Unit 24, Assignment 2

Building Control Systems

George Hotten

February 28, 2023

Designing a Control System

The control system I shall be designing is called Beam Sec. Beam Sec is an aggressive security system used to deal with intruders who break into an area they're not supposed to. Using an IR beam break sensor, the system will detect when someone enters said restricted area. When the break is detected, a drone will be launched. Using the drone's camera sensor, facial recognition shall be used to capture a photo of the intruder and then track their face around the room to follow and attack the intruder. Once the intruder is eliminated, it returns to its previous resting location and resets.

The control system consists of five components:

- Raspberry Pi Pico W - the microcontroller used to instruct the drone via Wi-Fi.
- IR Beam Break Sensor - the sensor used to detect when something goes between them.
- Buzzer - used as an alarm to show the system has been activated.
- 2" OLED display - used as a status indicator and shows logs of the drone's progress.
- DJI Tello Drone - used to attack and eliminate the intruder.

Ensuring the system completes the same actions every time

To ensure the control system completes the same set of actions each time it is used, the Raspberry Pi Pico W will be directly flashed with Beam Sec's Micro Python code which will instruct the system to perform the required actions. As this code will never change, the system will complete the same instructions each time it is used. To ensure the system works correctly after the intruder has been eliminated, the drone returns itself to where it launched and then the Pico W performs a hardware reset, which shuts down and restarts the microcontroller so the code is fully reset and ready to be reused.

Test Plan

During and after the building of Beam Sec, the following tests should be performed:

Test	**Expected Outcome**
Microcontroller connects to Wi-Fi	On startup, the Pico W should connect to the drone's Wi-Fi network.
The beam break is properly detected	Display status is updated and the drone is launched.
Drone instructions work every time	The drone should launch and follow instructions every time without fail.
Drone attacks	The drone flies towards the intruder and attacks them.
Display updates correctly	The display updates the system status and the drone's current action accordingly.
System reset	The drone should return to its original position and the system should reset back to armed.

Inputs and Outputs

Within the system, there are numerous inputs and outputs used to help the system detect intruders and attack them.

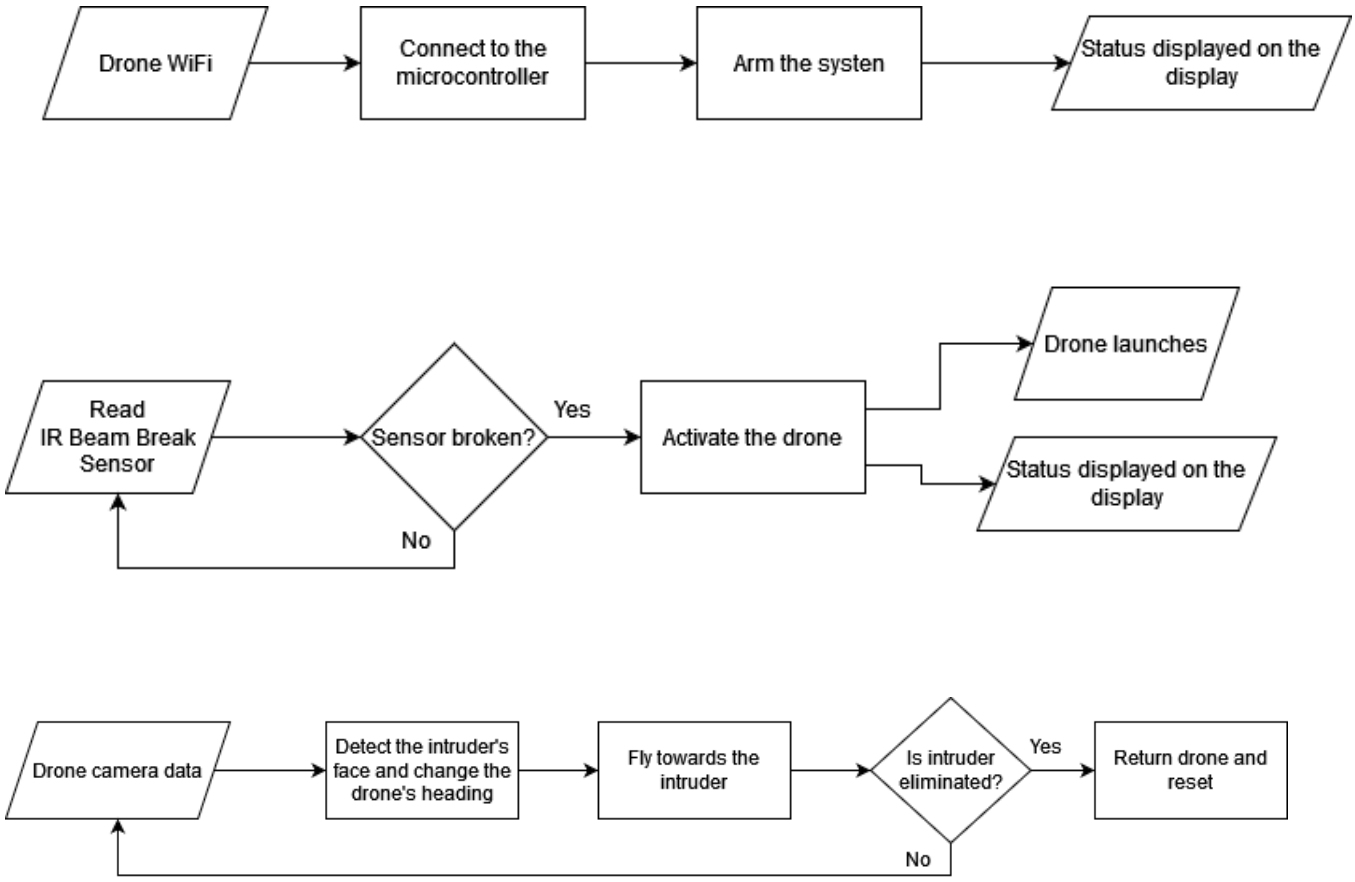

Figure 1: Input and output flowcharts of Beam Sec's control system

Inputs

- Drone Location and Status - the drone has an internal positioning system so the system knows where the drone is and can return it to its resting place. The drone can also provide battery and error information so the system knows when a fault has occurred.
- Drone Camera - the drone's built-in camera sensor sends digital data back to the microcontroller over an analogue wireless signal.
- IR Beam Break sensor - detects when an IR beam is broken and sends a low analogue signal back to indicate the beam is broken.

Outputs

- Drone System - following the microcontroller's commands, the drone completed the required action.
- OLED Display - shows the status of the system using an analogue I^2C connection.
- Buzzer - plays at a specified frequency using an analogue signal to indicate the system has been triggered.

Building the Control System

The Circuit

To connect the different components to the microcontroller, I had to look at the device's pinouts to understand what must be plugged in and where. The Pi Foundation have extensive online documentation to help show what each pin is used for. Using breadboards, jumper wires and resistors, I was able to connect the IR sensor, display and buzzer to the Pico W. To utilize the sensors, I would have to create MicroPython code to detect beam breaks and send instructions to the drone.

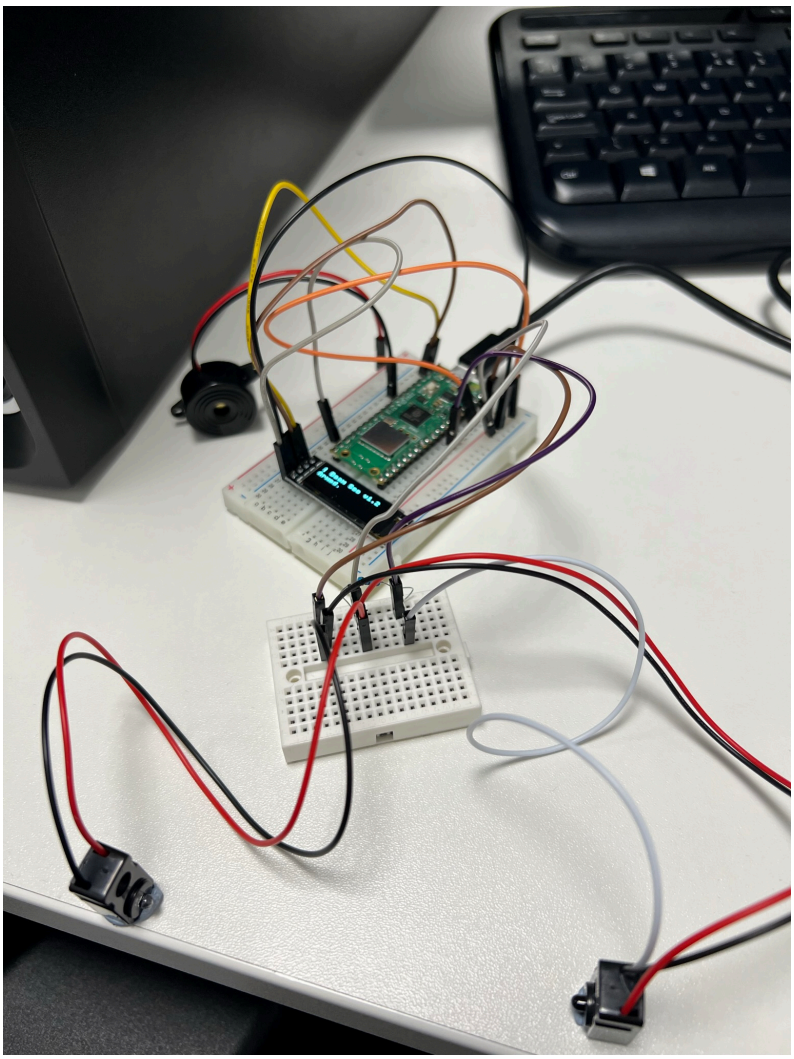

Figure 2: Beam Sec's completed circuit

The Code

Beam Sec's code is built using MicroPython and has had over eight iterations, resulting in its current version being "v1.4.1". Within the code, there are three main components: display, drone control and activation. Beam Sec's source code is publicly available on my GitHub.

Code Flow

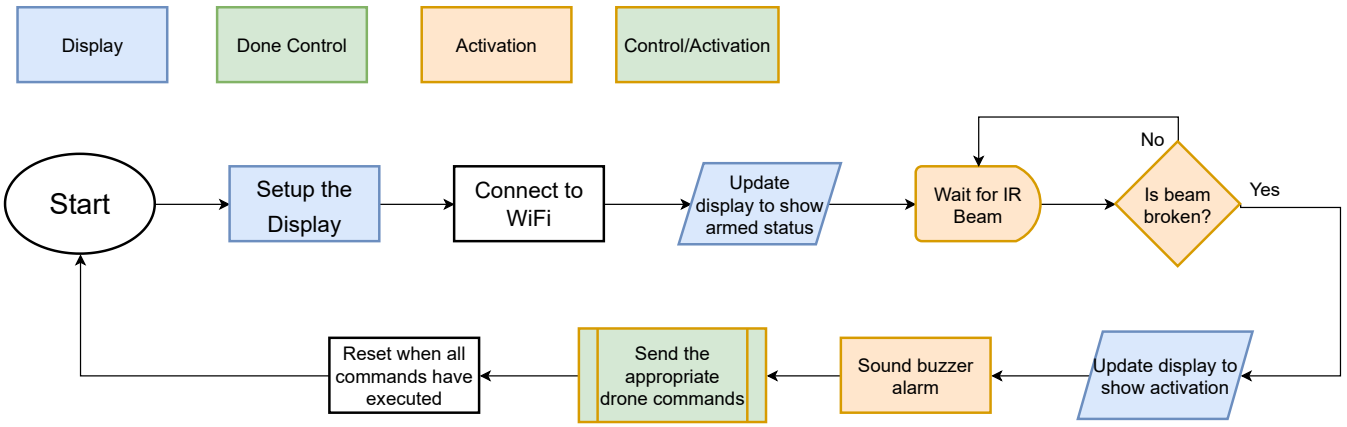

Figure 3: Flowchart of how Beam Sec's MicroPython code functions

Display

Beam Sec uses an SSD1306 OLED display which is controlled via I^2C. I^2C works by using 2 pins: a Serial Data Line (SDA) and a Serial Clock Line (SCL). The SDA line is used to send and receive data between the microcontroller and the display. The SCL line is used to carry the clock signal which is generated by the microcontroller.

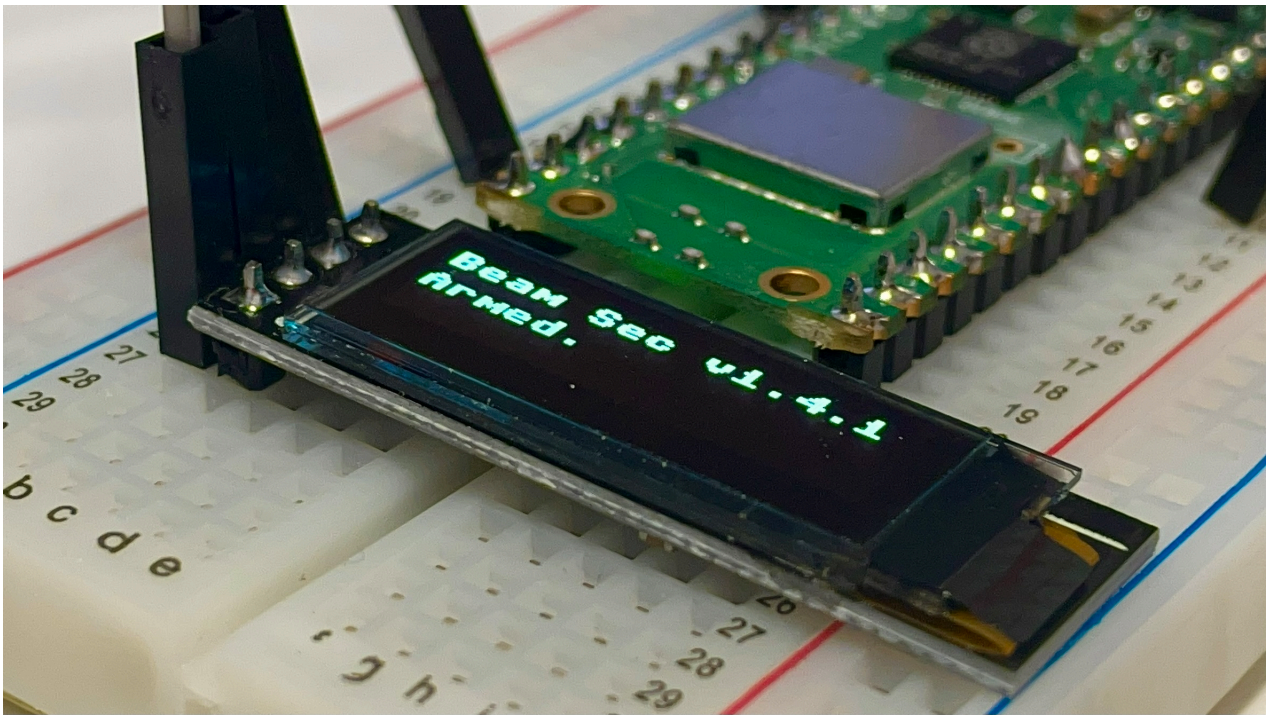

Figure 4: Close-up of the system's display

The code has a repeating function that reads from a top, middle and bottom variable and creates scrolling text to ensure that all information can be shown on the display's 16-character limit. Once the scrolling has been processed, the text is pushed to the display. Before repeating, the system waits for 0.2 seconds.

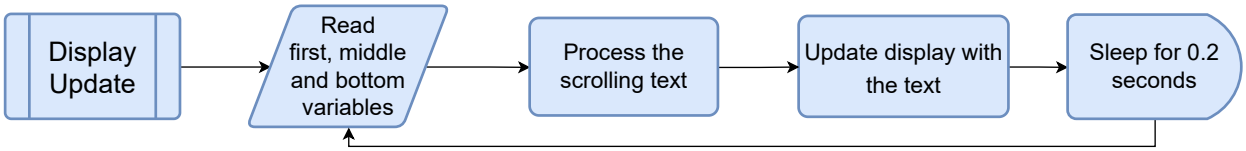

Figure 5: Flowchart of Beam Sec's display update function

For an example of how the display works, view the GIF here.

Drone Control

Beam Sec uses DJI's Tello drones which are controlled via Wi-Fi. The microcontroller connects to the drone's Wi-Fi network and can control it via a socket. The socket is used to send commands listed in Tello's v2 SDK.

Figure 6: DJI "O' Reliable" Tello Drone

The drone control function starts by updating the display with the command it is sending, then using the sockets it sends the command and waits for a response. Once a response is received, the response is logged to the display and the next command is executed.

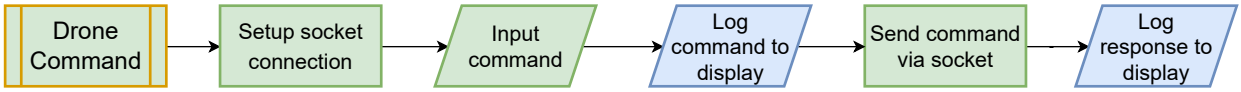

Figure 7: Flowchart of Beam Sec's drone control function

Activation

When in an armed state, Beam Sec is constantly checking the IR sensor's pin status to say if it has a high or low value. If the beam has a low value, it has been broken. Once broken, the system activates. It sounds the buzzer and updates the display to show the system has been activated. Once all the commands have run, the system restarts and re-arms.

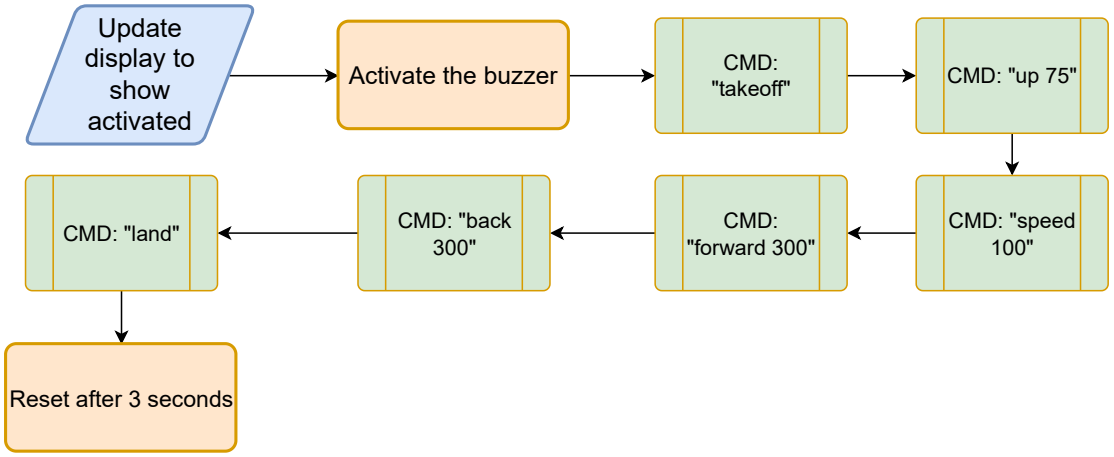

Figure 8: Flowchart of Beam Sec's activation function

Testing the Control System

These tests were conducted on Beam Sec's latest version as of February 15, 2023: v1.4.1.

Test #1: Microcontroller connects to Wi-Fi

Result: **INCONSISTENT**

Expected Outcome On startup, the Pico W should connect to the drone's Wi-Fi network.

Actual Outcome The microcontroller tends to connect to the drone after one failed attempt. The cause of this is unknown, but I believe this is due to my microcontroller timing out as the drone can take a while to connect. See the recording here.

Test #2: The beam break is properly detected

Result: **PASS**

Expected Outcome Display status is updated and the drone is launched.

Actual Outcome The microcontroller can consistently detect when the IR beam is broken and will launch the drone every time. See the recording here.

Test #3: Drone instructions work every time

Result: **FAIL**

Expected Outcome The drone should launch and follow instructions every time without fail.

Actual Outcome Whilst the drone is able to receive and attempt to execute all the commands given to it, it often fails to execute under the error "no valid imu". The drone being used in this system is old and is unable to calibrate to resolve the error. See the recording here.

Test #4: Drone attacks

Result: **PASS**

Expected Outcome The drone flies towards the intruder and attacks them.

Actual Outcome The drone successfully flies forwards towards the intruder. See the recording here of the intruder who happens to be a ghost, and therefore invisible!

Test #5: Display updates correctly

Result: **PASS**

Expected Outcome The display updates the system status and the drone's current action accordingly.

Actual Outcome The system updates the current command being executed correctly throughout the drone's activation. See the recording here.

Test #6: System reset

Result: **PARTIAL PASS**

Expected Outcome The drone should return to its original position and the system should reset back to armed.

Actual Outcome Whilst the system successfully resets itself and is re-armed, the drone fails to return to its original position and often lands where it moved to, due to the failed backwards command of the "no valid imu" error. See recording here.

Evaluating the Control System

What went well?

Overall, most of Beam Sec's design implementation was a success. Beam Sec was successfully able to detect an intruder via an invisible IR beam break sensors and launch a drone in response to eliminate the intruder.

Beam Sec utilized cheap and available hardware which allows the system to easily be built by anyone, as no extensive knowledge of electronics is required. With this hardware, Beam Sec can utilize DJI's simple API for their Tello drone via sockets over Wi-Fi. This allowed for easy control over the drone using basic commands such as "forward" and "back". With this, Beam Sec's code was able to send commands to the drone and have them execute instantly, allowing for the drone to be used to attempt to eliminate the intruder.

Looking further at the hardware, specifically the small 2" display, I was able to get around the limitation of display's 16-character limit by implementing scrolling text so that longer messages could be displayed. After spending time testing the speed at which the display should scroll, I settled on updating every 0.2 seconds. This allows the text to move fast whilst still being readable, allowing users to understand what the system is trying to output as fast as possible.

On the software side, Beam Sec's MicroPython code has worked flawlessly. The code performs the appropriate instructions every time without fail, and can quickly respond to unforeseen errors whilst running. For example, if executing a drone command throws an error, the system will move onto the next command without outright crashing.

What didn't go well?

Unfortunately, not everything went to plan with Beam Sec's design implementation. During testing, it was found that the DJI Tello drones lack reliability due to their old age. This caused issues where the drone was unable to perform instructions due to bad calibration, which because of the drone's age could not be rectified. Attempts were made to recalibrate, however those attempts were unsuccessful. It was also found that the drones lack accuracy. For example, if the command "forward 100" was executed, the drone would often move too far, too little or not move at all. If the drone did move, it would often drift left and right.
Of all the command that Beam Sec utilized, "back" seemed to be the most unreliable, with the drone only following the command less than 50% of the time. This most commonly failed because of the "no valid imu" error which is often caused by bad calibration.

Problems with the design and hardware

Due to the small nature of the built circuit and the hardware used, the IR sensor is unable to cover a large area and because of the cables it is pretty obvious that there is a sensor present. The cables used with the sensor are small further prevent a large area to be used. This limits the system's usage to small walking areas where the microcontroller and the rest of the circuit can be very close by.

The DJI Tello drones have a built-in auto sleep after around 5 minutes. This creates a flaw in the device as once the device has gone to sleep, the drone must be manually turned back on and any commands sent to drone aren't received.

Did it meet the design criteria?

Unfortunately the system does not fully meet the design criteria. Beam Sec is missing the planned facial recognition and photo taking ability used to attack the intruder. Because of this lack of implementation, the drone is programmed to move forward towards where the IR sensor is placed and then move backwards shortly after. To ensure this works, the drone must be placed facing towards the IR sensor within 300 meters.

Excluding the above-mentioned features, Beam Sec is fully implemented inline with the design criteria. Beam Sec is able to detect breaks in the IR beam, connected to the DJI Tello drone and attempt to attack the intruder. Once it is finished, it returns to its starting position and resets.

Potential improvements

More powerful drone

One of the biggest flaws within Beam Sec was the drone used, a Ryze DJI Tello. The drone was unreliable, didn't follow commands and was overall very unstable. This caused major issues within the system causing unexpected and unwanted actions to be performed by the drone. To improve the system, I would upgrade to a more powerful drone. Specifically the DJI Matrice 300 RTK. The Matrice 300 is an enterprise grade drone that is used for tracking, detecting and capturing footage of specific objects. This would give Beam Sec the capability of using facial recognition and tracking the intruder to eliminate it. As this is an enterprise grade drone, the issues of drifting and failing to follow commands will be eliminated. The drone also contains a mounting hole, which could be used to attach the appropriate intruder elimination objects to.

Including facial recognition

With the more powerful drone, Beam Sec could utilize facial recognition to enable it to follow the intruder around. This can be especially helpful if the intruder enters an area where there are lots of people as it will be able to lock onto the intruder and not start tracking the wrong person. This could allow a video stream to be sent back to a security control centre so they know where the intruder is at all times and will provide a view on what the intruder looks like so the appropriate information can be passed onto law enforcement.

More appropriate IR sensor

The current IR sensor used within Beam Sec is small and does not have the long range that would be required when deployed in production. The sensor is also not able to move far from the microcontroller due the short length of the cables. This can be improved by purchasing a larger IR sensor, with longer cables or with wireless capability. Wireless capability would be the best upgrade path as it allows for the sensor to be more hidden and reduces the restrictions introduced by cables. As the microcontroller, a Pi Pico W, has a built-in Wi-Fi chip, the activation from the IR sensor could be received wirelessly. This would only require small changes to the software, just implementing and connection and then establishing a channel between the microcontroller and the sensor to communicate when the beam is broken.