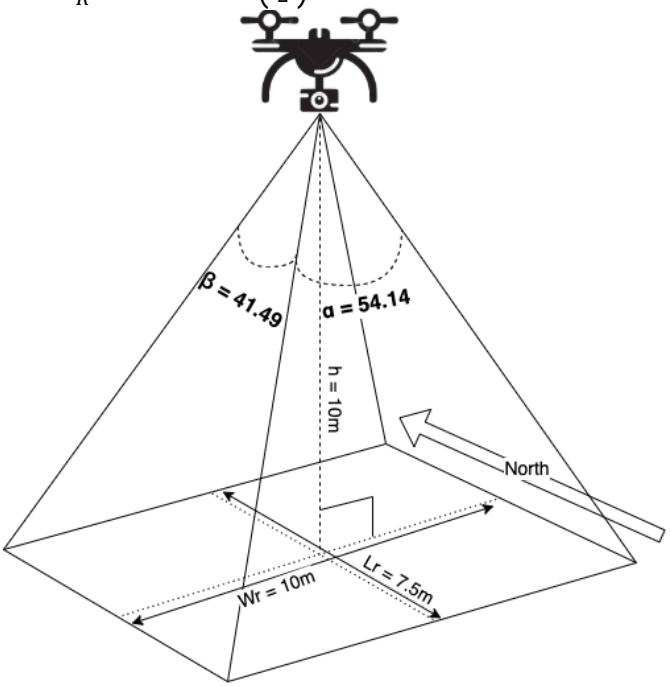


Figure 5.9: real-world diagram(values) .

- Now we will calculate the pixels to meter ratio using W_R, W_S, L_S, L_R from previous calculations as follows:

  con_W - Width Conversion (width ratio of pixels to meter): $\frac{W_R}{W_S}$

  con_L - Length Conversion (height ratio of pixels to meter): $\frac{L_R}{L_S}$

Example -

$$con_W - Width\ Conversion = \frac{10}{2224} = 0.0045$$

$$con_L - Length\ Conversion = \frac{7.5}{1668} = 0.0045$$

Virtual to Real transformation Calculations :

- Now that we know the number of pixels per meter, all that is left is to calculate the distance between the middle point and the selected point(screen touch) by using two variables that represent the real distance from the center of the screen to the destination point in meters.- which are the real distance from the center of the screen to the destination point in meters. To do that we can either use the Pythagorean theorem of subtraction between the points with attention to the four cardinal directions.
  And then we will get the real world distance from the screen distance.

  New_point = The point on the screen where the lifeguard pressed on.

  Middle_point = The middle of the screen. $(\frac{W_s}{2}, \frac{L_s}{2})$

- **Move_x** and **Move_y -** they are the real distance from the center of the screen to the destination point in meters on both axes.

  $$Move\_x = (New\_point - Middle\_point) * con_W$$

  $$Move\_y = (New\_point - Middle\_point) * con_L$$

Example -

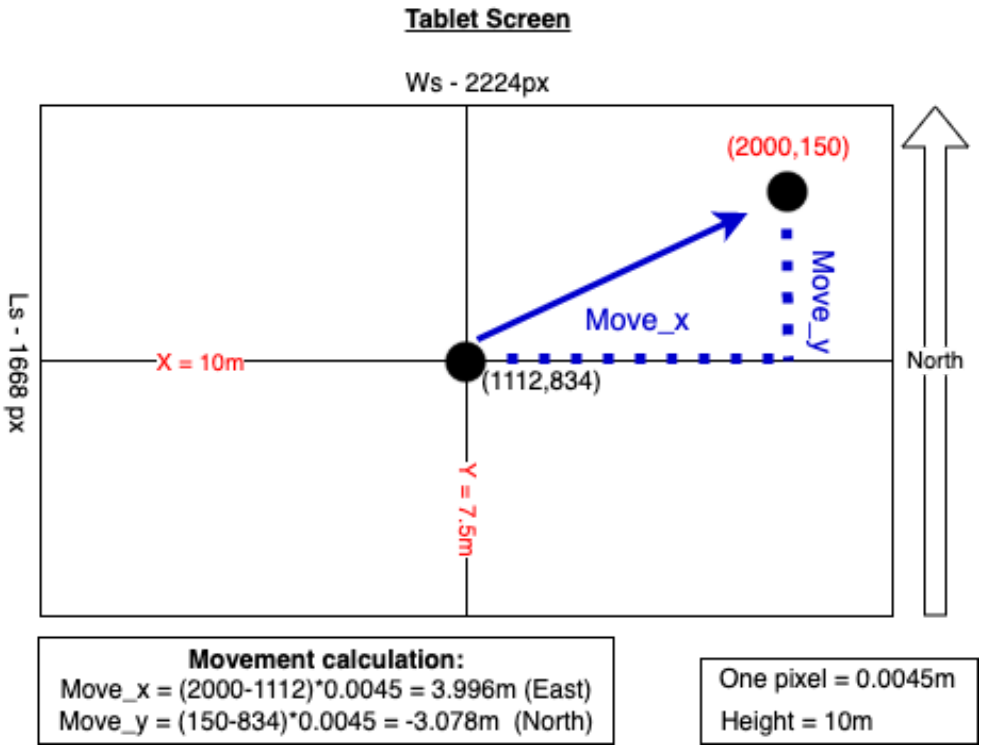


Figure 5.10: Movement calculation example

Illustration - Coordinates Transformation

Step I: LG touches the screen in order to navigate the drone to a specific point.

Step II: coordinates transformation calculation by the server (which runs on the RP), commands are sent to the drone, and the drone reaches the point.

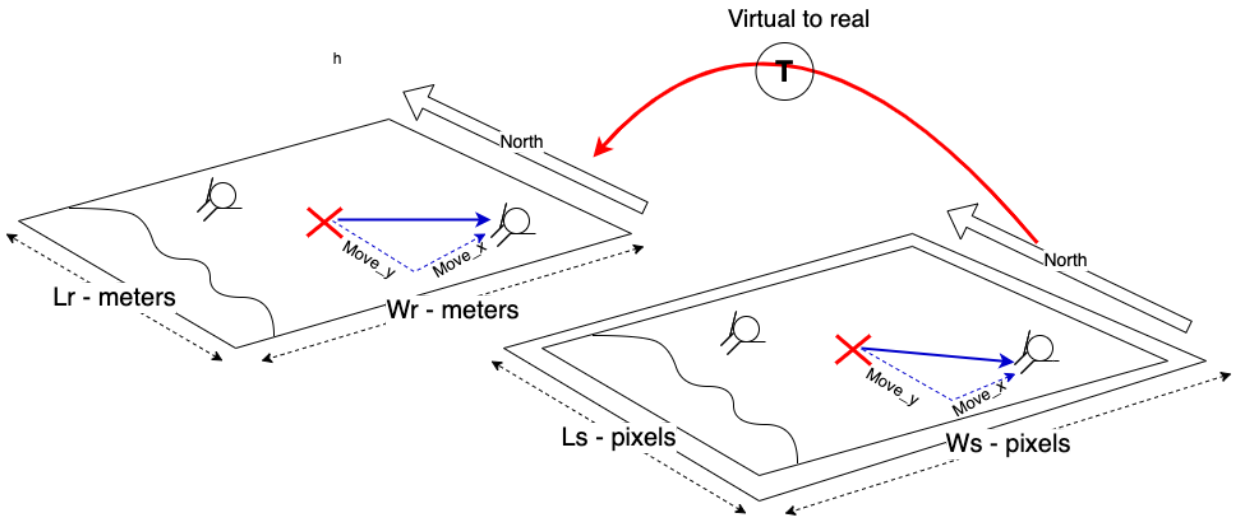


Figure 5.11: Virtual to Real transformation.

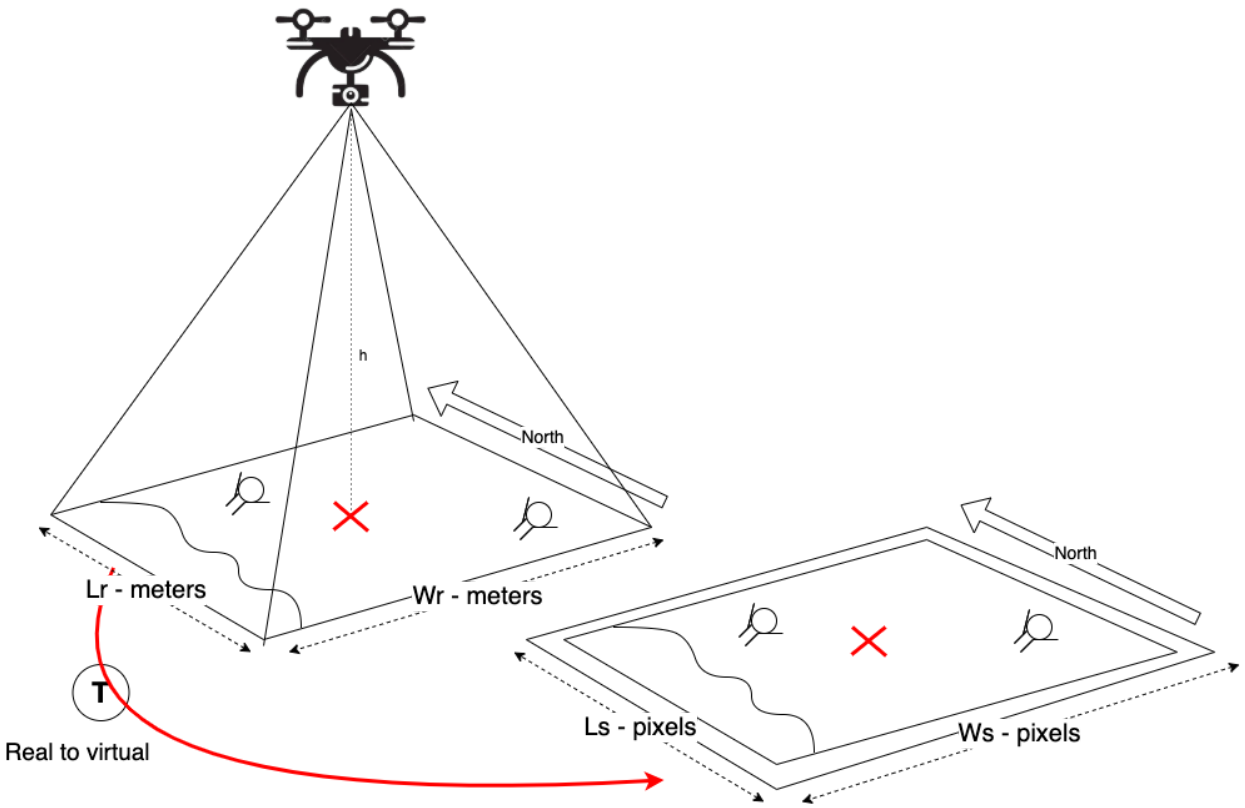


FIgure 5.12: Real to Virtual transformation.

Real to Virtual transformation algorithmic overview:

$$\alpha = 2 \cdot arctan\left(\frac{Sw}{2 \cdot f}\right)$$

$$\beta = 2 \cdot arctan\left(\frac{Sl}{2 \cdot f}\right)$$

$$W_R = 2h \cdot tan\left(\frac{\alpha}{2}\right)$$

$$L_R = 2h \cdot tan\left(\frac{\beta}{2}\right)$$

$$con_W = \frac{W_R}{W_S}$$

$$con_L = \frac{L_R}{L_S}$$

Virtual to Real transformation algorithmic overview:

middle point = $(\frac{W_S}{2}, \frac{L_S}{2})$

$$Move\_x = (New\_point - Middle\_point) * con_W$$

$$Move\_y = (New\_point - Middle\_point) * con_L$$

3. Literature Survey

a. Problem Survey

The lifeguard work is demanding and not simple.
It requires alertness to what is happening on a large surface area and a real-time response with as little procrastination as possible.
We are trying to give lifeguards a better way to monitor, zooming-in (navigating) and debriefing events(Figure 2.3). While doing so we face some challenges.
The first difficulty is the fact that in most cases rescuers do not know how to fly a drone because the operations of flying a drone are fundamentally complicated.
There are almost no built-in solutions for simple and intuitive navigation of a drone without prior experience and knowledge out there.
A drone can be an excellent tool for lifeguards. Allow for faster and more efficient monitoring that will allow improved focus ability for the user.
Today there are a number of projects and companies that can offer solutions to lifeguards as skimmers with cameras and other aids.

b. Solution Survey

Drone as a lifeguard assistant, a way to improve the quality of his work in maintaining order and safety on the beaches, and as a system for investigating safety and emergency incidents.
There is a growing understanding around the world that using drones as a rescue tool in many areas and especially at the beach, has many benefits.

- The Little Ripper - "Lifeguard drone completes world-first ocean rescue"
  An Australian company assembled a drone that can carry various rescue equipment such as life jackets, electric defibrillators, survival kits, and more in order to save lives in beach areas.
- The Pars - robot is an Iranian drone designed to rescue people from drowning. Pars was developed by Amin Rigi and Amir Tahiri at RTS Lab, a lab working on novel robotic innovations for increasing life safety and living conditions. The drone is capable of carrying up to 3 life-rings. An experiment in the Caspian sea, 75 meters from the shore the drone was able to release life-rings under 30 seconds from its launch, compared to the lifeguard who reached a drowning person within 90 seconds.

Drone control, many projects examine the control of drones from different points of view to come up with the most intuitive way to control them. In our project, it is crucial that the drone control be as intuitive as possible so that LG with no pre-knowledge of flying drones can easily handle the GUI and avoid technical obstacles and fly the drone according to his needs.

- Touchscreen Method of Piloting Drones - Intuitive drone touchscreen control by "finger drags"
  Two researchers developed a touch-screen method to navigate and take pictures with drones. The method, called FlyCam, works with the concept of combining the drone and the camera movements. FlyCam uses one- and two-finger drags across a smartphone or tablet to control the drone as it accelerates or turns and takes images. The drone moves forward or backward along the camera's axis with single or double taps to the screen.

- Drone Intuitive Control - describes a way to control a drone through voice and gestures. In this project, the pilot can perform interactive control, which is activated by voice command.

c. Discussion and Conclusion

As seen in the previous sections, the use of drones as monitoring and life-saving tools has been gaining momentum in recent years.
This is due to the recognition of the benefits that drones are equipped with, from the built-in tool and add-ons to the very different, new and unique angles of monitoring.
It allows us to quickly navigate to different and distant points from each other, whether used in an emergency, which can save valuable time.
With all the positive aspects of drone use, the issue of intuition and simplicity in its navigation is still lacking.
This is why our project will focus mainly on managing the simple and user-friendly interface combined with easy navigation capability using the touch screen of the tablet device, which is something that we haven't found a proper solution for in the market.
This is an issue that is still in its infancy and does not seem to be being emphasized at the moment.

4. Technological Survey

a. Problem Survey

Our Problem survey derives to few sub problems. In this section we will describe all of the sub problems:

Drone:

For this project, We had to find a drone which will be able to carry the weight of the embedded system and the rest of the components which will be added to the drone. We also needed to make sure the drone we bought would allow us to connect the embedded system to send flight commands.

Embedded System:

We need to find an Embedded System for our project that will allow us to connect components Physically and also support wireless connections. We will need to use this system to calculate some of the transformation algorithms and also for streaming live video to the LifeGuards tablet.

Communication:

We need to communicate wirelessly between the drone and the tablet(The user interface), and also transfer a large amount of data such as the live video stream from the embedded system and drone data to the tablet. Therefore we need to allow this kind of communication.

Navigation:

We need to be able to navigate the drone by calculating the data given from the drone (Altitude, Latitude, Longitude) and the image from the camera to a new real life coordinate and back to screen coordinates. As part of the navigation problem we need to transform the drone navigation control from the remote control to navigating by touching the screen in order to simplify the navigation task.

b. Solution Survey

Drone:
We have narrowed our options to three different drones which we considered as our best options. We made our decision based on the ability to sync to the drone system, max carry weight, price and max flight distance.

DJI Mavic:
This drone weighs 249g and can carry up to 195g. It has a maximum flight distance of 200 meters and costs 2100 ILS. We didn't choose this drone because after investigation we found out that it's sdk is not supported as much as we thought it would be and therefore might end up causing integration problems.

Assembled Drone:
The assembled drone is a self-built drone, it's using an external API with a dedicated OS. It can carry a weight of up to 500 grams and has a maximum flight time of 20 minutes. It costs 2800 ILS. The reason we did not choose this drone in the end was due to the big risk of a one shot project in case we crash it. The second reason we didn't choose this drone is because of the need to deal with a lot of firmware which we are not familiar with and it might be a big time consumer.

DJI Tello:
This is an educational drone, It weighs 80 grams and can carry up to 45 grams. It's max flight distance is 100 meters and it costs 500 ILS. The reason we chose this drone was mostly the ability to use it's sdk and use it to communicate and navigate the drone much easier than the other drones, also it gives us the ability to use backup drones in case of a crash.

Embedded System:

- Arduino - This controller makes cyclic operations as long as it's connected to a power source. It's computing power is relatively low. The Arduino has no Operating System although it can easily integrate and connect to other components.
- Raspberry pi zero WH - Linux based Computer, It's Operating System will be loaded from an SD card. It can connect wirelessly to Bluetooth and Wifi and also has physical connections for different components. It has high computing power and the ability to stream data.

Communication:

In order to communicate between the App and the Embedded system we will have to consider the fact that we need to use wireless communication since there will be a big distance between the user and the drone when the system will be in use.
We considered three different communication methods:

- Bluetooth - It is a short-range wireless technology standard that is used for data exchange between devices over short distances using Ultra High Frequency radio waves.
- Wifi - A technology that allows data to be transferred between devices connected to it wirelessly via radio waves. The price of the components required to operate it is relatively cheap, and it is quite simple to connect them.

c. Discussion and Conclusion

As discussed in the previous two sections there are a lot of options out there in the market. Most of the reasons for our choice have already been stated in the document. In our selection we have tried to be as close as possible to the needs of the project as we see them from the current point in time.
Of course there is a lot of uncertainty until the equipment is not in our hands and we can start testing the components together, as these are issues we have not dealt with before.
We conducted an in-depth study about which equipment to purchase and what are the most relevant technologies and communication manners that are appropriate for the project. The results of which it seems that our choices will indeed provide the goods of the realization of the project.

5. Design

a. **Data Design** - Database Description

DroneGuard database is straightforward and contains 3 tables which hold all the needed data for the system:

- **Lifeguard:** holds beach id (1:1) and an array of all his recorded videos (1: [optional] many) which can also be empty in case the lifeguard has not recorded any videos.

- **Video:** holds its own url as well as its thumbnail, duration in seconds, date and beach_id in which it was taken.

- **Beach:** holds its name, the city, area dimensions and wifi network host.

b. **Structural Design** - Class Diagram

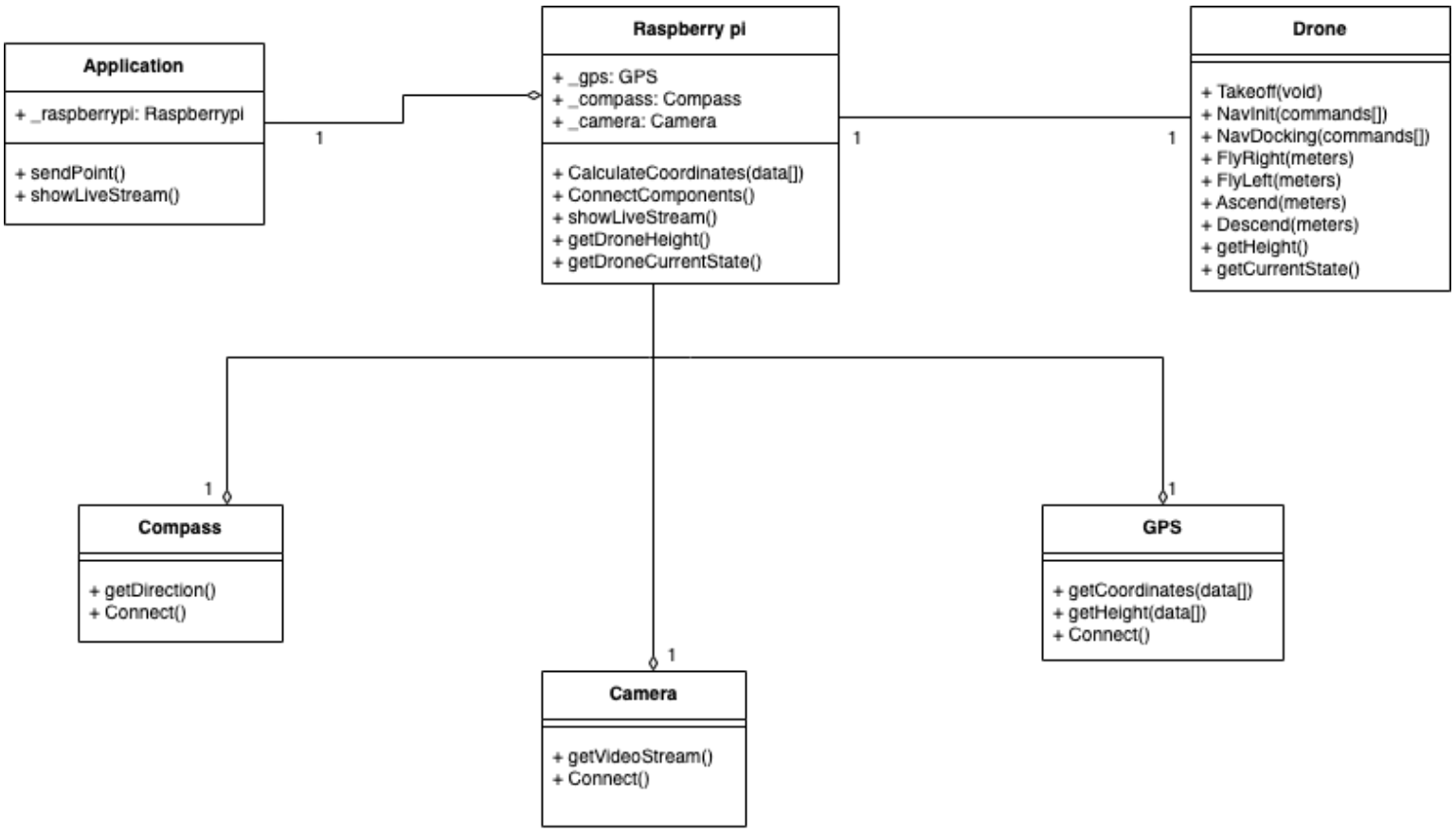


Figure 5.1: Class diagram.

c. Interactions Design

i. Use Cases

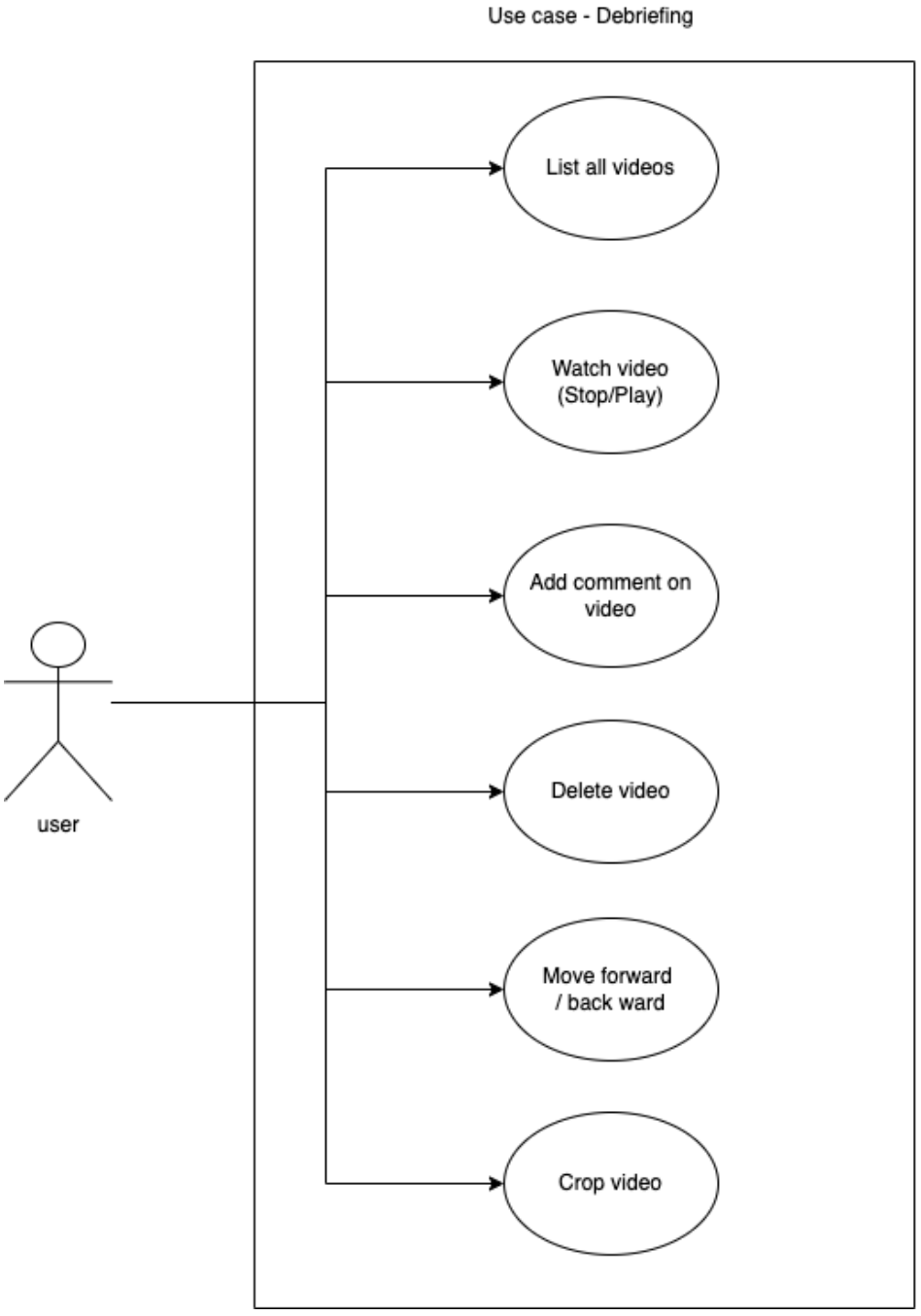


Figure 5.2: Debriefing use case.

The user will have the option to -

- List all videos that were recorded.
- Select a specific video to watch.
- Delete a video.
- Crop a video.
- Move forward and backward.
- Add a comment.

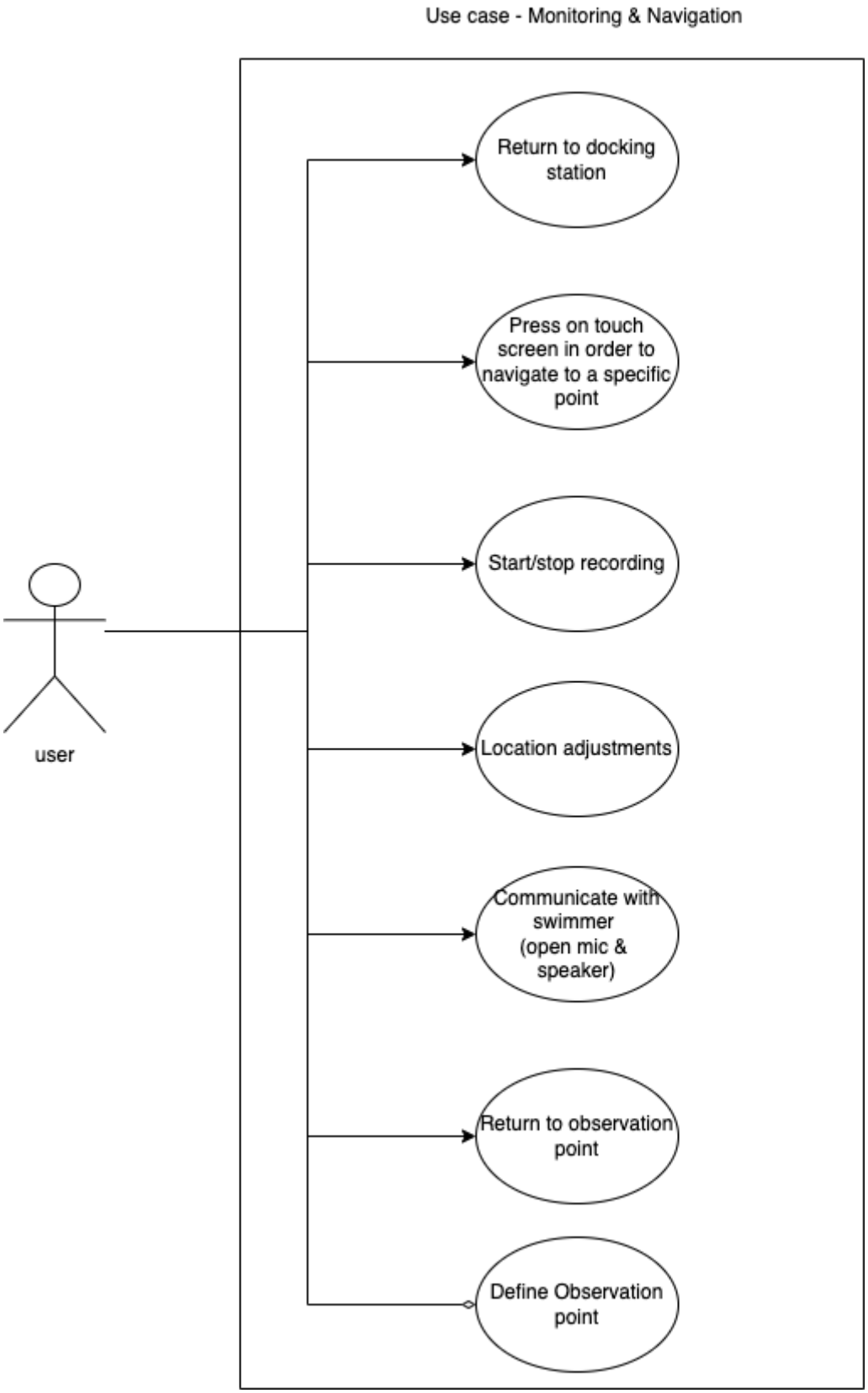


Figure 5.3: Monitoring & Navigation use case

ii. Sequence Diagram

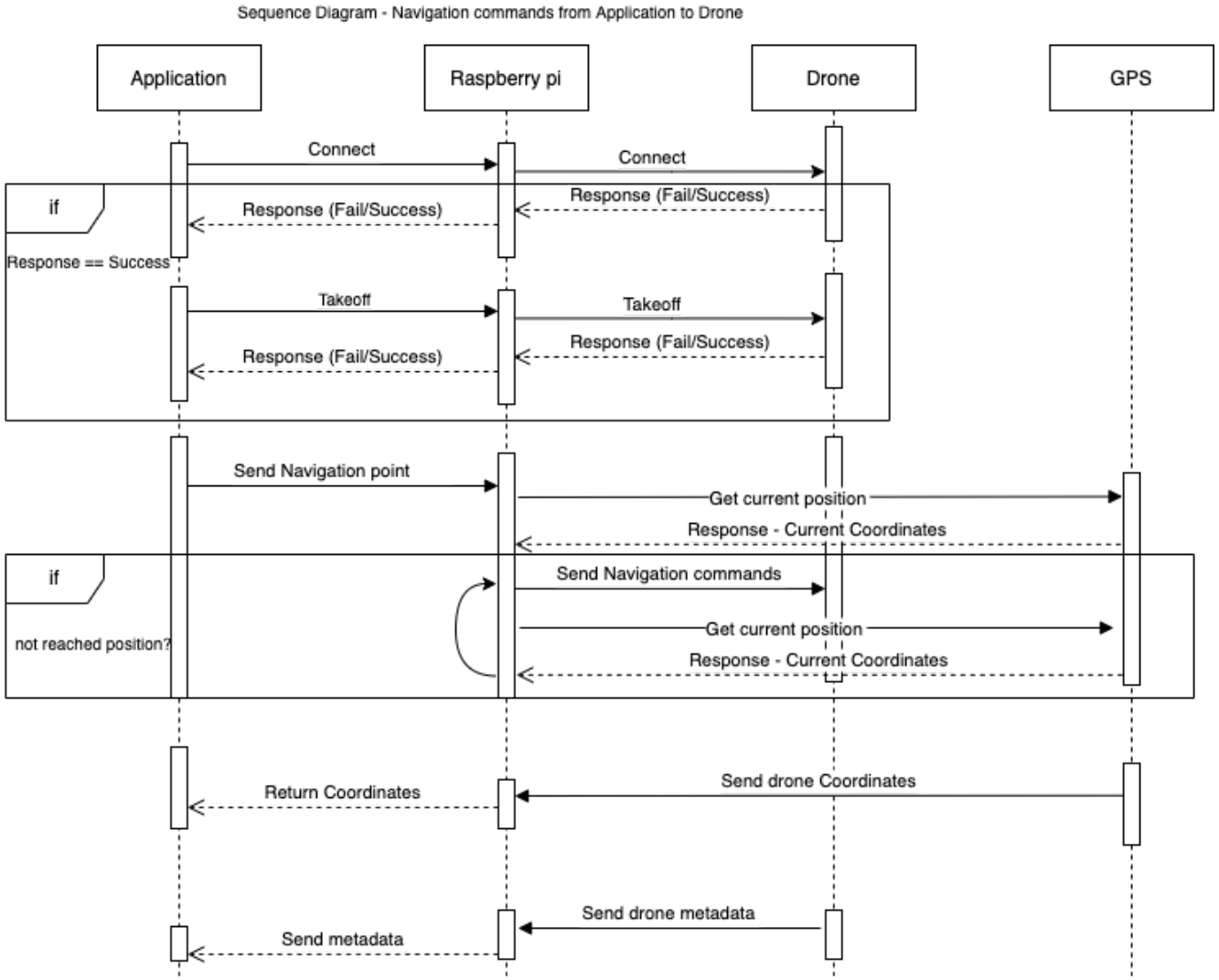


Figure 5.4: Sequence diagram

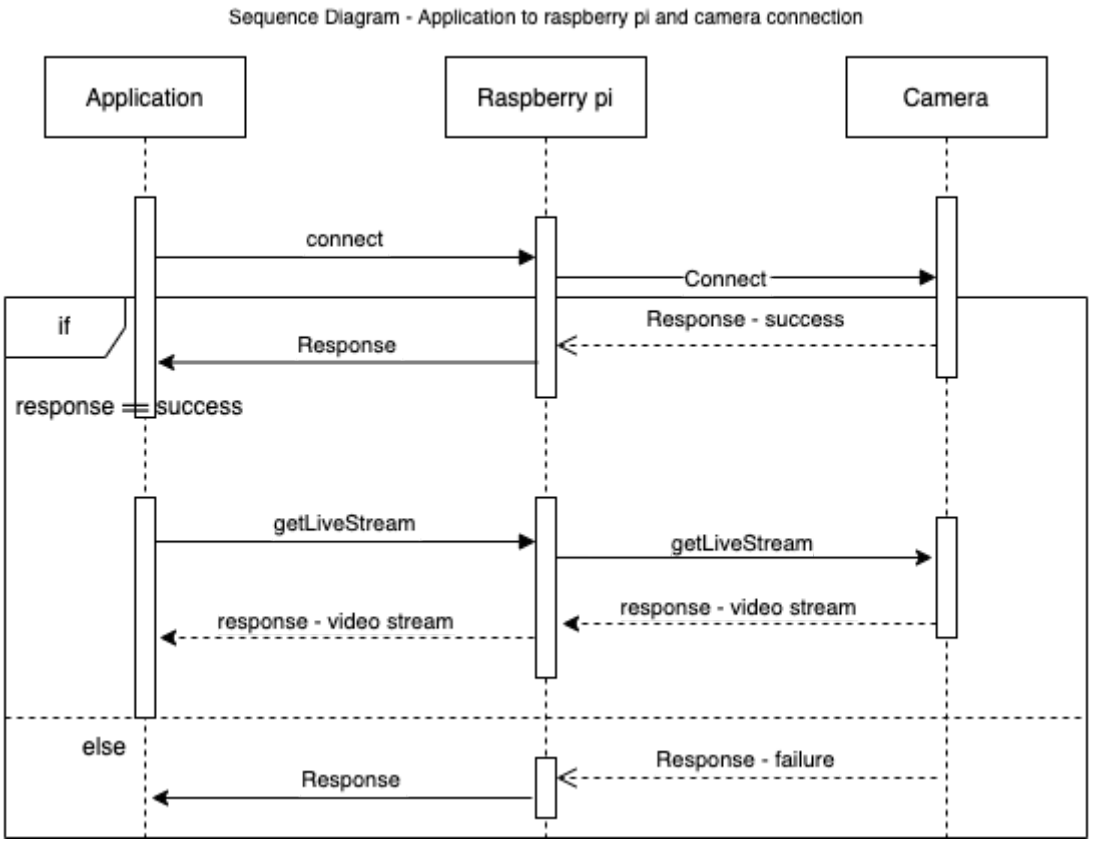


Figure 5.5: Sequence diagram.

iii. Activity Diagram / State / Processes

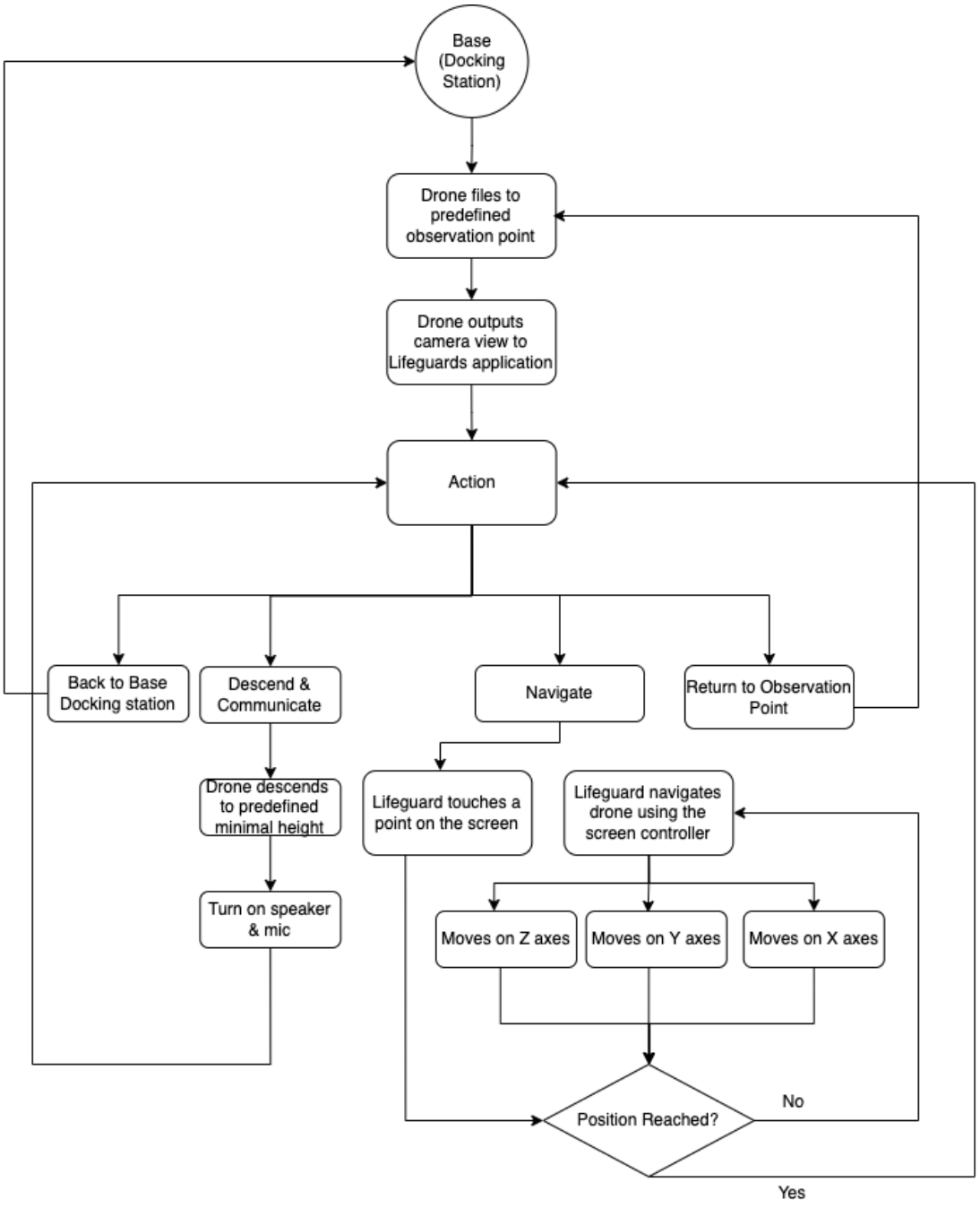


Figure 5.6: Navigation & Monitoring Activity Diagram

d. **Description of Algorithmic Components**

We will have an algorithm that was described and explained in section 2.c.
The algorithm will calculate the transformation between the virtual world coordinations and the real world coordinates.
The Algorithm will be -

1. alpha = calculate_alpha(sw,f)
2. beta = calculate_beta(sl,f)
3. wr = calculate_real_world_width(alpha,h)
4. lr = calculate_real_world_length(beta,h)
5. conW = calculate_virtual_to_real_world_width_converstion(wr,ws)
6. conL = calculate_virtual_to_real_world_length_converstion(lr,ls)
7. middleP = calculate_middle_point(ws,ls)
8. moveX, moveY = calculate_movement_of_drone_in_both_axis(newP, middleP, conW, conL)

By doing this algorithm we will get the movement from the middle point in the screen to the destination point that the LG pressed on the touch screen (**virtual world**) in a **real world** actual navigation commands in meters.

e. Software Architecture Pattern

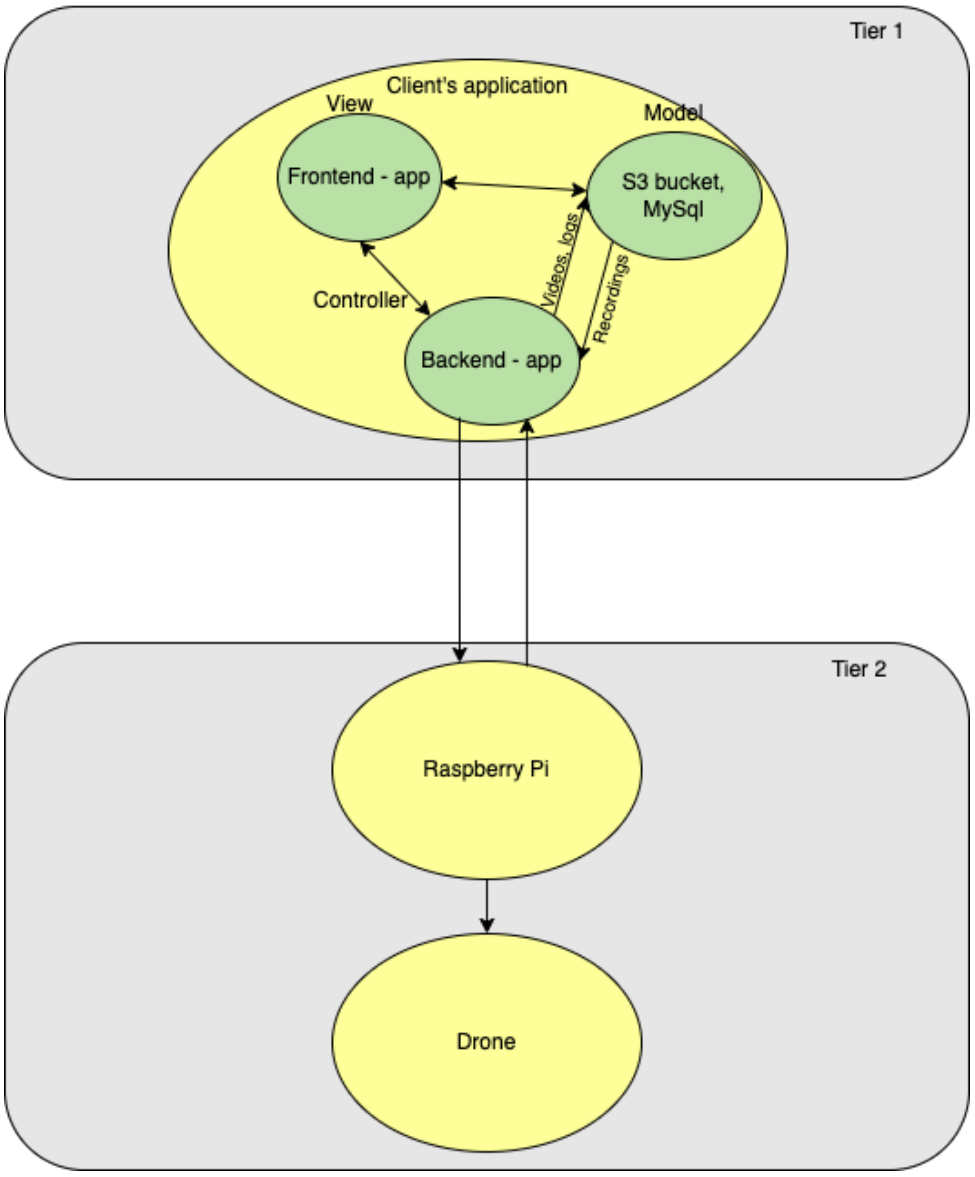


Figure 5.10: System architecture.

6. Risk Management

Hardware:

- A potential risk is that the hardware components that we will choose won't work properly together or won't satisfy our needs for the project. We'll need to investigate them carefully and do a lot of research about the best components for our project.
- The delivery time of the hardware component, due to the worldwide shipping issues and the lack of chips, there is an option that the component will arrive later than expected. We will need to do the ordering as quickly as possible or buy it in Israel(though prices will be much higher..).
- The drone can crash while testing its functionality when it tries to land for reasons such as strong wind, human mistake, or software error. We'll do our best to test the drone carefully.
- Assemble the drone (we will buy the parts separately - reduces costs) which is a knowledge we do not have. We will deal with this with the help of a drone expert, the hardware lab at Shenkar and guides from the Internet.
- Another big risk is the voice communication between the LG and the drone. We will do research if it's possible and if it is, we will check whether we can buy the required equipment with our limited budget.

Software:

- Issues can occur due to a lack of API's documentation or incoherence. In this case, we'll try to have a reference from a working project on GitHub (or any other open source code platform) and figure out its use.
- A complex action to perform is when the lifeguard touches the real-time display and the drone should move to that location (touch & go).

Interpersonal:

- Since all of us have jobs, planning and availability can be difficult to manage and overcome. We'll use a detailed schedule (using a dedicated management system such as Monday) in which we divide the work so that missions won't be missed.

7. Verification

a. Validation and Evaluation Plan

In order to verify our system, we need to take care of and test multiple factors that constitute the primary part of DroneGuard:

- **Hardware**
  - **GPS** - Make sure to receive accurate location data from the GPS component by comparing the data with google maps.
  - **Camera** - Make sure the camera video stream is clear and note if there is any delay in the video transition.
  - **Raspberry Pi** - Stress testing our RaspberryPi is simply running a series of processes that are designed to run the CPU at full power, and monitor the temperature and stability of the system.

- **Algorithm**
  - Calculating and converting the distance of the touched point in the screen in which the LG wants to reach. From pixels to meters to coordinates.
    The calculation should consider the camera angle (in our case 90 degrees vertical), the current height of the drone, with these we convert pixels to meters and to coordinates that can be transferred to the Raspberry Pi. The RP performs calculations in order to convert the given coordinates to commands and send them to the drone.

- **Communication**
  - Measure the maximum **distance** in which the drone is capable of receiving signals.
  - Check for the **latency** of the processing of the command (from touching the screen to executing).
  - **Stability**, calculate the factor of commands that were sent successfully.

b. Testing Platform

Our system contains various programming languages and platforms, each will be tested according to its need and kind:

Node.js:

- **Unit tests:**
  - **Sinon -** for stubs and mocks.
  - **Chai -** for assertion and http requests.

Python:

- **Unit tests:**
  - **Pytest** - testing API's and databases.
  - **Moto** - mocking aws resources.

8. Project Management

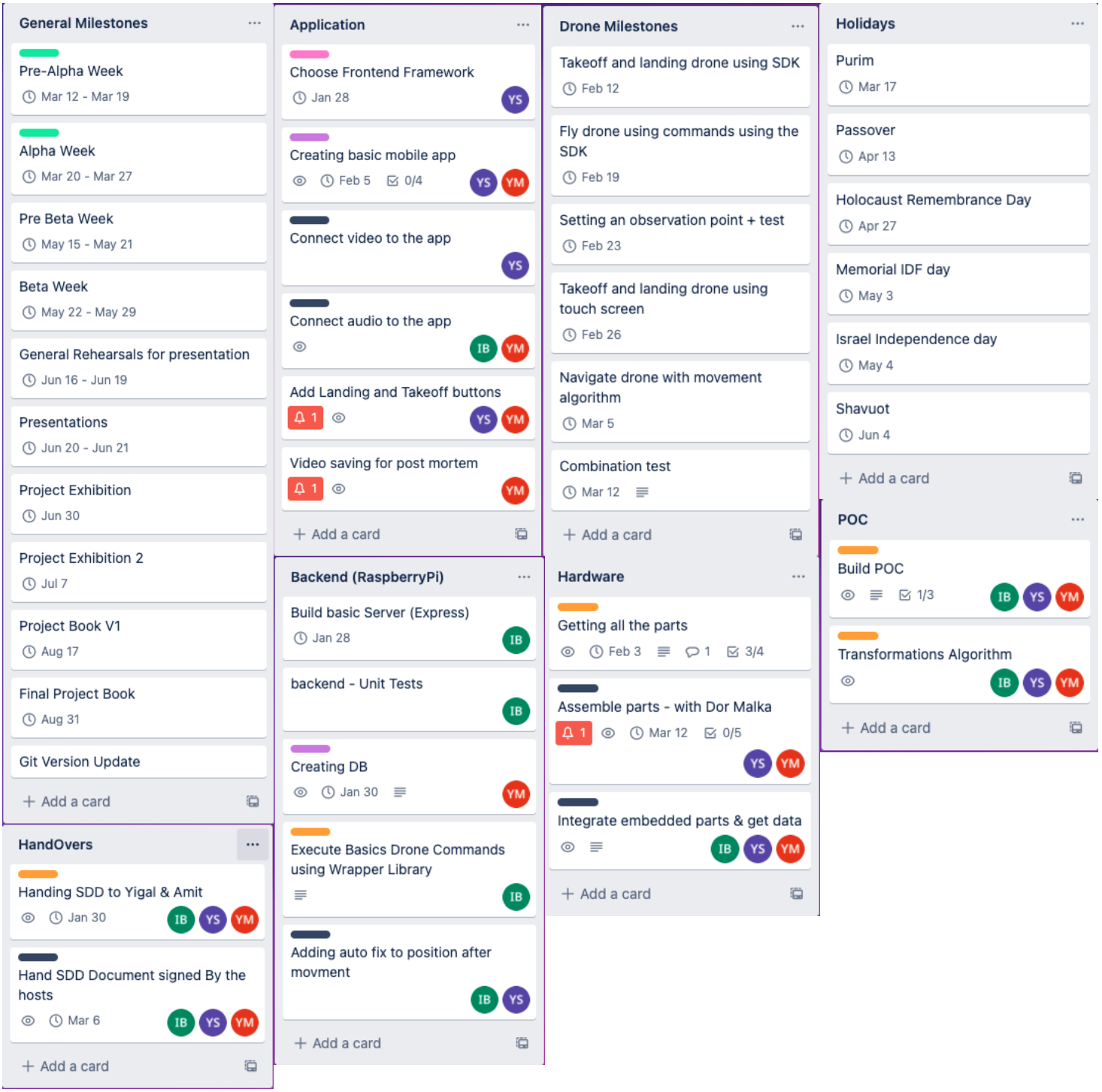
a. Source Control Platform for BU and Sync

- **Git -** We will use Git for version control of our code.
- **GitHub -** We will use GitHub to store and share the code in order to work together and in synced order and avoid conflicts in the code.

b. Alpha/Beta scopes

Subject	Phase	Mission	Yarin	Ido	Yali
Drone	Alpha	• Drone Assembly. • Drone flight tests.			
	Beta	• Full cycle test.			
Algorithm	Alpha	Building Coordinates Translation algorithm.			
Wifi	Alpha	Creating a shared wifi between the drone and application.			
Raspberry pi	Alpha	• Receiving and transferring data from and to the drone. • Receiving data from sensors and components. • Transfer data to the Application. • Connecting all of the components.			
Application	Alpha	• Receiving data from the Raspberry pi. • Sending orders to the Raspberry pi. • Building the application infrastructure.			
	Beta	Navigation • Streaming live video from the drone to the app. • Navigating by touching a spot on the screen. • User Management. • Send navigation commands by touching a dot on the screen. • Set an observation point. • Set the home location on the app. Monitoring • output audio stream via the drone. • Receive an audio stream from the microphone on the drone. Post Mortem • Build database. • Get data from the db to the app. • Save data to DB. • Keep logs from events. • Build application backend.			

c. Schedule / Gantt (possible print screen or sharable link)



d. Team Roles - final

Yarin	Yali	Ido
- assemble the hardware writes a transformation algorithm - write a user interface - Set up DB and link it with the application.	- set server interface with hardware components - writes the transformation algorithm - writes the user interface.	- set up a server and develop it on a microprocessor - write a transformation algorithm - writes user interface.

9. System Functional Screens

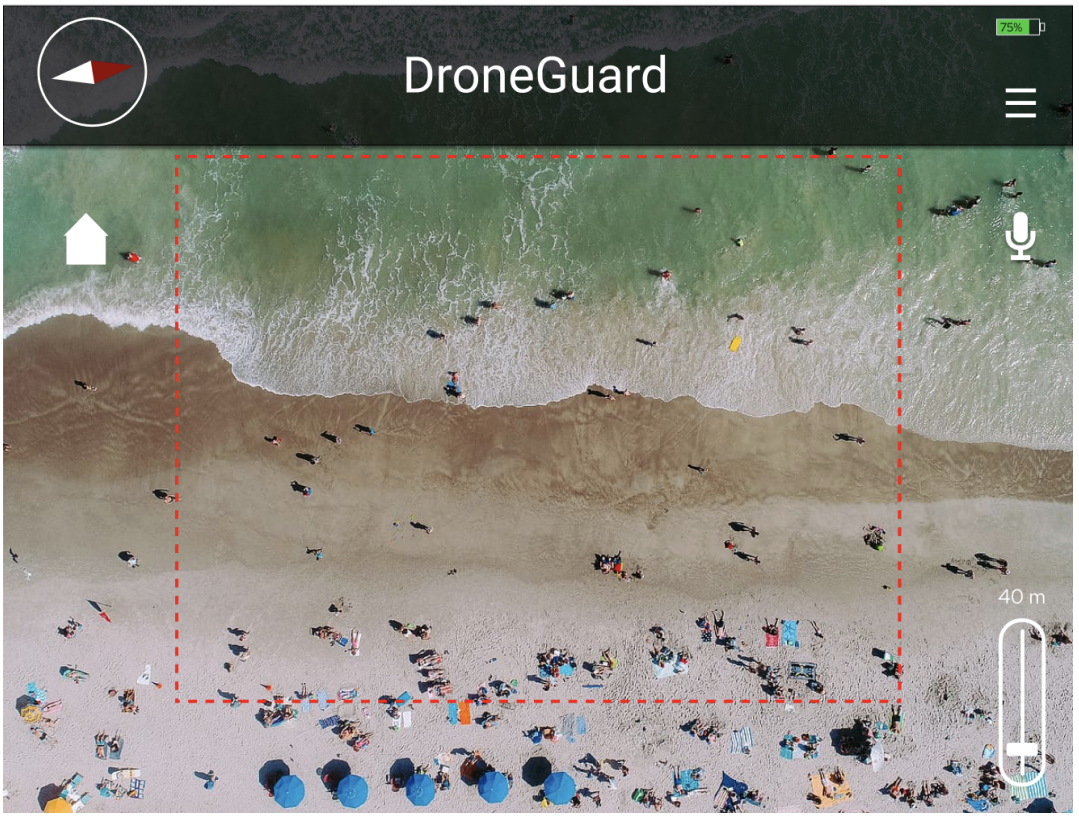


Figure 9.1: DroneGuard located at the observation point

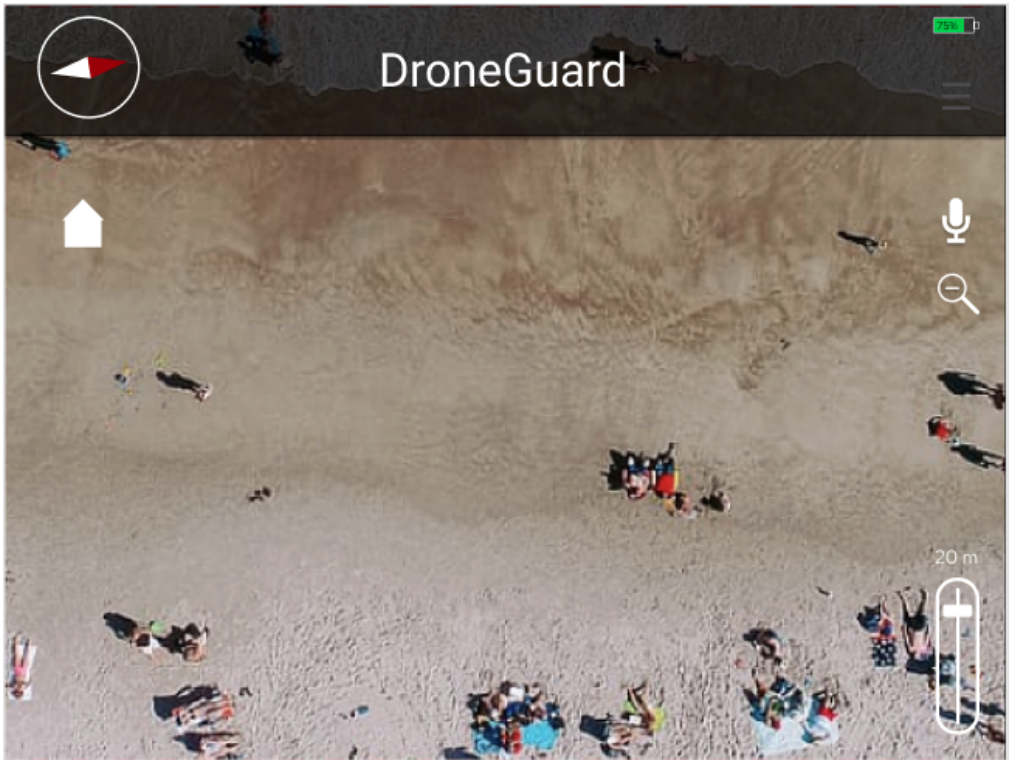


Figure 9.2: DroneGuard located at the chosen location (Zoomed In)

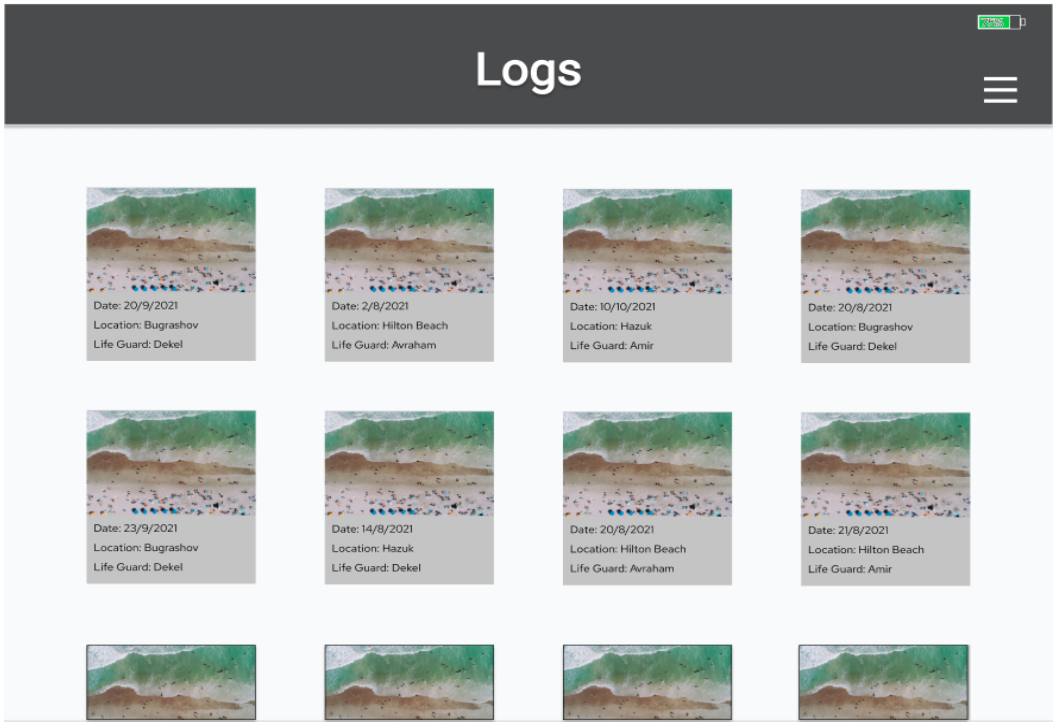


Figure 9.3: Post Mortem page with all of the previous records.

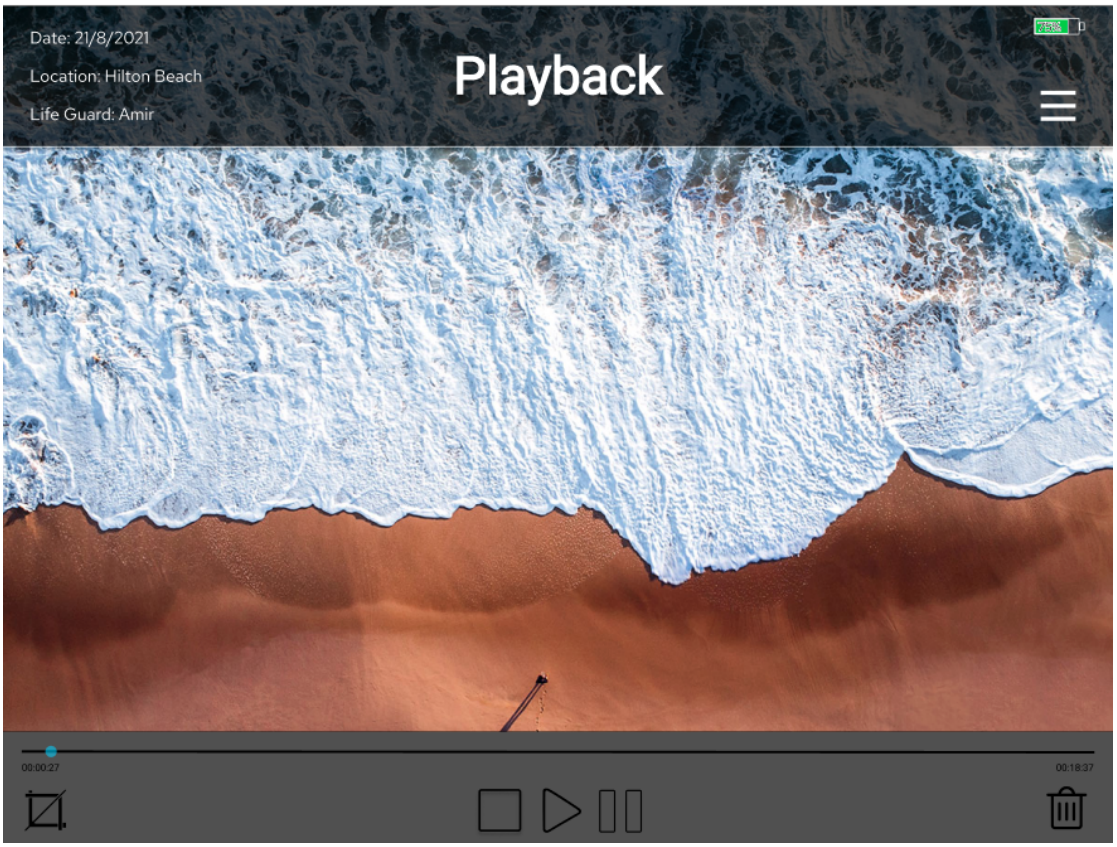


Figure 9.4: Post Mortem, video editing page

10. Appendix-A: Data Security

Data is a crucial part of many systems and it is better to be on the safe side when handling it. In your system, DroneGuard, we can ensure data security in three sensitive points of the system:

1. **Data encryption** - helps protect private information, sensitive data, and can enhance the security of communication between our client app and the server. In essence, when our data is encrypted, even if an unauthorized entity gains access to it, he will not be able to read it.

2. Secure shell (**SSH**) for remote login - we need to connect the RP remotely in order to implement and execute our code. SSH is a reliable and secure way for such action.

11. Appendix-B: POC

We were asked to do a POC that demonstrates the ability to do the transformation from the pixels(screen) to the real world coordinates.
We decided to do that by using a picture taken from a drone, when clicking the images in several points we get the distance from the middle(the drone location) in pixels and then converted to distance in meters.
Then there is a calculation between the current drone location and the selected point, which eventually returns new real-world coordinates that you can easily check in google maps to compare the two points.