Automated Fan for Smart Home

RME40002 – MECHATRONICS SYSTEM DESIGN

GROUP 8

ABDULSWAMAD Rama Salim (101229220)

ERANTHA Budika Sugathadasa Amaratunga (101231814)

Table of Contents

1. Introduction....5
1.1 Problem Statement....5
1.2 Scope and Objectives....5
1.3 Constraint and Limitation....5
2. Literature Review....6
2.1 The Following Fan....6
2.2 Air Sketcher Fan....6
2.3 Research on Intelligent Embedded Fan System Based on YOLOv2 Lightweight Algorithm.. 7
2.4 Real-time Monitoring of Fan Operation in Livestock Houses Based on the Image Processing7
3 Project Management and Finance....8
3.1 Job Distribution....8
Mechanical Fabrication, Design, and Assembly....8
Programming and Circuit Design....8
Circuit Fabrication, System Integration, and Testing....8
3.2 Project Milestone....9
3.3 Finance....10
4 Design, Method, and Procedure....11
4.1 Mechanical Design....11
Choosing the Servo Motors....15
4.2 Circuit Design and Component Sizing....15
Final Circuit Design....15
Components Specifications....17
2.1 Logic and Control....21
4.2.2 Real-time Human Detection....21
4.2.3 Buffering Mechanism....23
4.2.4 Communication between Arduino and Python code(computer)....25
4.2.5 Arduino Code....25
2.1 Results and Discussions....28
5 Documentation....28

5.1 Design Documentation....28
5.2 Code Listings and Software Documentation....28
5.3 Testing Protocols....28
5.4 User Manual....29
5.5 Project Report and Presentation Materials....29
6 Ethics....29
6.1 Ethical Issues....29
6.2 Implementation of Solutions....30
7 Conclusion....30
8 References....31
9 Appendix....32
9.1 Mechanical Design Drawings....32
9.2 Code Listings....34
Python code....34
Arduino code....35

Table of Figures

Figure 1: First Iteration of the Automated Fan 11
Figure 2: Second Iteration of the Automated Fan 12
Figure 3: Final Iteration of the Automated Fan 13
Figure 4: Labelled Components 14
Figure 5: Final Product 14
Figure 6: Smart Home Fan Circuit Schematic 16
Figure 7: Arduino UNO R3 17
Figure 8: 5V DC Fan and Manufacturer Specifications 18
Figure 9: Logitech C920 Webcam 19
Figure 10: DC12V Power Supply Unit 19
Figure 11 : Pan & Tilt Bracket Mounts 20
Figure 12: Plywood 21
Figure 13: Flowchart for Python Human detection code 22
Figure 14: Flowchart for Arduino Program for Servo and Fan Speed Control 27
Figure 15: Top and Bottom Sides of Casing 32

Figure 16: Front and Back Side of Casing 32
Figure 17: Right and Left Side of Casing 33
Figure 18: Servo Motors with Brackets (Pan and Tilt Setup) 33
Figure 19: Brackets to Attach Servo Motors to Case 34

1. Introduction

1.1 Problem Statement

Optimizing an Automated Fan's ability to dynamically modify rotation and speed in response to individual and collective human presence is difficult when integrating it with camera-based tracking technologies in smart homes. The main challenge is creating an intelligent system that can recognize and follow people, synchronize rotational changes based on their motions, and control fan speed based on the number of people in a given space. These features should be developed to strike the best possible balance between user experience, energy efficiency, and comfort.

1.2 Scope and Objectives

This report's scope includes a thorough examination of the Automated Fan intended for smart homes, particularly emphasizing how it incorporates camera-based tracking technology to modify rotation and control fan speed in response to individual and group human presence. It examines the system's operation, the technological foundation, and the effects of implementing it in a smart home setting.

The objectives of this assignment include:

- Design of Fan: Create different iterations of a desk fan which can be utilized.
- Technology Analysis: Analyze the underlying camera-based tracking system that allows the Automated Fan to track and identify people.
- Functional Evaluation: Examine the fan's capacity to dynamically modify rotation in response to individual movements and lower fan speed in response to the number of occupants detected locations.

1.3 Constraint and Limitation

The Constraints and Limitations of the project are as follows:

- Budgetary Restrictions: Limited financial resources for purchasing high-end equipment or advanced technologies for prototyping and experimentation.
- Resource Availability: Limited access to specialized tools, hardware, or software required for prototyping and testing the Automated Fan's functionalities.
- Technical Expertise: Constraints in the student's technical knowledge or skill set regarding sophisticated camera-based tracking systems and motor control mechanisms.

2. Literature Review

2.1 The Following Fan

The Following Fan is a pioneering system with patented technology that seamlessly tracks users using a built-in camera and sophisticated algorithms. This innovative design ensures that the fan automatically adjusts its position to direct a cooling airflow toward the individual. Not limited to just personalized cooling, this advanced fan also provides oscillation and stationary modes, catering to various scenarios—perfect for applications beyond person-tracking, like efficiently cooling car engines in automotive workshops.

Apart from its exceptional cooling abilities, the Following Fan prioritizes energy efficiency. Once the tracked individual exits its 135° range, the fan scans for the next closest person. If no one is detected, it smartly goes into a low-power state, conserving energy (Smart Sleeping). Continuously sensing its surroundings, the fan reactivates upon detecting someone, ensuring it operates only when necessary, eradicating waste and optimizing its efficiency.

2.2 Air Sketcher Fan

The groundbreaking innovative fan technology showcased by the Air Sketcher fan, with its remarkable ability to track user movements and precisely direct airflow, introduces a disruptive wave of innovation across a spectrum of industries. Notably, this advancement revolutionizes traditional fan functionality within the cooling sector, paving the way for a new era of personalized cooling experiences. Beyond altering the cooling industry landscape, this technology extends its reach into diverse sectors like fitness, healthcare, and gaming. In fitness, the Air Sketcher fan's body-tracking capabilities offer prospects for innovative applications, potentially revolutionizing workout experiences. Similarly, in healthcare, this technology holds promise for tailored cooling solutions for patients, enhancing comfort and personalizing care. Moreover, the adaptability of the Air Sketcher fan to cater to individual preferences sets a new standard, signalling significant shifts in the consumer electronics industry toward more personalized and user-centric solutions. Overall, this technological leap transforms cooling and ignites possibilities for disruptive innovation across multiple sectors, offering a glimpse into a more tailored and efficient future.

2.3 Research on Intelligent Embedded Fan System Based on YOLOv2 Lightweight Algorithm

Advancements in artificial intelligence have spurred research into applying intelligent algorithms to low-power embedded chips, presenting a contemporary area of exploration. This study focuses on optimizing the YOLOv2 algorithm, tailoring it for deployment on the K210 chip specifically to train a separate model for face object detection. Implementing this optimized YOLOv2 model on the K210 chip empowers the intelligent fan to detect and precisely locate characters within machine coordinates. Utilizing this information on character position and size, the fan intelligently adjusts its rotation angle and airflow volume. The experimental outcomes showcase this novel method as an innovative approach in embedded chip intelligence, enhancing the YOLOv2 algorithm and introducing unique tracking and ranging algorithms. These innovations significantly improve human perception in solo and crowd tracking, enhance air delivery accuracy and user comfort, and offer theoretical and practical foundations for integrating AI and embedded systems in diverse fields.

2.4 Real-time Monitoring of Fan Operation in Livestock Houses Based on the Image Processing

This study pioneers a real-time monitoring method that combines image processing and mathematical modelling to oversee exhaust fan operations, ensuring optimal airflow in a mechanically ventilated animal housing system. By analyzing visual data in real-time, this approach continuously supervises the running status of fans, guaranteeing they function efficiently to maintain proper ventilation. This method's innovation lies in its ability to swiftly detect and address any irregularities in fan operation, proactively ensuring a healthy environment for the housed animals. Integrating these technologies offers a dynamic solution to regulate and maintain optimal ventilation conditions, prioritizing animal welfare and productivity within the facility. Videos and images capturing the operations of exhaust fans within a livestock house underwent meticulous examination across a spectrum of operating frequencies, including 15 Hz, 25 Hz, 35 Hz, and 45 Hz. Through a systematic process, the individual pixels present in these images were isolated and translated into a defined coordinate system delineated by the X and Y axes. Employing advanced techniques, such as the Hough transformation, facilitated precisely identifying specific areas of interest within these images. Once these target areas were identified, a sophisticated method known as the dense optical flow technique was applied. This method enabled the calculation and assessment of pixel displacements occurring within these targeted zones, offering detailed insights into the dynamic movements within the captured imagery.

3 Project Management and Finance

3.1 Job Distribution

This project can be divided into three components. The first component corresponds to the Mechanical Design section of the project, which entails the Fabrication, Design and Assembly of the Automated Desk Fan. The second component corresponds to the Programming and Circuit Design. The final component consists of Circuit Fabrication, System Integration and Testing.

Mechanical Fabrication, Design, and Assembly

Fabrication: Erantha is responsible for physically constructing the mechanical components of the Automated Fan system. This involves building the fan structure, housing, and any physical parts required for the fan's operation.

Design: Erantha oversees the design phase, conceptualizing the mechanical structure and ensuring it meets the required specifications. This includes creating detailed plans, CAD models, and blueprints for the fan's mechanical components.

Assembly: Erantha assembles all the mechanical parts, ensuring they fit together correctly and function as intended. This involves assembling the fan's physical components based on the design specifications.

Programming and Circuit Design

Programming: Rama is responsible for developing the software and programming required for the Automated Fan system. This includes writing code for motion tracking, speed control, and other software components necessary for the system's operation.

Circuit Design: Rama handles the design of the electronic circuits needed for the fan system, including the layout, connections, and functionality of the electrical components.

Circuit Fabrication, System Integration, and Testing

Circuit Fabrication: Rama and Erantha collaborate on fabricating the electronic circuits designed by Rama. This involves physically creating the circuits, soldering components, and ensuring they function correctly.

System Integration: Together, Rama and Erantha work on integrating the mechanical and electronic components of the Automated Fan system. This includes connecting the circuits to the mechanical structure and ensuring proper alignment and compatibility.

Testing: Both team members collaborate on testing the entire system. They verify that the mechanical and electronic parts work together seamlessly, conduct performance tests, and troubleshoot any issues that arise during the testing phase.

This division of responsibilities ensures that each team member specializes in their area of expertise while collaborating on the critical phases of circuit fabrication, system integration, and testing to ensure a cohesive and functional Automated Fan system.

3.2 Project Milestone

To make sure that the project progresses appropriately, a project milestone Gantt Chart has been created. In between weeks 5, 6 and 7, the progress was a bit slower than expected since we were still working on the justification of most of the electrical components and had partially ordered some of the components and they ideally take time to arrive.

Work Activity	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12
Form Group & Finalise Topic												
Literature Review												
Components Survey												
Mechanical Design & Calculations												
Circuit Design & Component Sizing												
Logic and Control Design												
Mechanical Fabrication												
Circuit Fabrication												
Mechanical Assembly												
System Integration												
Programming												
Testing of Fan												

Table 1: Gantt Chart of Project Timeline

3.3 Finance

The following table is the list of items purchased for the project.

Component	Price (MYR)	Quantity	Total Cost (MYR)
Arduino UNO	37.81	1	37.81
Fan (5V DC)	43	1	43
Servo Motor (MG995)	16.9	3	50.7
Power Supply Unit AC220V to DC12V	13.58	1	13.58
Motor Bracket Mounts	70	1	70
Cytron Protoboard	14	1	14
7805 Voltage Regulator	1.79	3	5.37
Header 2.54mm Male Connector 40p	0.5	3	1.5
KF301 Terminal Screw Connector 2P	0.39	6	2.34
IN4007 Diode	0.99	4	3.96
M3 * 30mm Screw	1.49	30	44.7
IC L293D Motor Driver	15	3	45
A4 Size 9mm Plywood Custom Size	40	1	40
Terminal Screw Connector PCB 3P	0.59	1	0.59
Servo Mounting	4.49	1	4.49
Aluminium 25T Round Servo Holder MG995	3	2	6
M3 * 10mm Screw	0.13	12	1.56
M4 * 8mm	0.12	20	2.4
M4 * 16mm	0.19	9	1.71
M8 * 10mm Screw	0.5	5	2.5
M8 Nuts	0.5	5	2.5
Camera Mount Screw	0.4	1	0.4
Eveready Super Heavy Duty Batteries AAA	7.9	3	23.7
Construction Adhesive	5.9	1	5.9
TOTAL			**423.71 RM**

Table 2: Bill of Materials for Project

4 Design, Method, and Procedure

4.1 Mechanical Design

The mechanical design of the fan has gone through multiple iterations; in each iteration, changes were made to improve the fan's design. The first iteration is shown below.

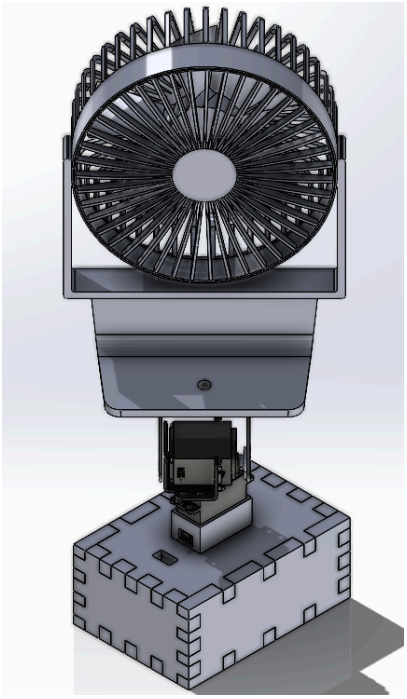

Figure 1: First Iteration of the Automated Fan

In the initial design, the group created a miniature box casing to house the components. A holder was placed directly on top of the box to hold the first servo motor in place. However, although this may have been simple to create on Solidworks, replicating this would have been tedious as glue and small pieces of plywood would be required to create the holder. When considering that glue would be used to fix the holder to the main casing, it must be considered that glue might not have been strong enough to hold the holder in place while the fan was rotating. Along with this came the realization that the casing was too small and might topple over as the fan tilted. This led to the creation of the second iteration.

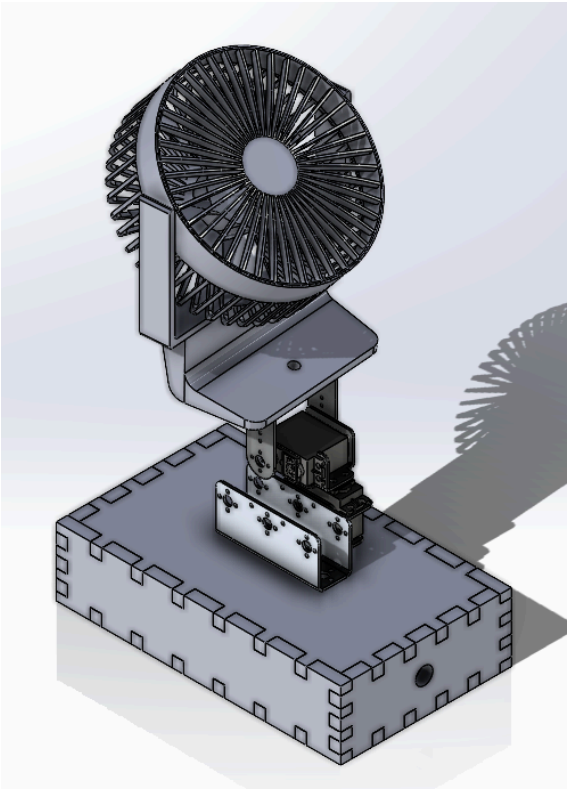

Figure 2: Second Iteration of the Automated Fan

In the second iteration, the redesign included a larger box for stability and a bracket, which was screwed into the plywood casing used to attach the servo motor. Upon the creation of the second iteration, the team realized that there was no hole at the top of the casing for all the wires to enter; due to this, there would be a cluster of wires outside the casing, and that would not be sufficient for a product to be sold in the marketplace. Along with this, the team realized that the casing was also too small to house all the electrical components, thus leading the third iteration to have a larger casing for all the components and for more improved stability.

The final iteration is shown below:

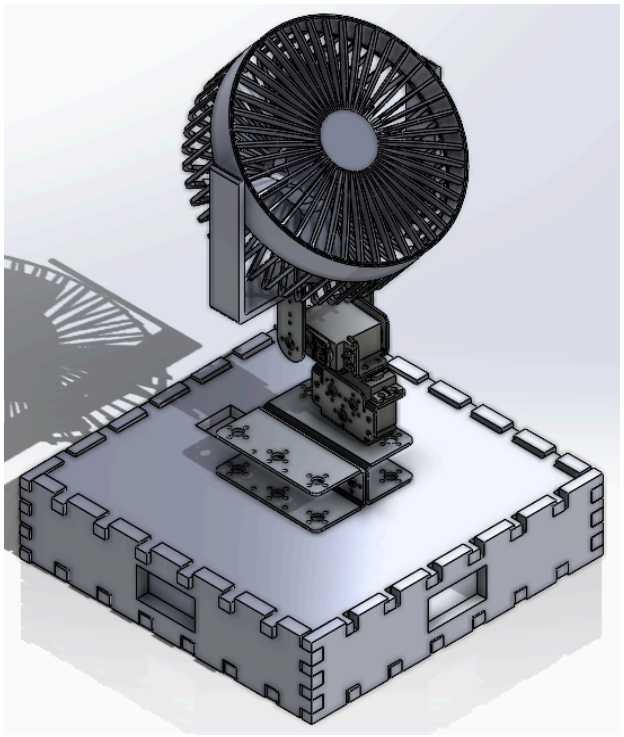

Figure 3: Final Iteration of the Automated Fan

This iteration consists of a far larger casing, containing a larger opening on each end and top of the casing. These openings are meant for cable management, reducing the cable clutter that arose from the previous iterations. This iteration also consists of two brackets attached, which were, in turn, attached to the top end of the casing; this would improve the stability of the fan as it tilted and rotated. Using a different type of bracket, the servo motor is attached to the base bracket. The bottom servo motor is responsible for the fan's rotation, while the top servo motor tilts the fan.

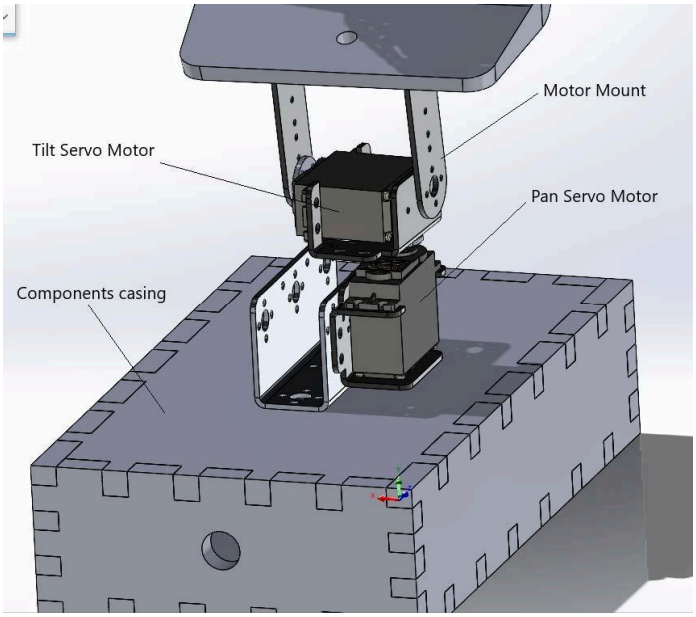

Figure 4: Labelled Components

Found below is the product, with a camera pasted to the front of the casing for motion tracking.

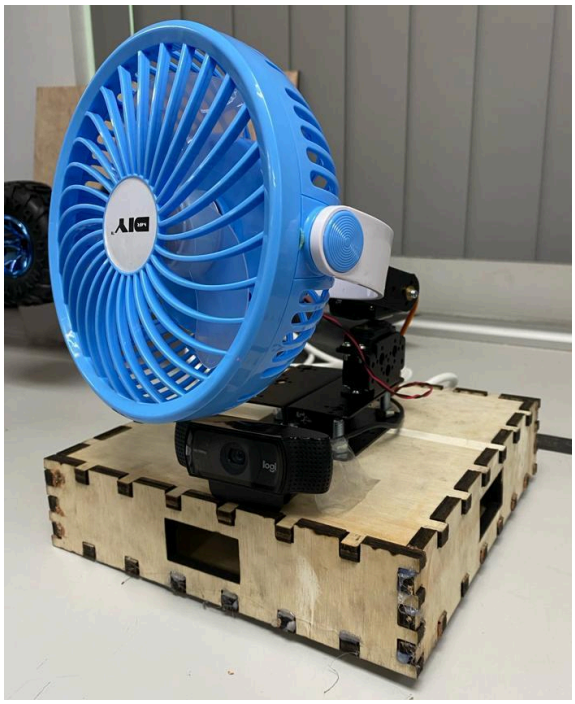

Figure 5: Final Product

Choosing the Servo Motors

For the tilt servo:

$Torque_{tilt}$ = Weight of the fan * g * (distance to the center of fan from tilt servo's rotational axis)

Given:

Weight of the fan = 0.365 kg
Distance = 5.5 cm (0.055m)
g = 9.81 m/s2
$Torque_{tilt}$ = 0.365 kg × 9.81 m/s2 × 0.055 m × sin (15) = 0.05097 kg-m (5.097kg-cm)

For the Pan servo:

Instead of using gravitational load, consider rotational inertia. The rotational inertial for a point mass *m* at a radius *r* from the axis of rotation is $I = m \times r^2$. Therefore, the torque needed to accelerate this inertia is $\tau = I \times \alpha$, where α is the angular acceleration.

Assume an angular acceleration of 5 rad/s^2.

$$Total\ Rotational\ Inertia = I_{Tilt\ Servo} + I_{Mount} + I_{Fan}$$

$$Rotational\ Inertia = \left(\left(0.055kg \times (0.054m)^2\right) + \left(0.036kg \times (0.087m)^2\right) + \left(0.365kg \times (0.087m)^2\right)\right)$$

$$Total\ Rotational\ Inertia = 0.003195549\ kgm^2$$

Therefore, the torque needed to achieve an angular acceleration of assuming 10 rad/s^2 is:

$$\tau = I \times \alpha = 0.003195549 \times 10\ rad/s^2 = 0.0319\ Nm \approx 0.325\ kgcm$$

Given that the MG995 servo motor has a maximum torque of 10 kg-cm, it can handle the torque requirements for the Pan motor. As for the tilt motor, which accounts for gravity, the torque comes to around 5.097, which is within the range of our stall torque. Remember that we have not accounted for mechanical inefficiencies and unexpected loads, which will become evident when we physically assemble and test the system.

4.2 Circuit Design and Component Sizing

Final Circuit Design

The following illustration shows the components that would be used to create the smart home fan circuit as well as how they are connected in the circuit. Below is the full schematic of our circuit design:

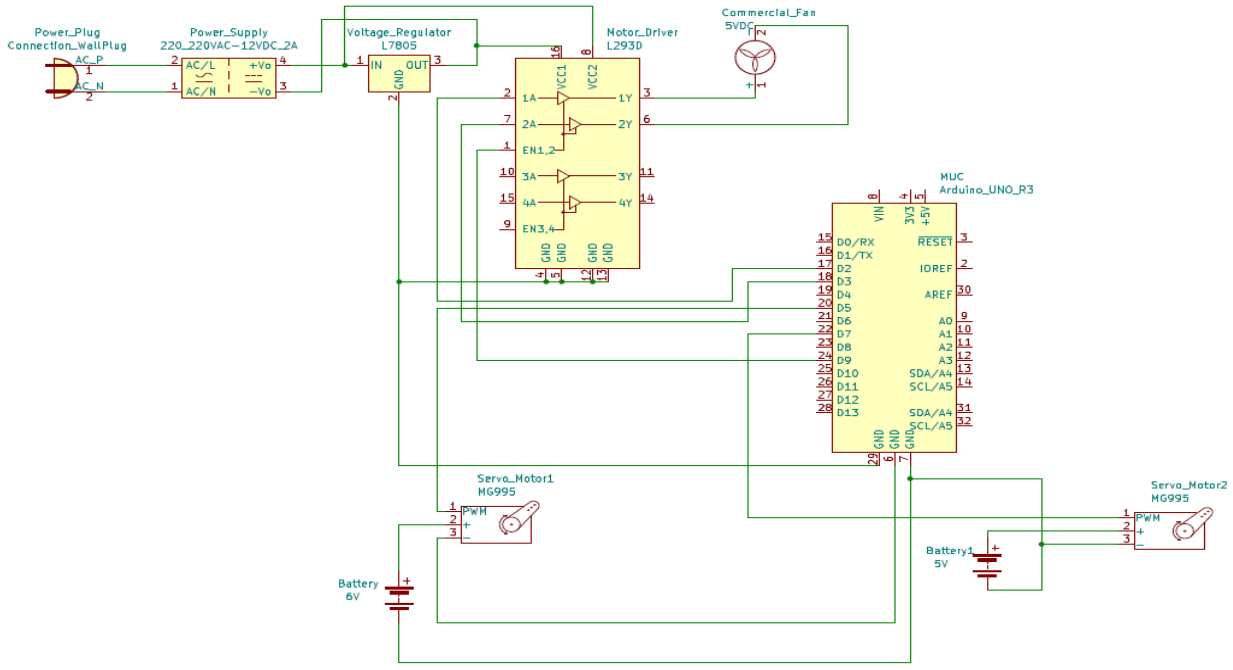

Figure 6: Smart Home Fan Circuit Schematic

COMPONENT	**REASON**
Arduino UNO	Controlling of tasks and versatile I/O capabilities
Fan Motor DC	Suitable for desired fan speed and power requirements
Webcam	High resolution imaging for human detection and tracking in project

L293 IC	Controls 5V DC fan motor
Servo Motor MG995	Provides necessary torque and range for fan's pan and tilt mechanism
Power Supply 12V 2A	Power the fan, with voltage regulation
Motor Bracket Mounts	For mounting the servo motors
Plywood	Used for base and housing of electrical components
Prototyping Shield	Integration and soldering of circuit components.
Diodes	Prevent against voltage spikes and back EMF from DC motor.

Table 3: Main Components Used

Components Specifications

4.2.1.1 Microcontroller

For this project, we chose the Arduino R3 Microcontroller board. This board contains a detachable ATmega328 microcontroller, which will run the software written by the Arduino IDE. Arduino comprises 14-digit I/O pins, 6 of which can be utilized as PWM outputs. This will allow us to control the fan's speed and two servo motors using PWM. The Arduino also has a flash memory of 32KB and 0.5KB, an SRAM of 2KB, and an EEPROM with 1KB of memory, which makes it suitable for the microcontroller application for controlling the different actuators.

Other peripherals, hardware, and components connected to the microcontroller in the Arduino board include voltage regulators (the Arduino runs at an operating voltage of 5Vdc), Atmel 16U2 (this chip manages the USB connection and removes the need for an external programmer), 16MHz quartz crystal (provides clock signal for the microcontroller), female header pins (gain access to the pins on the microcontroller), ICSP header (another method available to program the Arduino), Type B USB receptacle (provide power and data), DC power jack

(provide power), capacitors, and resistors. It also offers flexibility since it is compatible with multiple hardware, such as the motor driver shield later used in the water level control project and software tools, such as Arduino IDE, which allowed us to program the board online and offline.

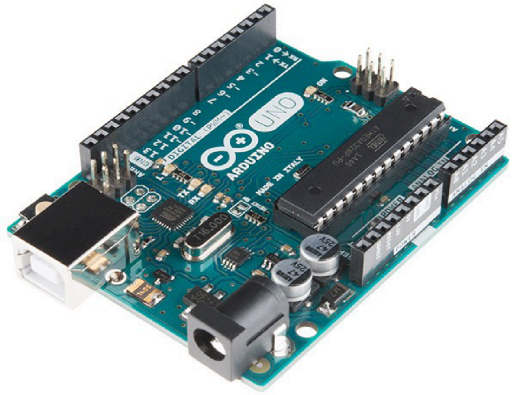

Figure 7: Arduino UNO R3

4.2.1.2 Commercial Fan 5V DC Motor

A commercial DC motor-based fan was chosen for the fan due to its compact size, power efficiency, and ease of control via PWM signals. The motor is rated for 5V, making it compatible with commonly available power supplies and motor drivers. Additionally, it provides adequate airflow for our application. The fan also comes with Li-ion(18650) rechargeable batteries from the manufacturer, but we had to replace these batteries since we are using a 12V power supply to power the fan.

The fan to be used in the project is shown below:

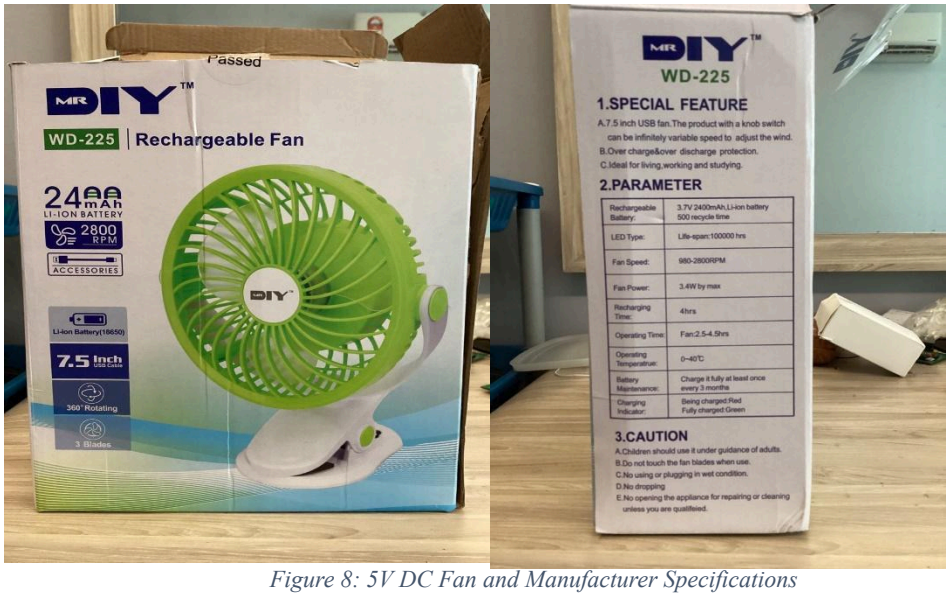

Figure 8: 5V DC Fan and Manufacturer Specifications

4.2.1.3 Logitech Webcam

We chose a computer-based Logitech webcam for its reliability, high-resolution capabilities, and compatibility with OpenCV for image processing tasks. Its widespread use in various computer vision applications ensures robust community support and a wealth of available resources. The Logitech webcam has a maximum resolution of 1080p / 30fps – 720p / 30fps, which is suitable for computer vision, especially human detection. Our camera's diagonal field of view (dFoV) is 78 degrees, which is enough and fits a sizeable room if the camera is placed at a further location. The camera is also tripod-ready and allows universal mounting clip fits, making it easily mountable on our system. The Webcam used is shown below.

Figure 9: Logitech C920 Webcam

4.2.1.4 12 V 2A Power supply

The 12V 2A power supply is used to power up the fan. The fan motor is designed to operate at 5V; thus, we will use a voltage regulator in our motor driver circuit to step down the voltage to 5V. Despite being equipped

initially with Li-ion (18650) rechargeable batteries, we opted for a 12V power supply due to the motor's underperformance when powered by 5V. Also, we had previously considered using a 5V power supply. However, the problem is that the 5V power supply going to the motor driver gets stepped down drastically to lower than 5V, thus making our fan rotate very slowly. By using a 12V 2A power supply and stepping down the voltage to 5V through the L293D motor driver, we achieved optimal fan speed and functionality.

Figure 10: DC12V Power Supply Unit

4.2.1.5 Diodes

Used to reinforce the internal diode protection of the L293D. These external diodes (IN4007) safeguard against voltage spikes and back EMF (Electromotive Force) generated by the DC motors. This is particularly important in scenarios where the motor stops abruptly or changes direction, which can generate significant voltage spikes capable of damaging the motor driver or other sensitive components in the circuit. By implementing external diodes, we ensure an extra layer of protection, enhancing the longevity and safety of our system.

4.2.1.6 Bracket Mounts

The bracket mounts used in this project were based on their compatibility with the MG995 servo motor, which has a 180-degree range of motion. The mounting multi-function bracket and long U-shaped bracket, made of 2mm of high-quality Aluminium, offer excellent structural support and are lightweight. Their dimensions and weight make them suitable for use with our selected servo motors and fan without adding significant load. The sandblasting oxidation finish adds durability to the components.

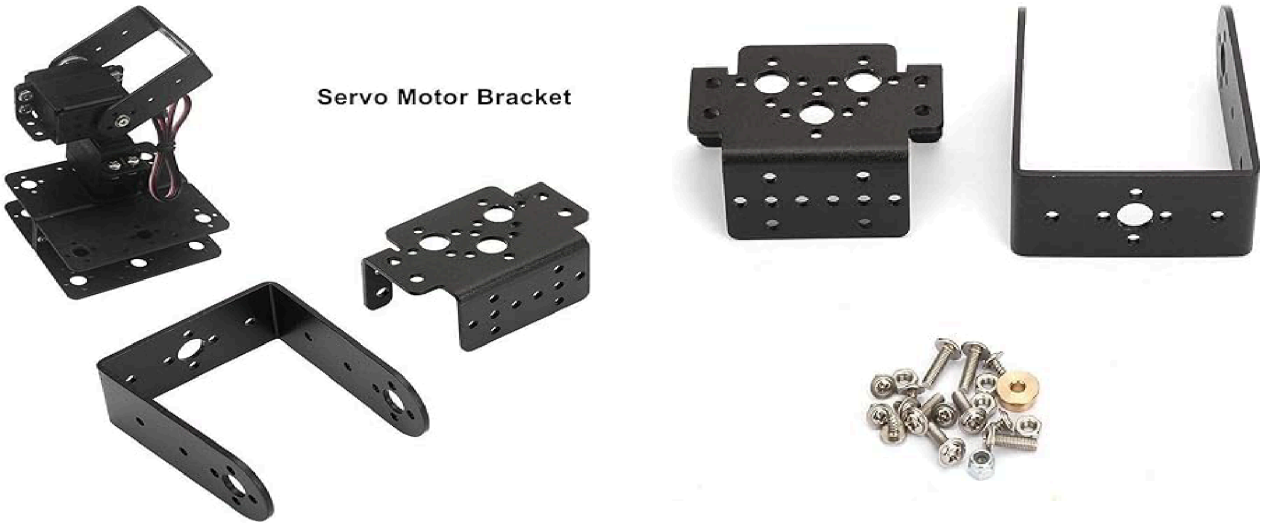

Figure 11 : Pan & Tilt Bracket Mounts

4.2.1.7 Plywood

In constructing the base and housing for our fan system, we chose plywood due to its versatility, strength, and ease of fabrication. Plywood is an ideal material for this application for several reasons:

- Durability and Stability: Plywood offers high strength and stiffness per unit weight.
- Ease of Modification: Plywood can be easily laser-cut, drilled, and shaped to fit our project's specific dimensions and design requirements.
- Cost-Effectiveness: Plywood is cheaper than other materials like Aluminium, making it suitable for our budget constraints.

Figure 12: Plywood

2.1 Logic and Control

4.2.2 Real-time Human Detection

This section details the design, method and procedure of a human presence detection and response system. The system utilizes computer vision techniques for detection and microcontroller-based hardware for response execution. The Python script serves as the backbone of the detection system.

```python
import cv2
import serial
import time
import numpy as np

# Initialize the serial connection to Arduino
arduino = serial.Serial('/dev/ttyACM0', 9600)
time.sleep(2)
```

It starts with importing necessary libraries, such as OpenCV for computer vision tasks, and establishes a serial communication link with the Arduino board. The script then configures the video capture object, setting the frame resolution and rate to optimize the balance between image quality and processing speed. Below is the flowchart of the Python object detection code:

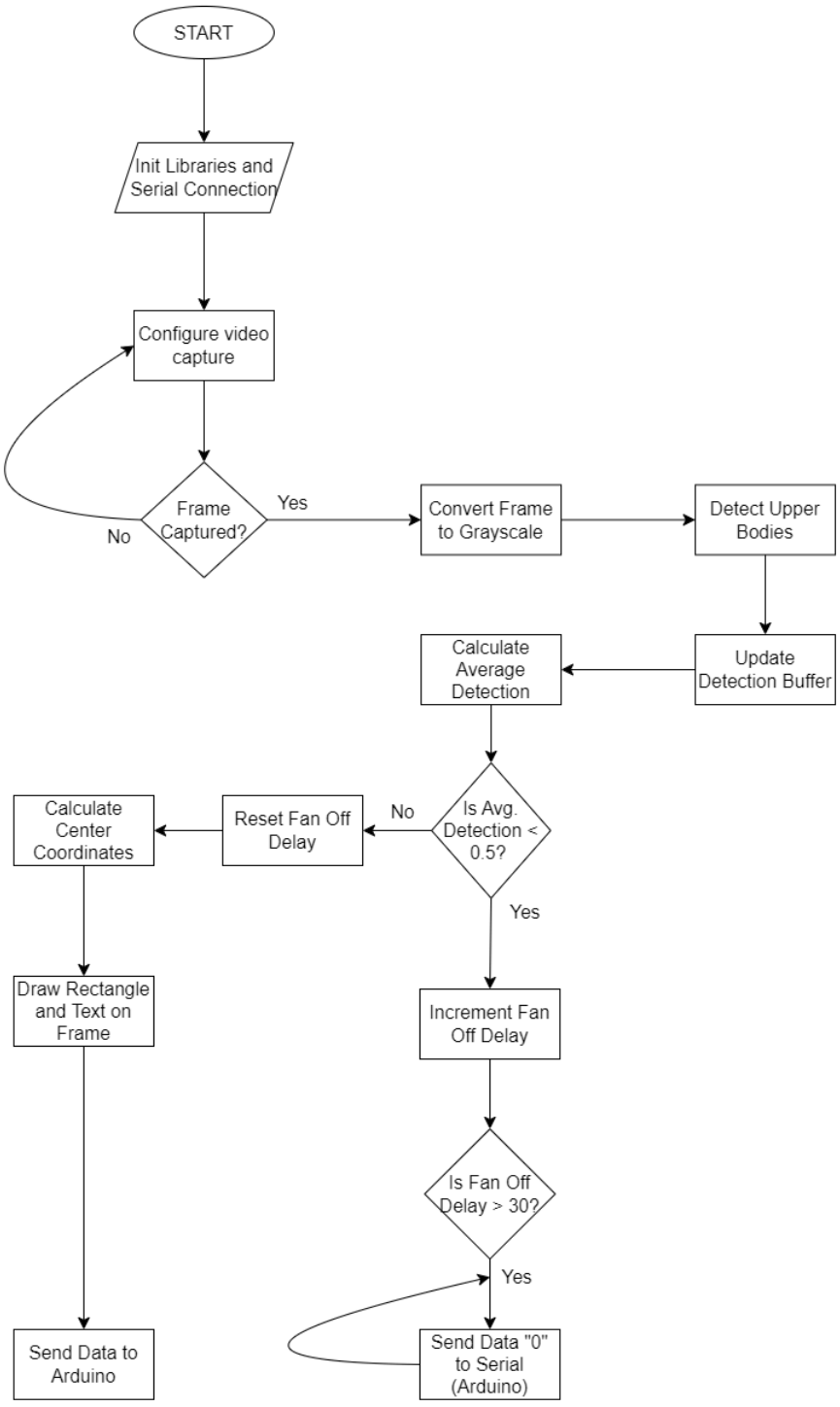

Figure 13: Flowchart for Python Human detection code

The video capture settings are set so as to capture the video in 1920 * 1080 resolution with the MJPEG video format. This format is widely used for computer vision and is the most suitable for human detection.

```
# Start capturing video
cap = cv2.VideoCapture(2)

# cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
# cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

W, H = 1920, 1080
cap.set(cv2.CAP_PROP_FRAME_WIDTH, W)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, H)
cap.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc('M', 'J', 'P', 'G'))
cap.set(cv2.CAP_PROP_FPS, value: 30)
```

A continuous loop captures video frames in real-time while each frame undergoes preprocessing, where it is converted to grayscale. This is a standard practice in computer vision to reduce the computational load. The algorithms used are Haar Cascades, specifically the upper body detection since the fan is placed on top of the table and thus the camera is focused is focused on the upper body of the human.

A critical aspect of the script is the buffering technique used to stabilize detection results. This technique employs a buffer that stores the detection results of the last several frames, smoothing out short-term fluctuations. Such an approach is vital in reducing false positives and ensuring the system's reliability. The script then sends the processed detection results to the Arduino via serial communication, formatted for easy parsing and execution.

4.2.3 Buffering Mechanism

Because of momentary detection errors and environmental noise, buffering is used to stabilize the detection of human presence. It is designed to average the detection results over a sequence of video frames thus reducing the errors. Here, a buffer array of size 30 is initialized with all elements set to 0. This array will hold the count of human detections for the last 30 frames.

```
while True:
    _, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


    upperbodies = upperbody_cascade.detectMultiScale(gray, scaleFactor: 1.1, minNeighbors: 3)
    all_detections = list(upperbodies)

    human_count = len(all_detections)
    detection_buffer[buffer_index] = human_count
    buffer_index = (buffer_index + 1) % len(detection_buffer)

    avg_detection = sum(detection_buffer) / len(detection_buffer)

    if avg_detection < 0.5:  # Adjust
        fan_off_delay += 1
        if fan_off_delay > 30:  # Adjust
            arduino.write("0,0,0\n".encode())
            fan_off_delay = 0
    else:
        fan_off_delay = 0
        for (x, y, w, h) in all_detections:
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
            center_x = x + w // 2
            center_y = y + h // 2
            cv2.putText(img, text: f"({center_x}, {center_y})", org: (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, fontScale: 0.5, color: (255, 255, 255), thickness: 2)
            cv2.putText(img, text: f"({center_x}, {center_y})", org: (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, fontScale: 0.5, color: (255, 255, 255), thickness: 2)
            arduino.write(f"{center_x},{center_y},{human_count}\n".encode())
    cv2.imshow( winname: 'Detection', img)
    if cv2.waitKey(1) == ord('q'):
        break
```

For each frame, the number of detected humans (human_count) is added to the buffer at the current index (buffer_index). The buffer index is then incremented and wrapped around if it reaches the end of the buffer array. The average detection over the buffer is calculated by summing all elements in detection_buffer and dividing them by the buffer's length. Mathematically, this can be represented as:

$$average\ detection\ = \frac{\sum_{i=0}^{n} detection\ buffer[i]}{n}$$

where n is the size of the buffer

The system checks if the average detection is below a threshold (0.5). This threshold acts as a decision point for determining the presence or absence of a human. The threshold value of 0.5 implies that if, on average, less than half of the frames in the buffer have detected a human, the system will consider the room unoccupied.

4.2.4 Communication between Arduino and Python code(computer)

The Arduino was connected to the laptop through USB cable to enable UART communication. Therefore, the OpenCV can send coordinate values to the Arduino. This way, computing intensive tasks are separated and given the Arduino has very low processing power and it cannot handle being connected to the camera due to its low specifications. For the Python program, pyserial package needed to be installed first, then import serial and create serial object.

4.2.5 Arduino Code

For the microcontroller programming, the Arduino sketch illustrates the logic and decision-making process used to control the servo motor and fan based on the data received from the Python script via serial communication. This data determines the controlling of the two servo motors, for pan or tilt movement. The Arduino begins with the setup function which initializes the serial communication at a baud rate of 9600. The rate must match the rate used by the Python script to ensure seamless data transmission. The initial state of the fan is set to off, ensuring that the system starts in a neutral state.

```
void setup() {
  Serial.begin(9600);          // Start serial communication at 9600 baud rate
  panServo.attach(5);          // Attach servo on pin 5
  pinMode(enablePin, OUTPUT);  // Set fan pin as output
  pinMode(in1Pin, OUTPUT);
  pinMode(in2Pin, OUTPUT);

  digitalWrite(in1Pin, LOW);
  digitalWrite(in2Pin, HIGH);
  analogWrite(enablePin, 0);  // Set speed to 0
}
```

In the main loop, the Arduino continuously monitors the serial port for incoming data, if available, the Arduino reads the incoming string until it encounters a newline character. This string contains critical information: the X and Y center coordinates of the detected human and the human count, formatted and sent by the Python script. Parsing this string is a crucial step where the Arduino extracts the X and Y coordinates and the human count by identifying comma separators within the data.

4.2.5.1 Servo Motor Control Logic

The Arduino employs a mapping function to translate the detected X coordinate into a corresponding angle for the servo motor. This translation is vital for ensuring that the servo pans towards the direction of the detected human. The map function's role here is to linearly transform the range of detected X coordinates to a range of servo angles.

```
void loop() {
  if (Serial.available() > 0) {
    String data = Serial.readStringUntil('\n');
    int firstCommaIndex = data.indexOf(',');
    int secondCommaIndex = data.indexOf(',', firstCommaIndex + 1);

    int centerX = data.substring(0, firstCommaIndex).toInt();
    int centerY = data.substring(firstCommaIndex + 1, secondCommaIndex).toInt();
    int humanCount = data.substring(secondCommaIndex + 1).toInt();

    if (centerX == 0 && centerY == 0) {
      // No humans detected
      analogWrite(enablePin, 0);
    } else {
      // Adjust fan speed based on human count
      int fanSpeed = humanCount > 1 ? 255 : 150;

      // Control the fan
      digitalWrite(in1Pin, LOW);  // Set IN1 high to enable the fan
      analogWrite(enablePin, fanSpeed);

      // Gradual Servo Movement
      int currentPos = panServo.read();
      int newPos = map(centerX, 300, 1400, 150, 90);

      if (newPos > currentPos) {
        panServo.write(currentPos + 4);
      } else if (newPos < currentPos) {
        panServo.write(currentPos - 4);
      }

    }
  }
}
```

Mathematically, the mapping function can be expressed as follows: newPos =((currentX - minX) / (maxX - minX)) * (maxAngle - minAngle) + minAngle. In this formula, currentX is the detected X coordinate, minX and maxX represent the range of X coordinate values, and minAngle and maxAngle denote the range of angles for the servo. The Arduino then smoothly adjusts the servo's position towards the new angle with increments of 4 degrees. This increment can be increased to pan the servo much faster.

Below is the flowchart of the Arduino program:

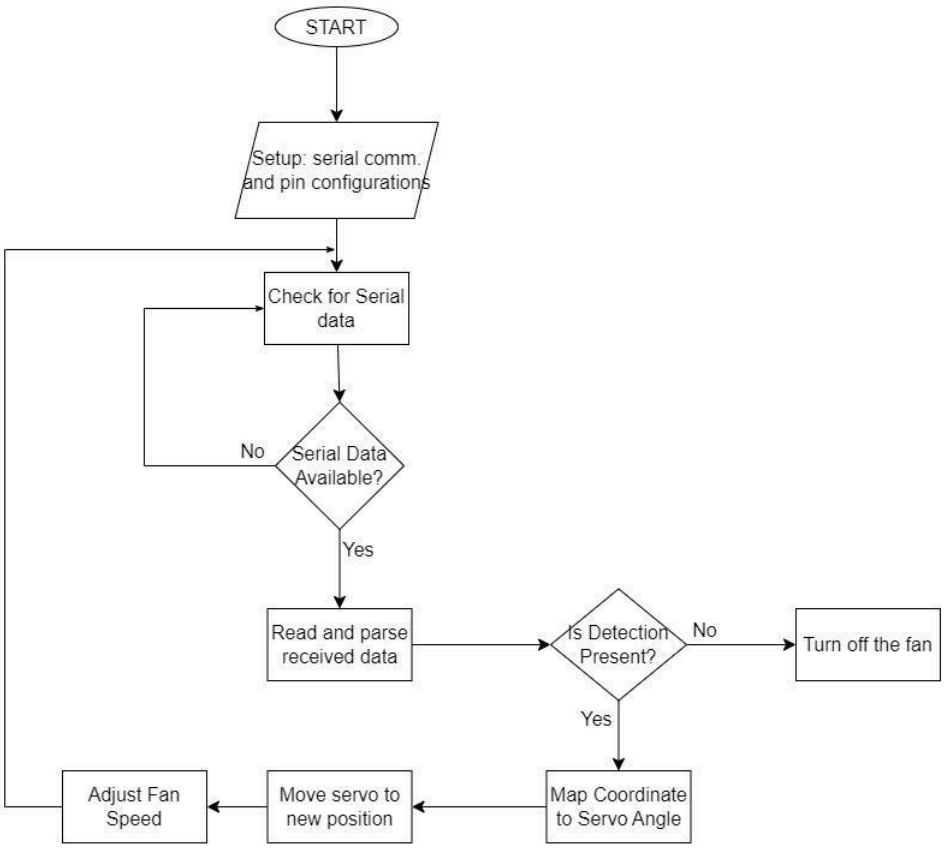

Figure 14: Flowchart for Arduino Program for Servo and Fan Speed Control

2.1 Results and Discussions

Our automated fan system for smart homes has yielded promising results. We successfully integrated mechanical components and electronic circuitry to create a robust framework for the fan system. The Python script for image processing and the Arduino sketch for motor control demonstrated commendable functionality. We addressed ethical concerns by implementing privacy measures, security protocols, and bias mitigation strategies. We encountered challenges during development, such as minor inaccuracies in detection and technical limitations. Future considerations involve refining algorithms for enhanced accuracy and exploring additional security measures for robustness. Our project has delivered a functional automated fan system while also highlighting areas for refinement and growth. We have taken a responsible approach in technology implementation by addressing ethical concerns and laying the groundwork for future advancements.

2.1 Documentation

5.1 Design Documentation

We maintained comprehensive design documentation throughout the project. This included detailed CAD drawings of the mechanical setup, electronic schematics of the circuit, and flowcharts representing the software logic. We used tools like SolidWorks for mechanical designs and KiCad for circuit schematics.

5.2 Code Listings and Software Documentation

The project's codebase is well-documented, with extensive comments for clarity. Critical segments include the Python script for image processing and the Arduino sketch for motor control. The software documentation details the implementation logic and integration steps with the hardware.

5.3 Testing Protocols

A structured approach was adopted for testing. This included unit tests for individual components and integrated system tests. We documented each test scenario, including the objectives, procedures, observed outcomes, and any necessary adjustments.

5.4 User Manual

A comprehensive user manual was compiled, providing instructions for setting up and operating the fan system. It includes safety guidelines, maintenance tips, and troubleshooting advice, ensuring that end-users can effectively utilize the product.

5.5 Project Report and Presentation Materials

The project report, alongside presentation materials, was prepared to articulate the project's journey, findings, and outcomes. These materials serve as a professional record of the project, demonstrating our methodical approach and attention to detail.

2.1 Ethics

Identifying possible ethical issues related to a project involves a critical assessment of potential areas where ethical concerns may arise. In the context of an automated fan system for smart homes:

4.3 Ethical Issues

- **Privacy Concerns:** The utilization of a camera for tracking individuals might raise privacy concerns regarding the collection and use of personal data. The use of cameras to track individuals' movements within their homes raises concerns about the collection and use of personal data. This data could be used to create detailed profiles of individuals' habits and routines, which could then be used for targeted advertising or other purposes without their full knowledge or consent.
- **Bias and Discrimination:** The system's algorithms might unintentionally exhibit bias or discriminatory behaviour based on factors such as race, gender, or age, impacting the accuracy and fairness of detection; thus leading to lack of comfort and inconvenience in using the product.

4.4 Implementation of Solutions

- **Privacy Protection:** Implementing measures such as anonymization of collected data, ensuring data encryption, and providing clear consent mechanisms for data usage can mitigate privacy concerns. .
- **Bias Mitigation:** Continuously monitor and audit algorithms for bias, employing diverse datasets during training, and incorporating fairness measures into the algorithm's design to ensure equitable outcomes.

Ethical considerations in technological projects are an ongoing process. Regular review, transparency, and proactive measures to address potential ethical issues contribute to ensuring a responsible and ethical approach in implementing and evolving technology. By carefully considering and addressing these concerns, we can ensure that the development and implementation of automated fan systems for smart homes align with ethical principles and respect the privacy and autonomy of individuals.

2.1 Conclusion

The project’s conclusion encapsulates a meticulous journey characterized by precision design, exhaustive testing, and an unwavering commitment to user-centric solutions in developing an advanced automated fan system for modern smart homes. Initiating detailed design documentation, the team harnessed the capabilities of SolidWorks and KiCad to craft mechanical blueprints and electronic schematics intricately, ensuring a robust and optimized system architecture. A paramount emphasis on code clarity and functionality materialized in meticulously documented Python scripts for image processing and Arduino sketches governing motor control. Richly annotated, these scripts underscore the team’s dedication to transparent and comprehensible coding practices. Testing protocols were executed meticulously, encompassing comprehensive unit and integrated system tests. Each scenario underwent rigorous evaluation, meticulously documented to ensure the system’s reliability across diverse operational scenarios.

5 References

- PR.COM. (2022) *Following Fan, Person-Tracking Smart Fan, Now Commercially Available*, PR.COM. Available at: https://www.pr.com/press-release/854404 (Accessed: 25 October 2023).
- Yusuf, O. (2011) *Stay Constantly Cool with the AirSketcher Fan,* TRENDHUNTER. Available at: https://www.trendhunter.com/trends/airsketcher-fan1 (Accessed: 25 October 2023).
- Wei, Y., Zhao, K., Lv, X., Feng, J., Hu, H., Badarch, T., and Zhao, Z. (2022) *Research on Intelligent Embedded Fan System Based on YOLOv2 Lightweight Algorithm,* PubMed Central. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9337961/ (Accessed: 26 October 2023).
- Ding, L., Lv, Y., Yu, L., Ma, W. (2022) *Real-time MMonitoring of Fan Operation in Livestock Houses Based on the Image Processing*, ResearchGate. Available at: https://www.researchgate.net/publication/362957125_Real-time_monitoring_of_fan_operation_in_livestock_houses_based_on_the_image_processing (Accessed 26 October 2023).
- Team, T.A. (no date) *Getting started with Arduino products*, *Arduino*. Available at: https://www.arduino.cc/en/Guide (Accessed: 29 October 2023).
- *OpenCV modules* (no date) *OpenCV*. Available at: https://docs.opencv.org/4.x/ (Accessed: 29 October 2023).
- automaticaddison, A. (2020) *How to determine what torque you need for your Servo Motors*, *Automatic Addison*. Available at: https://automaticaddison.com/how-to-determine-what-torque-you-need-for-your-servo-motors/ (Accessed: 29 October 2023).
- Cheng, R. (2021) *Circuit schematic of generic L298N driver board*, *New Screwdriver*. Available at: https://newscrewdriver.com/2021/01/28/circuit-schematic-of-generic-l298n-driver-board/ (Accessed: 29 October 2023).
- Engineer.ahsin. (2022) *WHADDA MG995 Servo Motor Towerpro user Manual*, *Manuals+*. Available at: https://manuals.plus/whadda/mg995-servo-motor-towerpro-manual?expand_article=1 (Accessed: 29 October 2023).

6 Appendix

6.1 Mechanical Design Drawings

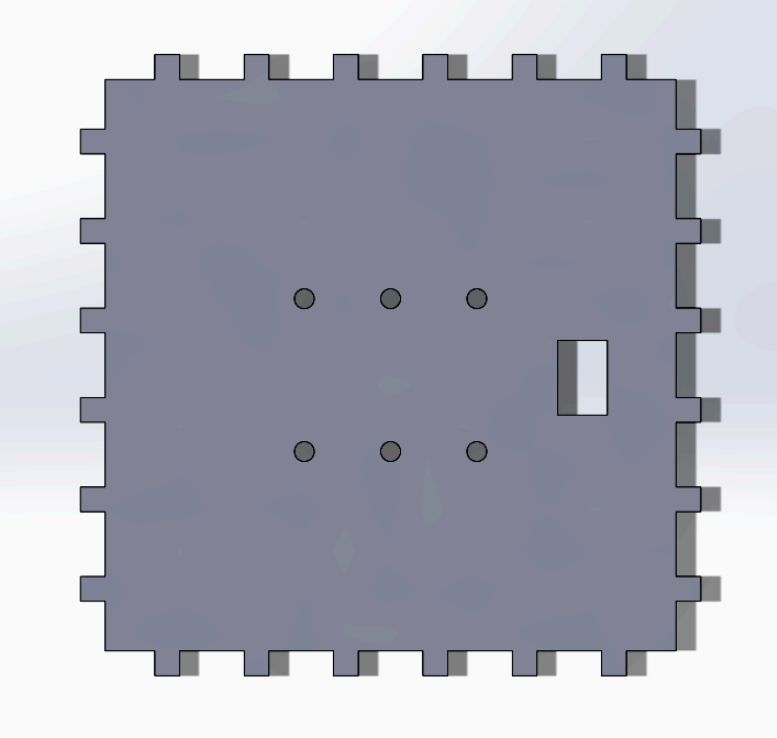

Figure 15: Top and Bottom Sides of Casing

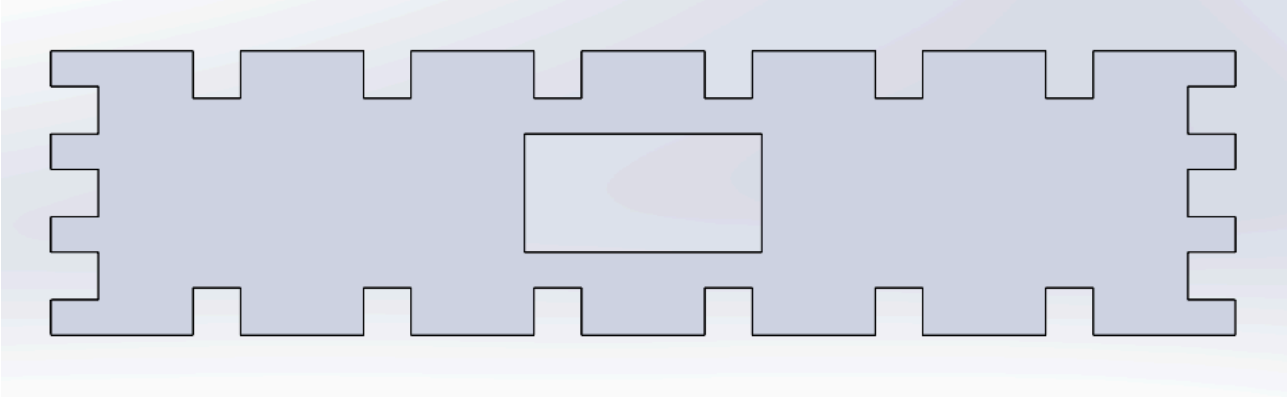

Figure 16: Front and Back Side of Casing

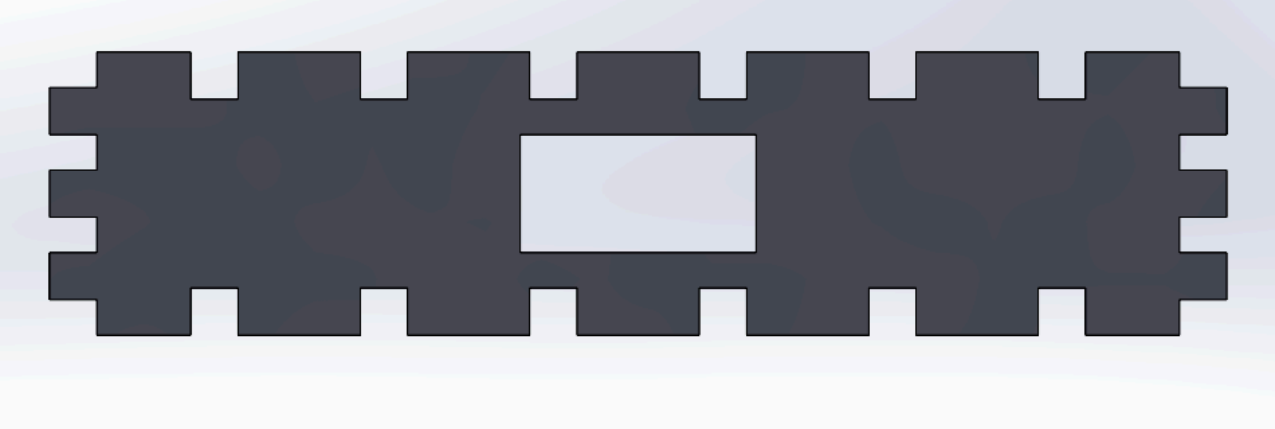

Figure 17: Right and Left Side of Casing

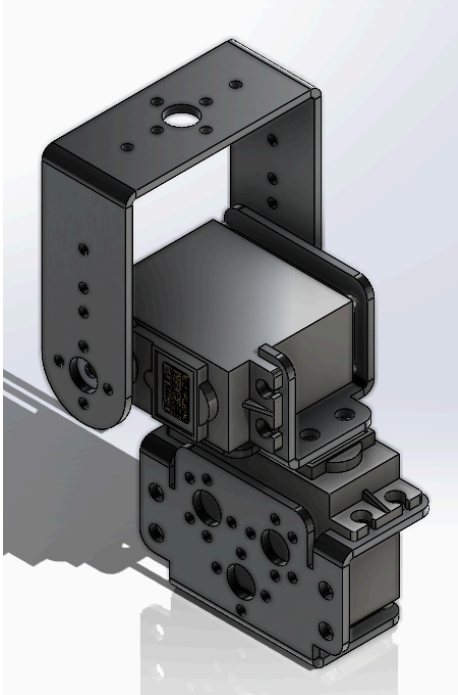

Figure 18: Servo Motors with Brackets (Pan and Tilt Setup)

Figure 19: Brackets to Attach Servo Motors to Case

6.2 Code Listings

Python code

```
import cv2
import serial
import time
import numpy as np

# Initialize the serial connection to Arduino
arduino = serial.Serial('/dev/ttyACM0', 9600)
time.sleep(2)

# Load Haar Cascades
upperbody_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_upperbody.xml')


# Start capturing video
cap = cv2.VideoCapture(2)


W, H = 1920, 1080
cap.set(cv2.CAP_PROP_FRAME_WIDTH, W)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, H)
cap.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc('M', 'J', 'P', 'G'))
cap.set(cv2.CAP_PROP_FPS, 30)
```

```
# Detection buffer
detection_buffer = [0] * 30  # Buffer for the last 30 frames
buffer_index = 0
fan_off_delay = 0

while True:
    _, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


    upperbodies = upperbody_cascade.detectMultiScale(gray, 1.1, 3)
    all_detections = list(upperbodies)

    human_count = len(all_detections)
    detection_buffer[buffer_index] = human_count
    buffer_index = (buffer_index + 1) % len(detection_buffer)

    avg_detection = sum(detection_buffer) / len(detection_buffer)

    if avg_detection < 0.5:  # Adjust
        fan_off_delay += 1
        if fan_off_delay > 30:  # Adjust
            arduino.write("0,0,0\n".encode())
            fan_off_delay = 0
    else:
        fan_off_delay = 0
        for (x, y, w, h) in all_detections:
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
            center_x = x + w // 2
            center_y = y + h // 2
            cv2.putText(img, f"({center_x}, {center_y})", (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255),2)
            cv2.putText(img, f"({center_x}, {center_y})", (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255),2)
            arduino.write(f"{center_x},{center_y},{human_count}\n".encode())
    cv2.imshow('Detection', img)
    if cv2.waitKey(1) == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
arduino.close()
```

Arduino code

```
#include <Servo.h>

Servo panServo;     // Servo for pan movement
int enablePin = 9;  // Enable pin for motor driver

int in1Pin = 2;  // Input pin 1 for motor driver
int in2Pin = 3;  // Input pin 2 for motor driver

void setup() {
  Serial.begin(9600);         // Start serial communication at 9600 baud rate
```

```
  panServo.attach(5);         // Attach servo on pin 5
  pinMode(enablePin, OUTPUT);  // Set fan pin as output
  pinMode(in1Pin, OUTPUT);
  pinMode(in2Pin, OUTPUT);

  digitalWrite(in1Pin, LOW);
  digitalWrite(in2Pin, HIGH);
  analogWrite(enablePin, 0);  // Set speed to 0
}

void loop() {
  if (Serial.available() > 0) {
    String data = Serial.readStringUntil('\n');
    int firstCommaIndex = data.indexOf(',');
    int secondCommaIndex = data.indexOf(',', firstCommaIndex + 1);
    int centerX = data.substring(0, firstCommaIndex).toInt();
    int centerY = data.substring(firstCommaIndex + 1, secondCommaIndex).toInt();
    int humanCount = data.substring(secondCommaIndex + 1).toInt();

    if (centerX == 0 && centerY == 0) {
      // No humans detected
      analogWrite(enablePin, 0);
    } else {
      // Adjust fan speed based on human count
      int fanSpeed = humanCount > 1 ? 255 : 150;

      // Control the fan
      digitalWrite(in1Pin, LOW);  // Set IN1 high to enable the fan
      analogWrite(enablePin, fanSpeed);

      // Gradual Servo Movement
      int currentPos = panServo.read();
      int newPos = map(centerX, 300, 1400, 150, 90);

      if (newPos > currentPos) {
        panServo.write(currentPos + 4);
      } else if (newPos < currentPos) {
        panServo.write(currentPos - 4);
      }
    }
  }
}
```