PDE3413 - FINAL PROJECT PRESENTATION SLIDES

ALARMBUZZ
AUTONOMOUS BURGLAR DETECTION ROBOT

Presented by: Dereck Lam Hon Wah (M00826933)

Date of Submission: 03.04.2023

Lab Tutor: Praveer Towakel

(Call Centre Helper, n.d.)

PROBLEM BEING SOLVED ?

Increase in home invasions and burglaries that impacts economy and public safety. Every 25.7 seconds, or over 3,300 times each day, a robbery takes place (Denver Criminal Defense Attorney | Criminal Lawyer Denver, Colorado, 2021).

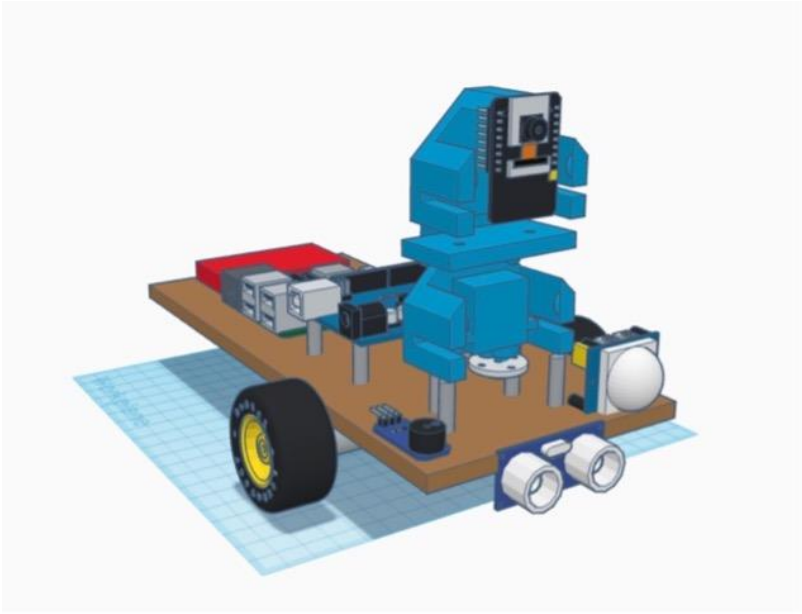
Less than 30% of buildings have effective security systems; 300% more likely to be robbed (Homan, n.d.).

Unaffordable for low-income homeowners and small businesses (NBC News, n.d.).

Emotional and financial distress for victims; only one-third feel healed after a month (www.ojp.gov, n.d.).

AIMS AND OBJECTIVES

- Enhance sense of security and offers a defense against the rising rate of burglaries.
- Provide an affordable and effective security solution for homeowners and small businesses
- Utilize computer vision and AI for accurate human intrusion detection even in dark environment
- Minimize false alarms by differentiating between humans and animals
- Allow remote access for monitoring and cloud storage for real-time footage as evidence.



Nimbo (hellonimbo.com, n.d.)

EXISTING SOLUTION- NIMBO

- Intelligent security robot showcased at CES 2018
- Incorporates advanced technologies: LIDAR, AI, machine learning, SLAM, IoT, GPS, etc (in.micron.com, n.d.).
- AI-powered for critical thinking and learning from past performance
- Equipped with video analytics platform to recognize human and animal invasions
- IoT integration for push notifications, email alerts, evidence review, and cloud storage
- Can patrol predetermined or self-optimized routes while monitoring surroundings
- Offers HD video, two-way audio, auto charging, and continuous video history (www.securityinfowatch.com, n.d.)
- **Not suitable** for residential use due to high cost ($12 per hour)

TECHNOLOGIES USED

- Autonomous Navigation
- Computer Vision
- Human Detection
- Real time alert mechanism

(Canonical, 2023)

(Python.org, 2019)

(Discord, n.d.)

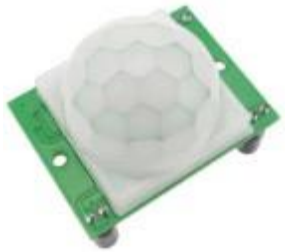
(GitHub, n.d.)

(GitHub, n.d.)

SENSORS/EQUIPMENT USED

(grobotronics.com, n.d.)

Pi camera v2



(grobotronics.com, n.d.)

HC-SR501
PIR Sensor

(Sparkfun.com, 2017)

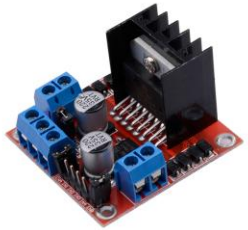
HC-SR04
Ultrasonic
Distance Sensor

(grobotronics.com, n.d.)

Arduino Sensor Shield
V5.0

(grobotronics.com, n.d.)

5V DC
Gear Motor

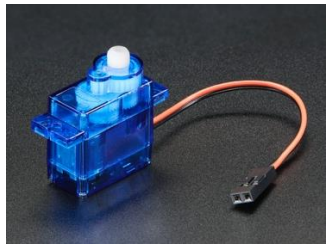


(Amazon, n.d.)

L298N
Motor Driver

(Ardumotive Arduino Greek Playground, n.d.)

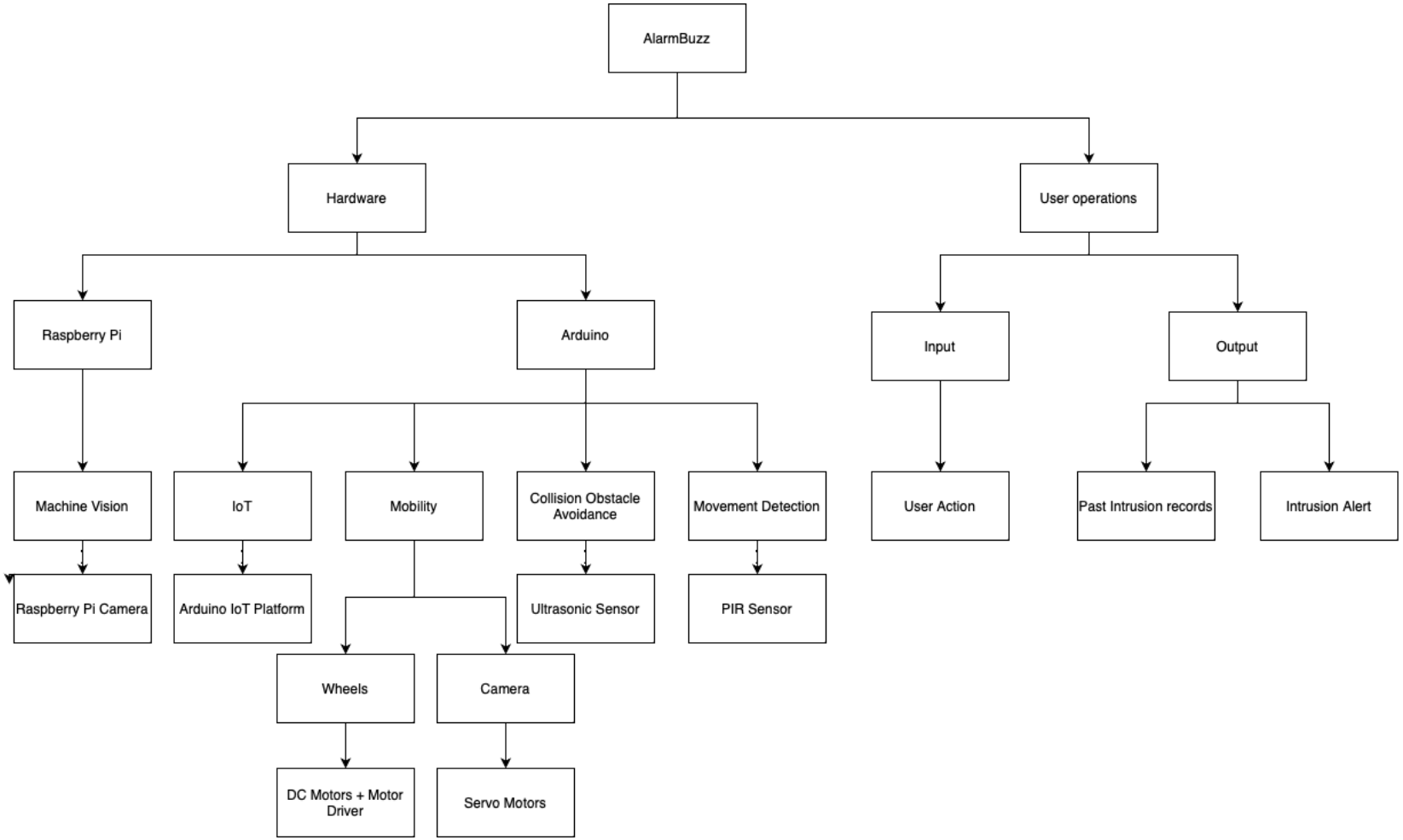
Passive Buzzer
Module



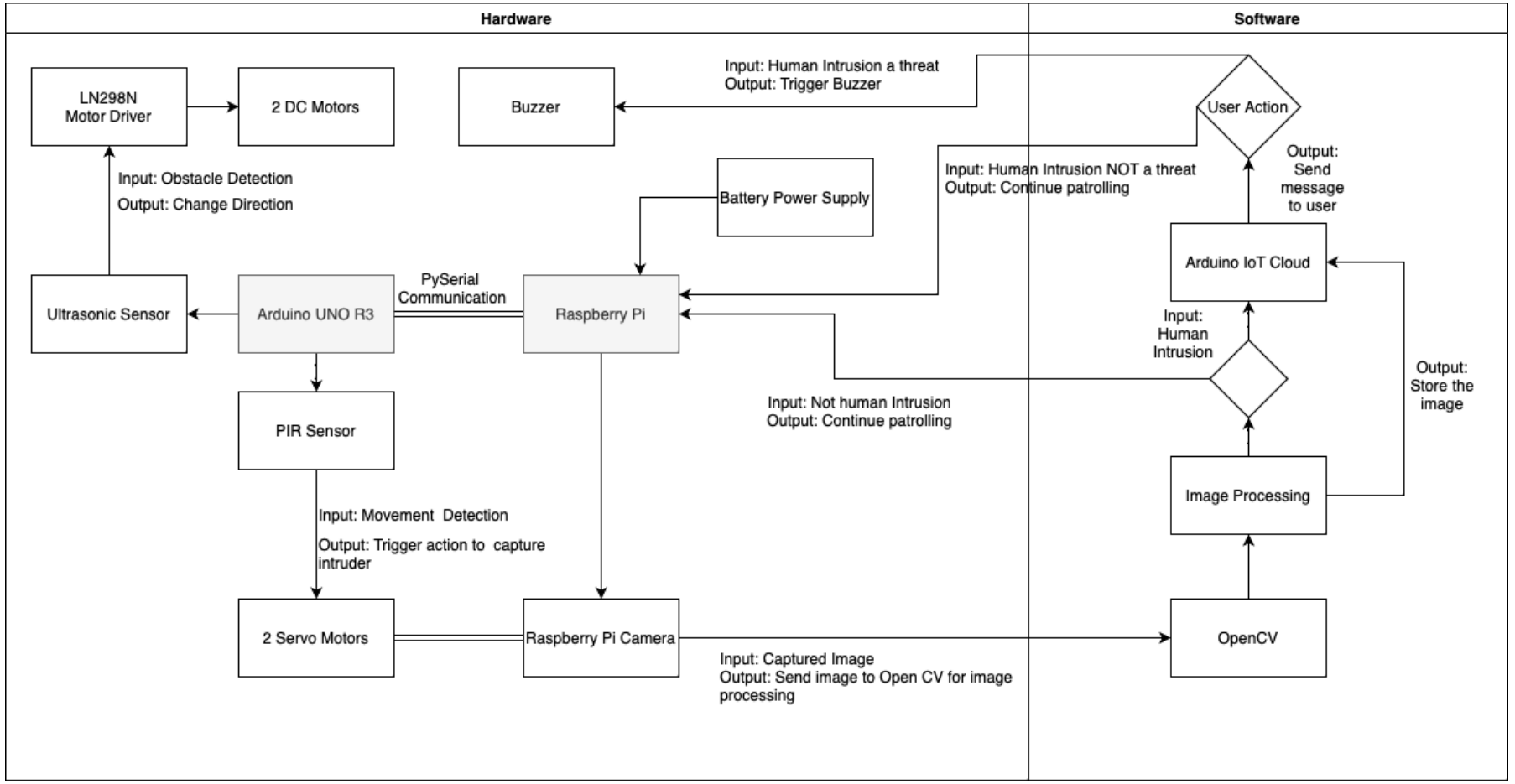
(Instructables, 2015)

FS90R
Servo Motor

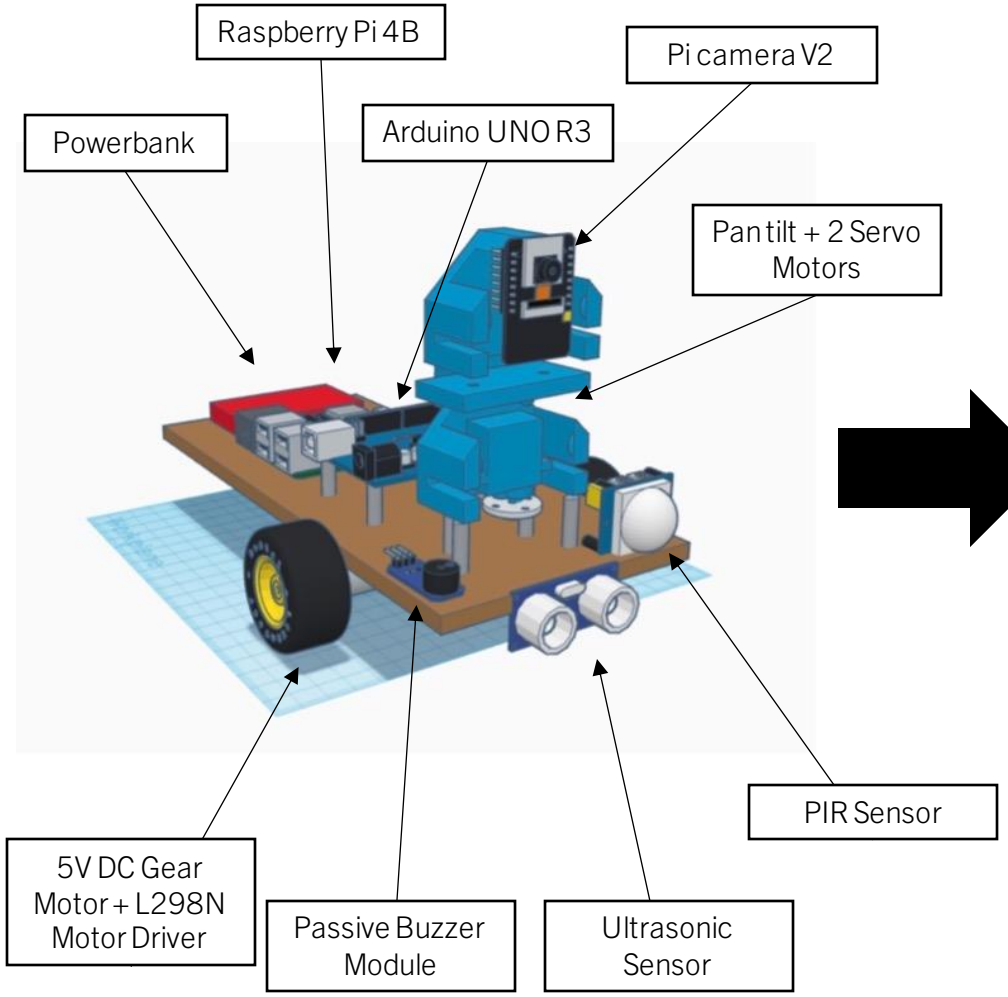
SYSTEM BREAKDOWN

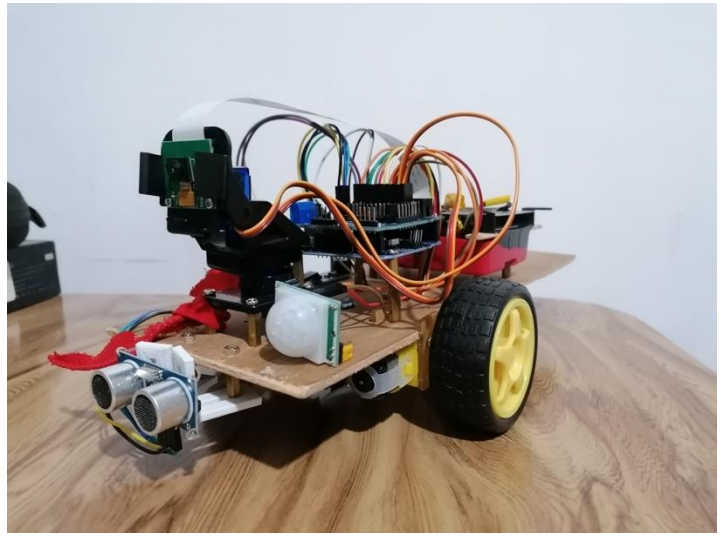


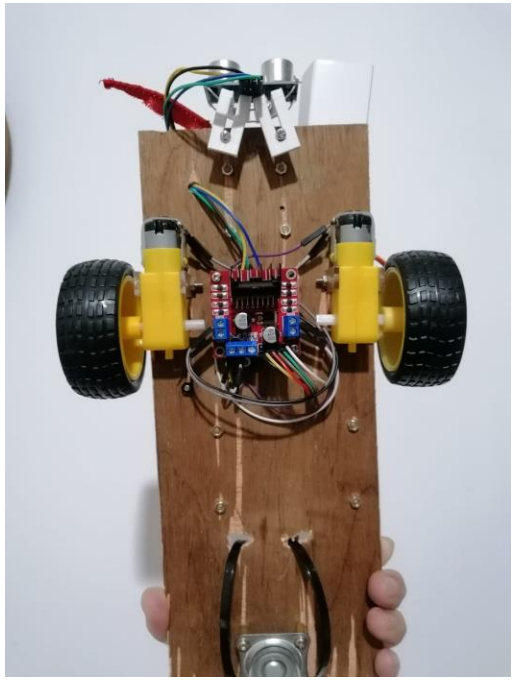
CONCEPT BEHAVIOUR



HARDWARE IMPLEMENTATION







SOFTWARE IMPLEMENTATION

Main.py

```
You, 4 days ago | 1 author (You)
import serial
import time
import sys
import os
import cv2      You, 4 days ago • feat: integrate human detection and send a signal…

from humanDetection import hogDescriptor

if __name__ == "__main__":
    print("╔═══════════════════════════════╗")
    print("║        Welcome to AlarmBuzz!        ║")
    print("╚═══════════════════════════════╝")

    # Wait for 1 second before continuing
    time.sleep(1)

    # Get the user's name
    name = input("What's your name? ")

    # Greet the user
    print(f"\nHi {name}! Let's set up AlarmBuzz duration.\n")

    # Wait for 1 second before continuing
    time.sleep(1)

    # Define the duration of the loop in seconds
    while True:
        try:
            duration = int(input("How long do you want the alarm buzz to run for? (in seconds
            break
        except ValueError:
            print("Sorry, AlarmBuzz only accept integer :(\n")

    # Confirm the alarm time and duration
    print(f"\nAlarmBuzz will run for {duration} seconds. Stay safe!\n")

    # Open a serial connection and write the 'start' command to the microcontroller
    ser = serial.Serial('/dev/ttyUSB0', 9600, timeout=1)
    ser.flush()
    time.sleep(2)
    ser.write(b'start\n')

    # Get the current time
    start_time = time.time()

    try:
        # Loop the script for the specified duration
        while (time.time() - start_time) < duration:
            # Check if there is any data in the serial buffer
            if ser.in_waiting > 0:
                # Read a line from the serial port
```

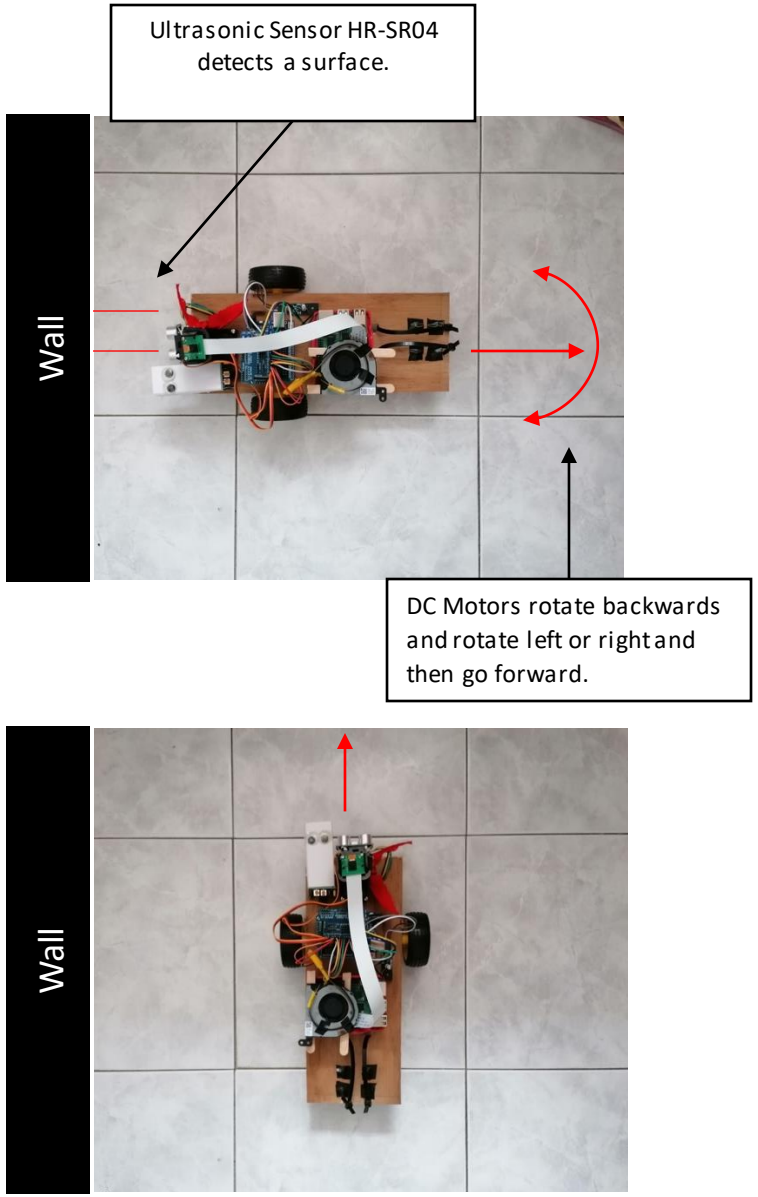
```
try:
    # Loop the script for the specified duration
    while (time.time() - start_time) < duration:
        # Check if there is any data in the serial buffer
        if ser.in_waiting > 0:
            # Read a line from the serial port
            line = ser.readline().decode().rstrip()
            if line == "motion":
                # Initialize a HumanDetector object and check if humans are detected
                detector = hogDescriptor.HumanDetector(0)
                if (detector.detect_humans()):
                    # If humans are detected, write 'human' to the serial port and print a message
                    ser.write(b'human')

except KeyboardInterrupt:
    print("AlarmBuzz has been stopped.")

print("Thanks for using AlarmBuzz! Sweet dreams and have a great day ahead!")

# Close the serial connection and write the 'stop' command to the microcontroller
ser.write(b'stop\n')
ser.close()
```

```
❯ python3 main.py
╔═══════════════════════════════╗
║        Welcome to AlarmBuzz!        ║
╚═══════════════════════════════╝
What's your name? dereck

Hi dereck! Let's set up AlarmBuzz duration.

How long do you want the alarm buzz to run for? (in seconds) 60
```

Obstacle Avoidance System

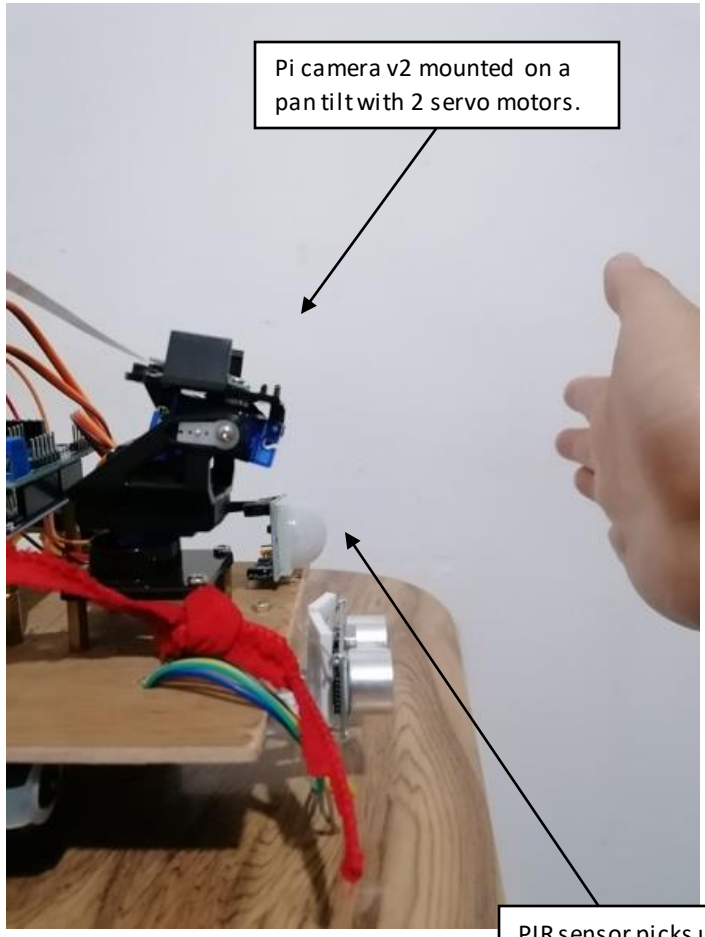
```
/*
 * Returns the distance to the obstacle as an integer
 */
int doPing()
{
    int distance = 0;
    int average = 0;

    // Grab four measurements of distance and calculate
    // the average.
    for (int i = 0; i < 4; i++)
    {

        // Make the Trigger LOW (0 volts)
        // for 2 microseconds
        digitalWrite(Trigger, LOW);
        delayMicroseconds(2);

        // Emit high frequency 40kHz sound pulse
        // (i.e. pull the Trigger)
        // by making Trigger HIGH (5 volts)
        // for 10 microseconds
        digitalWrite(Trigger, HIGH);
        delayMicroseconds(10);
        digitalWrite(Trigger, LOW);

        // Detect a pulse on the Echo pin 8.
        // pulseIn() measures the time in
        // microseconds until the sound pulse
        // returns back to the sensor.
        distance = pulseIn(Echo, HIGH);

        // Speed of sound is:
        // 13511.81023622 inches per second
        // 13511.81023622/10^6 inches per microsecond
        // 0.013511811 inches per microsecond
        // Taking the reciprocal, we have:
        // 74.00932414 microseconds per inch
        // Below, we convert microseconds to inches by
        // dividing by 74 and then dividing by 2
        // to account for the roundtrip time.
        distance = distance / 74 / 2;

        // Compute running sum
        average += distance;

        // Wait 10 milliseconds between pings
        delay(10);
    }
    // Return the average of the four distance
    // measurements
    return (average / 4);
}

/*
 *  Forwards, backwards, right, left, stop.
 */
void goForward()
{
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
}
void goBackwards()
{
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
}
void goRight()
{
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
}
void goLeft()
{
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
}
void stopAll()
{
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
}
```

```
/*
 * Obstacle detected, avoid it
*/
void avoidObstacle()
{       You, 5 days ago • feat: integrate pir sensor reading …
    int distance = doPing();

    // If obstacle <= 10 inches away
    if (distance >= 0 && distance <= 12)
    {
        // Serial.println("Obstacle detected ahead");
        goBackwards(); // Move in reverse
        delay(1000);

        /* Go left or right to avoid the obstacle*/
        if (random(2) == 0)
        {               // Generates 0 or 1, randomly
            goRight(); // Turn right
        }
        else
        {
            goLeft(); // Turn left
        }
        delay(1500);
        goForward(); // Move forward
        // Serial.println("Obstacle detected completed");
    }
    delay(50); // Wait 50 milliseconds before pinging again
}
```



Motion detection

```
if (start_flag) {
    val = digitalRead(inputPin); // read input value
    if (val == HIGH)
    {                            // check if the input is HIGH
        digitalWrite(ledPin, HIGH); // turn LED ON
        if (pirState == LOW)
        {
        // we have just turned on
        // We only want to print on the output change, not state
        pirState = HIGH;
        stopAll();
        delay(200);

        sweepStart();
        Serial.println("motion");
        sweepRunning();
        sweepEnd();
        }
    }
    else
    {
        digitalWrite(ledPin, LOW); // turn LED OFF
        if (pirState == HIGH)
        {
        // we have just turned of
        // Serial.println("Motion ended!");
        // We only want to print on the output change, not state
        pirState = LOW;
        goForward();
        }
    }
    avoidObstacle();
}
```

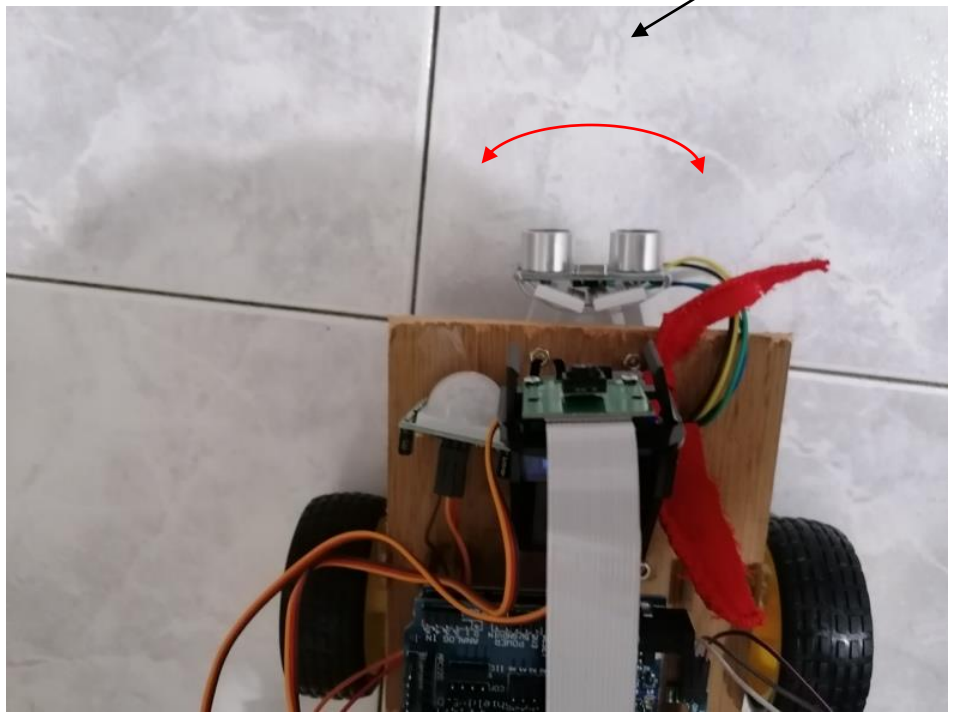


Pan tilt sweep movement

Servo motors are triggered and position the pan tilt to start sweeping the area in a 180-degree motion for 20secs.

```
 * Sweep Start movement
*/
void sweepStart()
{
    for (posY = posY; posY <= 45; posY += 1)
    {
        servoY.write(posY); // move the vertical servo
        delay(50);          // delay between movements
    }

    for (posX = posX; posX >= 0; posX -= 1)
    {
        servoX.write(posX); // move the horizontal servo
        delay(50);          // delay between movements
    }
}

/*
* Sweep Running movment
*/
void sweepRunning()
{
    unsigned long start_time = millis(); // get the current time
    while (millis() - start_time < 10000)
    { // run the loop for 10 seconds

        for (posX = posX; posX <= 180; posX += 1)
        {
            servoX.write(posX); // move the horizontal servo
            delay(50);          // delay between movements

            if (Serial.available() > 0 && Serial.readString() == "human")
            {
                sweepEnd();
                buzzerTone();
                return;
            }
        }
        for (posX = posX; posX >= 0; posX -= 1)
        {
            servoX.write(posX); // move the horizontal servo
            delay(50);          // delay between movements

            if (Serial.available() > 0 && Serial.readString() == "human")
            {
                sweepEnd();
                buzzerTone();
                return;
            }
        }
    }
```

```
/*
 * Sweep End Movement
*/
void sweepEnd()
{
    // stop the servos and wait for 1 second before starting the next loop
    for (posX = posX; posX <= 90; posX += 1)
    {
        servoX.write(posX); // move the horizontal servo
        delay(50);          // delay between movements
    }

    for (posY = posY; posY >= 0; posY -= 1)
    {
        servoY.write(posY); // move the vertical servo
        delay(50);          // delay between movements
    }
}
```

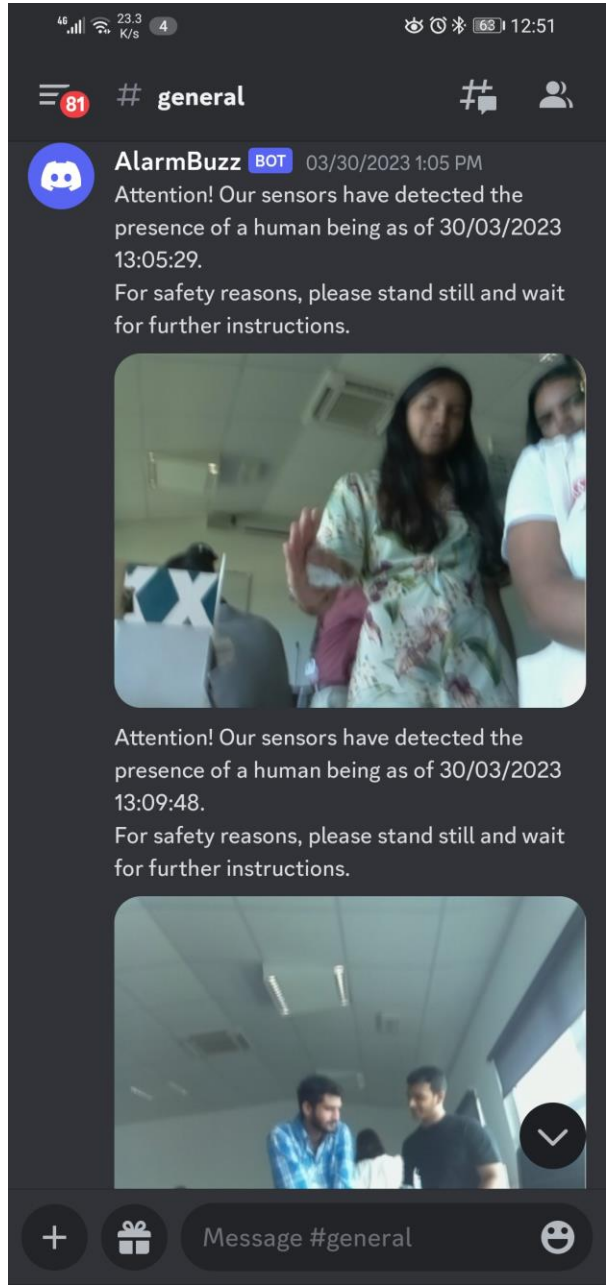


Human Detection Algorithm – HOG Descriptor

```python
import cv2
import imutils
import time
from dat
from dho

class Hu
    def __init__(self, camera_id):
        # Initialize the HOG descriptor with the default people detector
        self.hog = cv2.HOGDescriptor()
        self.hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

        # Initialize the video capture device
        self.cap = cv2.VideoCapture(camera_id, cv2.CAP_V4L2)

        # Set the duration of the human detection loop in seconds
        self.duration = 20

    def detect_humans(self):
        # Get the current time in seconds
        start_time = time.time()

        # Loop until the duration of the human detection loop has elapsed
        while (time.time() - start_time) < self.duration:
            # Read a frame from the video capture device
            ret, src = self.cap.read()

            # Flip the frame vertically to correct the orientation
            image = cv2.flip(src, 0)

            if ret:
                # Resize the frame to a smaller size to speed up the detection
                image = imutils.resize(image,
                                       width=min(400, image.shape[1]))

                # Detect humans in the frame using the HOG descriptor
                (regions, _) = self.hog.detectMultiScale(image,
                                                         winStride=(4, 4),
                                                         padding=(4, 4),
                                                         scale=1.05)

                # If humans are detected, take a photo and send a notification to Discord
                for (x, y, w, h) in regions:
                    # cv2.rectangle(image, (x, y),
                    #               (x + w, y + h),
                    #               (0, 0, 255), 2)

                    # Take a photo of the frame when humans are detected
                    current_time = datetime.now()
                    current_time_image_format = current_time.strftime("%Y%m%d_%H%M%S")
                    human_detection_image_path = f"images/photo_{current_time_image_format}.jpg"
                    cv2.imwrite(human_detection_image_path, image)

                    # Release the video capture device and close the image window
                    self.cap.release()
                    cv2.destroyAllWindows()

                    # Send a message and image to Discord using a Webhook object
                    hook = Webhook("https://discord.com/api/webhooks/1090684363714351144/NMGFwquH-LvDjJBHL8SmxQtXRjFjT8ZQ8Nqb-r8IM2P7fkqLJHylM1jY138FRv4zH3Us")

                    # Format the current time as a string
                    dt_string = current_time.strftime("%d/%m/%Y %H:%M:%S")

                    # Create a message with the current time and instructions for the user
                    data = f"Attention! Our sensors have detected the presence of a human being as of {dt_string}.\nFor safety reasons, please stand still and wait
                    human_detection_image = File(human_detection_image_path)

                    # Send the message and image to the Discord webhook
                    hook.send(data, file=human_detection_image)

                    print("AlarmBuzz has detected a human presence! Please check your surroundings.\n")
                    return True

                # Display the image with bounding boxes around detected humans
                cv2.imshow("Image", image)

                # Wait for a key press and exit the loop if the 'q' key is pressed
                if cv2.waitKey(25) & 0xFF == ord('q'):
                    break
            else:
                break

        # Release the video capture device and close the image window when done
        self.cap.release()
        cv2.destroyAllWindows()

if __name__ == "__main__":
    # Create a HumanDetector object with the default camera ID (0)
    detector = HumanDetector(0)

    # Call the detect_humans method to start detecting humans
    detector.detect_humans()
```

Discord Webhooks

```python
# Take a photo of the frame when humans are detected
current_time = datetime.now()
current_time_image_format = current_time.strftime("%Y%m%d_%H%M%S")
human_detection_image_path = f"images/photo_{current_time_image_format}.jpg"
cv2.imwrite(human_detection_image_path, image)

# Release the video capture device and close the image window
self.cap.release()
cv2.destroyAllWindows()

# Send a message and image to Discord using a Webhook object
hook = Webhook("https://discord.com/api/webhooks/1090684363714351144/NMGFwquH-LvDjJBHL8SmxQtKRjFjT8ZQ6Nqb-r8IMZP7fkqLJHylMijY1J8FRv4zH3Us")

# Format the current time as a string
dt_string = current_time.strftime("%d/%m/%Y %H:%M:%S")

# Create a message with the current time and instructions for the user
data = f"Attention! Our sensors have detected the presence of a human being as of {dt_string}.\nFor safety reasons, please stand still and wait for further instructions."
human_detection_image = File(human_detection_image_path)

# Send the message and image to the Discord webhook
hook.send(data, file=human_detection_image)
```



TESTING

Component testing

System requirements		
Hardware	ID	Requirements
Arduino Uno R3	HA01	Communication established with Raspberry PI 4
	HA02	Communication with Ultrasonic and PIR sensor should be established
	HA03	Communication with actuators (motor driver, dc motors, servo motors and buzzer) should be established.
	HA04	Instructions written in C++ using the Arduino IDE.
	HA05	Connection with the Arduino IoT cloud.
	HA06	Process instructions to use Arduino IoT cloud webhooks to send communication to alert the user.
	HA06	Process instructions to send the snapchats for storage on the cloud.
Pi Camera V2	HPC01	Communication established with Raspberry PI 4
	HPC02	Instructions written in python using OpenCV for object detection and image processing to distinguish between human and animal intruder.
	HPC03	Communication with Arduino when PIR sensor detect a movement to trigger the camera functionality.
	HPC04	Communication with servo motors and Arduino to calibrate itself to focus and take a snapshot of the intruder.
Ultrasonic Sensor HC-SR04	HUS01	Communication with the Arduino to be established.
	HUS02	Read and update distance.
	HUS03	Data collected should be communicated to Arduino for process.
FS90R Servo Motor	HSM01	Communication with the Arduino to be established.
	HSM02	Smooth and stable movement to calibrate the camera.
	HSM03	Follow instructions provided by the pi camera and Arduino.
SR501 PIR sensor	HPIR01	Communication with the Arduino to be established.
	HPIR02	Read and detect movement.
	HPIR03	Data collected should be communicated to Arduino for process.
Raspberry Pi4	HRP01	Communication established with Arduino UNO R3.
	HRP02	Instruction written in python.
	HRP03	Connection with the Pi camera v2 using OpenCV for image processing and object detection.
	HRP04	Connection with the servo motors as intermediate for the PI camera for its calibration.
Buzzer	HBUZ01	Communication with the Arduino to be established.
	HBUZ02	Follow instructions provided by the Arduino.
Motor Driver LM298N	HMD01	Communication with the Arduino to be established.
	HMD02	Follow instructions provided by the Arduino.
	HMD03	Avoid collision with obstacle.
	HMD04	Lightweight but powerful.
5VDC Motors	HDC01	Communication with the Motor driver to be established.
	HDC02	Follow instructions provided by the motor driver.
	HDC03	In good condition for smooth mobility.

Unit Testing

Unit Test Cases		
Hardware	ID	Requirements
Arduino Uno R3	HA01	Test communication established with Raspberry PI 4
	HA02	Test Communication with Ultrasonic and PIR sensor.
	HA03	Test Communication with actuators (motor driver, dc motors, servo motors and buzzer).
	HA05	Test connection with the Arduino IoT cloud.
	HA06	Use Arduino IoT cloud webhooks to send dummy communication to alert the user.
	HA06	Trial access of the snapchats for storage on the cloud.
Pi Camera V2	HPC01	Test Communication established with Raspberry PI 4
	HPC02	Using OpenCV test for object detection and image processing to distinguish between human and animal intruder.
	HPC03	Test Communication with Arduino when PIR sensor detect a movement to trigger the camera functionality.
	HPC04	Test the movement of the servo motors and Arduino to calibrate itself to focus and take a snapshot of the intruder. Implement sudden movement to test if the servo motor can cope with.
Ultrasonic Sensor HC-SR04	HUS01	Test Communication with the Arduino.
	HUS02	Compare distance with real life values.
	HUS03	Using the serial to see if the Arduino are reading the data correctly.
FS90R Servo Motor	HSM01	Test Communication with the Arduino to be established.
	HSM02	Test the Smooth and stable movement to calibrate the camera. Implementing sudden movement.
	HSM03	Test the Smooth and stable movement to calibrate the camera. Implementing sudden movement.
SR501 PIR sensor	HPIR01	Test Communication with the Arduino to be established.
	HPIR02	Using the serial to see if the Arduino are reading the data correctly.
	HPIR03	Using the serial to see if the Arduino are reading the data correctly.
Raspberry Pi4	HRP01	Test Communication established with Arduino UNO R3.
	HRP03	Using OpenCV test for object detection and image processing to distinguish between human and animal intruder.
	HRP04	Test the Smooth and stable movement to calibrate the camera. Implementing sudden movement.
Buzzer	HBUZ01	Test Communication with the Arduino to be established.
	HBUZ02	Test when instructions passed if it emit a sound.
Motor Driver LM298N	HMD01	Test Communication with the Arduino to be established.
	HMD02	Using the serial to see if the Arduino are reading the data correctly from the ultrasonic sensor.
	HMD03	Create a track to test the avoid collision with obstacle.
	HMD04	Test if powerful enough to run with all components on.
5VDC Motors	HDC01	Test Communication with the Motor driver to be established.
	HDC02	Using the serial to see if the Arduino are reading the data correctly from the ultrasonic sensor.
	HDC03	Do a test run to see if it is operating smoothly

TESTING

Integration Testing

Integration Test Cases			
Test Name	ID	Requirements	Results
Obstacle Avoidance System	IT01	Two 5V DC motors are connected to a L298N motor driver which will read the output of an ultrasonic sensor to see if there is a surface in front of AlarmBuzz.	If the distance read from the ultrasonic exceeds the specified, there are no obstacles ahead, and it will continue to move forward. If the distance is closer than the specified distance, then an obstacle is in front of AlarmBuzz. It will therefore stop in this position, go backward, pause briefly again, and then turn in a random different direction.
Motion Detection trigger Pan tilt and pi camera activation	IT02	A PIR sensor is connected to the front of AlarmBuzz which will detect an IR light emitted from a hot-blooded creature which will trigger the positioning of the pan tilt by the two servos and activate the pi camera live stream.	Motion detected by the PIR sensor triggered the positioning of the pan tilt in a timely manner and activate the camera for 20 secs and if no human detection found reposition the pan tilt to the unactive mode.
Human Detection Algorithm	IT03	Among the three selected algorithm, haarcascade, HOG descriptor and trained model, execution speed and performance to detect a human figure and no ghost human detected in the background.	After some tests, HOG descriptor has been selected as human detection algorithm for its execution speed and accuracy compared to the other two.
Human Detection and Pan tilt unactive mode	IT04	IT02 and IT03 need to be satisfied for the following test to be carried. It consists of moving the pan tilt position to unactive mode when a human has been detected.	Overall good performance, sometimes has a small delay in terms of repositioning of the horizontal servo but is okay in terms of prototype.
Connect to Arduino IoT cloud	IT05	IT03 need to be satisfied for the following test to be carried. Upon human detection, the date time of the intrusion and captured image is sent to the Arduino cloud for processing.	Failed. Lack of time for implementation
Connect to Discord webhooks	IT06	IT03 need to be satisfied for the following test to be carried. Upon human detection, the date time of the intrusion and captured image is sent to the discord for processing using a webhook.	Successfully connected to the defined server and received the picture and time of intrusion without delay.
Continue patrolling following no human or human intrusion	IT07	IT03 need to be satisfied for the following test to be carried. Following a human/non-human intrusion, AlarmBuzz shall continue to patrol until the predefined time.	Successfully continue to patrol the area until the predefined time, some form of delays detected when the main.py end while the human detection script is being run.

System Testing

System Test Cases			
Test Name	ID	Requirements	Results
Patrolling without any motion	ST01	IT01, IT02, IT03, IT07 are mandatory to be satisfied for the following test to be carried. AlarmBuzz will patrol an area with any hot-blooded creature intrusion	Passed. It continues to patrol without detecting any motion.
Patrolling, motion detected and it is a human intrusion.	ST02	IT01, IT02, IT03, IT04, IT06, IT07 are mandatory to be satisfied for the following test to be carried. Patrolling then the PIR sensors detect a motion and the Pi camera classify the intrusion as human and send a discord bot message.	Passed. Patrolling then the PIR sensors detect a motion and the Pi camera classify the intrusion as human and send a discord bot message.
Patrolling, motion detected and it is not a human intrusion	ST03	IT01, IT02, IT03, IT04, IT06, IT07 are mandatory to be satisfied for the following test to be carried. Patrolling then the PIR sensors detect a motion and the Pi camera classify the intrusion as non-human and continue to patrol the area for 20 second sweeping the area	Passed but have a 20% chance of detecting ghost human.
Patrolling, motion detected, human intrusion and continue to patrol until X sec execution.	ST04	IT01, IT02, IT03, IT04, IT06, IT07 are mandatory to be satisfied for the following test to be carried. Patrolling then the PIR sensors detect a motion and the Pi camera classify the intrusion as human and send a discord bot message. After X seconds, AlarmBuzz stops	Passed. Patrolling then the PIR sensors detect a motion and the Pi camera classify the intrusion as human and send a discord bot message. After X seconds, AlarmBuzz stops

CONCLUSION

Difficulties faced

- Poor power supply for the raspberry pi 5v == 2A was not enough.
- Pi camera v2 extension cable was too short, had to buy a new one with an additional pin to be cut off.
- Lack of time to implement Arduino IoT cloud fully

Future Implementations

- Improve AI and machine learning for human/animal detection
- Enhance Object Collision Avoidance System for larger robot size
- Implement live feed with better servo motor and camera calibration
- Address mobility and stability weaknesses for outdoor use and harsh terrain
- Develop a more robust and weather-resistant design for various environments
- Incorporate solar panels for environmentally-friendly power source
- Create a security network with multiple AlarmBuzz units for continuous monitoring
- Establish a control center for coordinated patrols and PIR sensor integration
- Add robotic arms for intervention and collaboration with law enforcement

QUESTION ???

REFERENCE LIST

1. Ardumotive Arduino Greek Playground. (n.d.). How to use a passive buzzer module with Arduino. [online] Available at: https://www.ardumotive.com/how-to-use-a-passive-buzzer-module-en.html [Accessed: 2 Nov. 2022].
2. Call Centre Helper. (n.d.). What Is Robotic Process Automation? [online] Available at: https://www.callcentrehelper.com/robotic-process-automation-2-153272.htm [Accessed 8 Nov. 2022].
3. Canonical (2023). [online] Ubuntu.com. Available at: https://design.ubuntu.com/brand.
4. Denver Criminal Defense Attorney | Criminal Lawyer Denver, Colorado. (2021). Burglary vs Robbery: What's the Difference? - Weeden Law. [online] Available at: https://www.weedenlaw.com/burglary-vs-robbery-whats-the-difference/#:~:text=According%20to%20FBI%20crime%20statistics [Accessed 3 Apr. 2023].
5. Discord. (n.d.). Discord's Branding Guidelines. [online] Available at: https://discord.com/branding.
6. GitHub. (n.d.). Arduino. [online] Available at: https://github.com/arduino/ [Accessed 3 Apr. 2023].
7. GitHub. (n.d.). OpenCVLogo. [online] Available at: https://github.com/opencv/opencv/wiki/OpenCVLogo [Accessed 3 Apr. 2023].
8. grobotronics.com. (n.d.). PIR Sensor Module - HC-SR501. [online] Available at: https://grobotronics.com/pir-sensor-module.html [Accessed: 17 Oct. 2022].
9. grobotronics.com. (n.d.). Raspberry Pi Camera Module V2 (8MP,1080p). [online] Available at: https://grobotronics.com/raspberry-pi-camera-module-v2-8mp-1080p.html [Accessed: 17 Oct. 2022].
10. grobotronics.com. (n.d.). Sensor Shield V5.0 for Arduino. [online] Available at: https://grobotronics.com/sensor-shield-v5.0-for-arduino.html?sl=en [Accessed 3 Apr. 2023].
11. Homan, R. (n.d.). Burglary Statistics 2022. [online] Bankrate. Available at: https://www.bankrate.com/insurance/homeowners-insurance/house-burglary-statistics/#stats. [Accessed: 15 Oct. 2022]
12. NBC News. (n.d.). Security robots expand across U.S., with few tangible results. [online] Available at: https://www.nbcnews.com/business/business-news/security-robots-expand-across-u-s-few-tangible-results-n1272421. [Accessed: 1 Nov. 2022]
13. www.ojp.gov. (n.d.). Impact of Burglary Upon Victims | Office of Justice Programs. [online] Available at: https://www.ojp.gov/ncjrs/virtual-library/abstracts/impact-burglary-upon-victims. [Accessed: 1 April. 2022]
14. Python.org. (2019). The Python Logo. [online] Available at: https://www.python.org/community/logos/.
15. Santhana_krishnanMore (n.d.). SERVO MOTOR TEST. [online] Instructables. Available at: https://www.instructables.com/SERVO-MOTOR-TEST/. [Accessed: 6 Nov. 2022]
16. Sparkfun.com. (2017). Ultrasonic Distance Sensor - HC-SR04 - SEN-15569 - SparkFun Electronics. [online] Available at: https://www.sparkfun.com/products/15569. [Accessed: 17 Oct. 2022]
17. www.securityinfowatch.com. (n.d.). StackPath. [online] Available at: https://www.securityinfowatch.com/perimeter-security/robotics/product/12390271/segway-nimbo-security-robot-from-segway-and-turing-video [Accessed: 15 Oct. 2022].