SMART AIR CIRCULATION CONTROLLER FOR AC SYSTEM

A PROJECT REPORT

Submitted by

A.ABINAND RAMAN

K.P.KAMALNATH

G.RAAHUL

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

AUTOMOBILE ENGINEERING

SRI VENKATESWARA COLLEGE OF ENGINEERING

(An Autonomous institution; Affiliated to Anna University, Chennai-600 025)

ANNA UNIVERSITY :: CHENNAI 600 025

MAY 2021

SRI VENKATESWARA COLLEGE OF ENGINEERING

(An Autonomous institution; Affiliated to Anna University, Chennai-600 025)

ANNA UNIVERSITY :: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**SMART AIR CIRCULATION FOR AC SYSTEM**" is the bonafide work of "**A.ABINAND RAMAN, K.P.KAMALNATH, G.RAAHUL**" who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.J.Venkatesan	Dr.V.Ganesh
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	Associate Professor
Automobile Engineering	Automobile Engineering

Submitted for the project viva-voce examination held on ______________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The effort required in achieving the desired output can be effectively and economically decreased by the implementation of better designs and technologies. Ventilation is mainly used to control indoor air quality; it can also be used to control indoor air temperature, humidity, and air motion to benefit thermal comfort and satisfaction with other aspects of the indoor environment. Considering this air conditioners were introduced within a car that allows to cool the interior air of the vehicles and helps in air circulation. The windows remain closed while using the air conditioner for a long period of time. During this period the level of carbon dioxide surges past the optimal level i.e. oxygen level reduces which can cause nausea, memory disturbance, lack of concentration, tachycardia, and lead to death. This project aims to overcome those effects by building a system with artificial intelligence. This system monitors the rate of carbon dioxide inside the cabin and hence immediately neutralizes the air in the cabin by altering the directions of the flaps on the servo motor to supply optimal air to the cabin from the atmosphere through the vents. The significance and purpose of this work is to provide a hassle-free and comfortable ride by automating the air circulation in the cabin by using artificial intelligence. This project will further reduce the accidents and require less human energy to operate and hence provide more concentration on driving.

ACKNOWLEDGEMENT

It is our privilege to express our sincerest regards to each and every one of those, who gave us their useful insights to aid in our overall progress for this project. Firstly, we take this opportunity to thank our Respected Principal, **Dr. S. GANESH VAITDYANATHAN**, for all his efforts and administration in educating us in his premiere institution.

Furthermore, we thank our Head of the Department, **Dr. J. VENKATESAN**, and all the other Project Review Committee members for their valuable inputs, much helpful guidance, encouragement, whole-hearted cooperation and constructive support throughout the duration of our project work.

We also convey our sincere thanks to our Project Supervisor, **Dr. V. GANESH**, Associate Professor for his much valuable suggestions, guidance, and encouragement throughout the project, and also for his technical support rendered during the course of our project.

We deeply express our sincere thanks to our parents and all the faculty members for their support and encouragement throughout our career. Last but certainly not the least, we express our thanks to our family and friends for their cooperation and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

NO	ABBREVATIONS & SYMBOLS	EXPANSION
1	AC	Air Conditioner
2	HVAC	Heating, Ventilation, and Air Conditioning
3	CO	Carbon Oxide
4	NOx	Nitrogen Oxide
5	CO2	Carbon-di-Oxide
6	NH3	Ammonia
7	VOC	Volatile Organic Compounds
8	OHSA	Occupational Safety and Health Administration
9	ECU	Electronic Control Unit
10	Q	Airflow Rate
11	V	Volts
12	A	Amps
13	GPIO	General Purpose Input/Output
14	Hz	Hertz
15	PWM	Pulse Width Modulation

CHAPTER 1

INTRODUCTION

1.1 AIR CONDITIONER

Car AC (air conditioning) is a system within a car that allows cooling the interior air of the vehicle in hot weather, providing for a cooler environment for the occupants. The concept of air conditioning in a car was first properly developed by the Packard Motor Company in the United States. In 1939, they launched air conditioning as an add-on extra for purchasers of their cars. A car's air conditioning system is made up of three main parts – the compressor, condenser, and the evaporator. They all work together moving a substance called refrigerant through a high-pressure/low-pressure closed-loop system. Refrigerant changes from gas to liquid, and back to gas, and is a vital part of the air conditioning system and process. The compressor is driven by a belt attached to the car engine. This is where the low-pressure refrigerant gas is compressed into a high-pressure, high-temperature gas before being pumped to the condenser. The condenser works like your car's radiator by dissipating out heat but also cooling the high-pressure refrigerant gas so it forms into a high-pressure liquid. This high-pressure liquid then has any water removed from it by a small until called the receiver-dryer before being pumped to the thermal expansion valve. Here, the high-pressure liquid is allowed to expand and become a low-pressure liquid as it enters the low-pressure side of the loop system before moving to the evaporator which is located within the vehicle interior. At this point, the low-pressure refrigerant liquid again turns into a gas and moves out of the evaporator taking the heat from the interior of the vehicle. During this process, a fan blows over the exterior of the compressor, blowing cool air into the interior of the vehicle. The low-pressure refrigerant gas now enters the compressor once more and the process begins again.

1.1.2 NEED FOR AN AIR CONDITIONER IN A VEHICLE

As we have already mentioned, the air conditioning system inside a cabin is required to provide a relaxed and fatigue-free atmosphere to the humans which in turn provides comfort which is essential for the health of a human being. But in a car, it is required due to the following reasons.

- When a vehicle is running through cities like Delhi in India, passengers have to face lots of traffic that causes slow motion of the vehicle due to which a vehicle needs to be equipped with a system that can provide a comfort zone for the passengers.
- In India, during summers there is lots of hot and humid air in the atmosphere which causes difficulty for a driver to drive and also for the passengers to travel, so during summer, an effective air conditioning system is required in a car which can provide a cool and humid free environment inside a passenger’s compartment.
- Air conditioning does not only condition the environment inside a passenger’s cabin but also provides filtered air which is very essential in cities like Delhi due to the very hazardous air pollution condition that can cause health issues to the passengers.
- In winters a vehicle faces a problem of fog or vapor formation over a front mirror which causes visibility problems and makes it difficult for a driver to drive a car, so a vehicle needs to be equipped with a system that can fight this problem.
- There is lots of climate change in countries like India during a long run of a vehicle that can cause health issues to the passengers, so a vehicle should be equipped with a system that can maintain a human comfort zone in the passenger’s cabin throughout the journey.

1.1.3 COMPONENTS OF AIR CONDITIONING SYSTEM IN A CAR

The components of an AC system in a car are almost the same as a room’s AC but there are a lot of modifications made in an automobile AC to make it compact and to fit in with the engine's component. The components used in automobile AC are:-

- **COMPRESSOR**

It is also known as the heart of the AC system, a compressor providing pressure rise to the refrigerant to convert the vapor refrigerant into liquid refrigerant which in turn enables the further flow of the refrigerant through the condenser. The compressor of the car air conditioning system is driven by the crankshaft of the engine through the belt drive.

- **CONDENSER**

It is a device that is similar to a small radiator and is used after the compressor as it provides condensing i.e. lowers the temperature of the high-pressure and high temperature liquid refrigerant sent by the compressor through forced convection provided either by radiator fan or by separated fan used with the condenser.

- **EXPANSION VALVE**

It is a device used in car air conditioning systems to expand the high pressure, low-temperature liquid refrigerant sent by the condenser to release the pressure of the refrigerant before sending it to the evaporator for further process.

- **EVAPORATOR**

It is a device that looks like another heat exchanger and is placed just behind the AC vent over a dashboard of a car. The evaporator takes heat from the passenger's compartment and converts the liquid refrigerant sent by the expansion valve into vapor, which in turn provides cooling through the fan inside a passenger's cabin. A thermal expansion valve is used in vehicles that enable the passenger to change the temperature according to the requirement by just adjusting the knob provided over the dashboard in the passenger's cabin.

- **RECEIVER DRYER**

It is a safety catch used in an automobile or car air conditioning system as there is a chance that, instead of vapors, some liquid also flows towards the compressor which can damage the compressor, so the receiver dryer is used in between evaporator and compressor to convert that remaining liquid into vapors before sending it to the compressor for compression.

- **REFRIGERANT**

  It is a heat sensitive fluid with very low boiling point that is used in AC as a medium of heat exchange.

1.1.4 WORKING OF AN AIR CONDITIONER IN A CAR

- The whole working starts with the Compressor. It compresses or pressurizes the refrigerant and converts it into the liquid from its gaseous state. The compressed liquid refrigerant has to pass through certain tubes located in the condenser. Here, the fresh air from outside comes in contact with liquid refrigerants. The condenser contains a high-temperature liquid and that's why there is a temperature incline between liquid and fresh air. Later the heat moves from the liquid and mixes with the air.
- Then, the refrigerant moves into the receiver dryer or accumulator as shown in Figure 1.1. The desiccant removes the moisture from the air and refrigerant that leads to the creation of a cooler refrigerant while maintaining the system.
- The refrigerant, which is already in the cool liquid state, flows into the expansion valve or orifice tube. This process reduces overall fluid pressure and allows it to move to the evaporator.
- The converted refrigerant will then move to the evaporator. The air from the car will be drawn into the evaporator and go inside the evaporator core. Till now, the refrigerant temperature is cooler and it can convert the outside heat into the cold air.

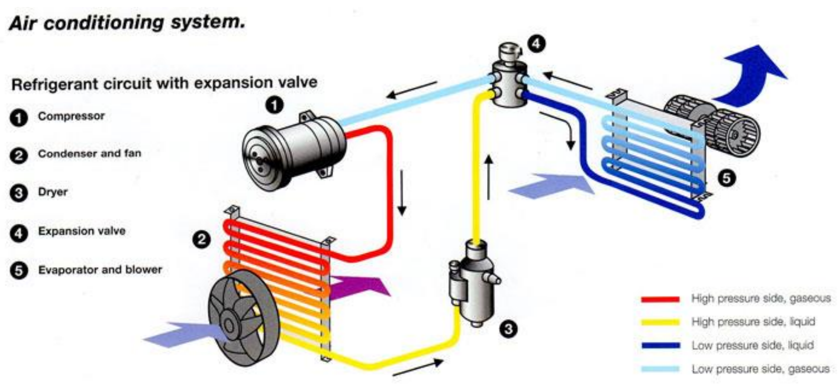

Figure 1.1 Working of an Air Conditioner

- Fans near the passenger seat help in blowing the cold air through vents and make the car's interior temperature cool. This process also removes moisture from the air and allows you to enjoy fresh and dry air. (During this process, the collection and draining of the condensate also take place). As the liquid refrigerant in the AC system becomes hotter after working, it again turns into a gaseous state.
- This hot and low-pressure gaseous refrigerant again circulates and goes back to the compressor. This is how the new cycle takes place and you get the cool, dry and fresh air.

1.2 VENTILATION

Ventilation is the process by which clean air is intentionally provided to a space and stale air is removed. This may be accomplished by either natural or mechanical means.

1.2.1 AUTOMOTIVE HVAC SYSTEM AND CABIN AIRFLOW

The automotive HVAC (heating, ventilation, and air conditioning) system is designed to provide thermal comfort to passengers in all weather conditions and to provide outside air into the vehicle cabin interior. It is also responsible for effective air movement within the cabin and to clear windows and windshield from fog and ice buildup to provide occupant vision safety. The duct distribution system directs air strategically depending on what the occupant demands.

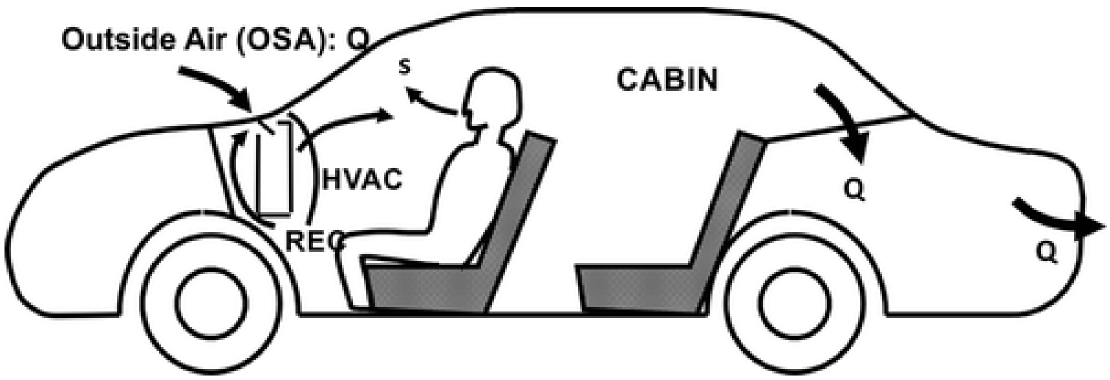

Figure 1.2 Cabin Air Flow and HVAC

Heated air for passenger comfort comes through the heater ducts, which are placed near the cabin floor area, whereas cooler air is generally directed towards the face area. Buoyancy effects play a major role in moving and mixing hot and cold air and the distribution of conditioned air throughout the cabin interior. A door controls the amounts of air passing through the heater core and the evaporator core, allowing either cold or hot air or a mix of the two streams to be delivered. Temperature sensors track and serves as a feedback system that controls the occupant's desired temperature. The recirculation door controls the amount of fresh air intake into the cabin, as sketched in Figure 1.2. In the recirculation mode, the recirculation door shuts the fresh air intake path so that only cabin air is recirculated, although some manufacturers ensure that a minimal amount of fresh air flow occurs even in the recirculation mode. In the outside air mode, the recirculation air path is closed so that 100% of air delivered by the HVAC system comes from outside.

Air leaves the vehicle cabin through the air-relief valves, which are dedicated for that purpose, and through uncontrolled leak paths, which are generally a result of unintentional gaps in the vehicle body. Relief valves are usually placed in areas where there is a negative pressure in the exterior in comparison with the interior cabin pressure when the vehicle is in motion, so that cabin air can readily pass through. They employ one-way, self-closing flaps that open only when there is a positive pressure differential between the cabin interior and exterior. There is generally a slightly higher pressure inside the car cabin so that it is less likely that outside air enters the cabin compartment through leakages. This is of much interest to the car manufacturers so that unwanted odor or dust particles are not allowed to enter the car cabin. Modern-day automobiles are engineered such that uncontrolled leakage through door seams, hinges, etc., is minimized by careful fit-and-finish design execution. Such measures also minimize the conditioned-air exchanges with the outside. As vehicles get older, the gaps in the body generally widen and, hence, an increase in uncontrolled leakage is more common.

1.3 PROBLEMS OF LOW OXYGEN INSIDE A CAR

1.3.1 REASONS FOR POOR AIR QUALITY IN CARS

The three key sources that cause a drop in air quality inside cars are explained in the following paragraphs:

- **EXHAUST FUMES**

Exhaust fumes, including noxious gases like carbon monoxide (CO), oxides of nitrogen (NOx) and hydrocarbons (HC), can enter the vehicle via heating ventilation and air conditioning (HVAC) systems. These gases are present in vehicles at a level beyond the acceptable limits set by the World Health Organization (WHO). They can cause breathing difficulties in people with respiratory problems, such as asthma. In elderly people with cardiovascular risk, these gases can even trigger heart attacks. They are also known to cause dizziness, headaches, poor hand-eye coordination and nausea, which all could result in accidents. Accidental poisoning could be the result of carbon monoxide gas released by cars. Traffic jams are potential scenarios that increase the chances of drivers being exposed to harmful exhaust fumes.

- **CAR INTERIOR MATERIALS**

Materials used in the fabrication of the interiors such as the dashboard, seats and steering wheel, can also impact the quality of air within the vehicle. This is because of the use of flame retardants comprising bromine and antimony, PVC to produce the seats, and chromium to treat leather. All of these can emit volatile organic compounds (VOCs). When cars are parked in the sun, or when temperatures rise due to turning on the heating, the chances for VOC emissions increase. Symptoms, including headaches, dizziness and nausea, can occur due to exposure to VOCs.

- **BREATHING IN AN ENCLOSED ENVIRONMENT**

Poor air quality can also be the result of the abnormal breathing process when it happens in an enclosed environment. For instance, in a car with the windows closed and HVAC being used in recycle mode to stop pollutants from entering the vehicle, oxygen is depleted over time and displaced by carbon dioxide. According to the Occupational Safety and Health Administration (OHSA) in the USA, below 19.5% oxygen concentration is categorized as hazardous. One study revealed that in a car with two people the oxygen levels were found to be just 19.1% after 15 minutes. Increased heart rate and impaired muscle coordination and judgment are experienced by drivers when oxygen levels are depleted. Similarly, increased carbon dioxide levels may cause drowsiness, leading to fatigue. This is already one of the most serious concerns for road safety. Driver’s reaction times can become slower, thus making them less alert. Studies show that nearly 20% of all accidents and a quarter of fatal or serious ones are caused because of fatigue.

1.3.2 HAZARDS OF DRIVING UNDER LOW OXYGEN LEVELS

Low oxygen levels can result in abnormal blood circulation and cause the following symptoms:

- Shortness of breath
- Headache
- Restlessness
- Dizziness
- Rapid breathing
- Chest pain

- Confusion
- High blood pressure
- Visual disorders
- Sense of euphoria

1.3.2.1 SEVERITY OF LOW OXYGEN

Hypoxemia, or oxygen levels below the normal values, may be caused by:

- Not enough oxygen in the air
- The inability of the lungs to inhale and send oxygen to all cells and tissues
- The inability of the bloodstream to circulate to the lungs, collect oxygen, and transport it around the body
- Memory disturbance, lack of concentration.
- Causes tachycardia and it might lead to death.

1.4 SOLUTION

The basic solution to eradicate the problem mentioned above is by maintaining a sustainable level of oxygen within the cabin.

1.4.1 OUR SOLUTION TO THE PROBLEM

Our project aims to provide the finest breathable air to the passengers by maintaining the optimal level of oxygen inside the vehicle's cabin with the help of microcontrollers.

1.5 MICROCONTROLLER

A microcontroller is a small computer on a single metal-oxide-semiconductor integrated circuit chip. A microcontroller contains one or more CPUs along with memory and programmable input/output peripherals.

1.5.1 MICROCONTROLLER IN A CAR

In specific, the existence of various microcontrollers in automobiles plays a vital role. Microcontrollers in automobiles are present in large equipment to small devices and are used to manage ECU (Electronic Control Unit) functions such as air conditioners, power windows, braking, headlights, seats, steering, taillights, steering, greater vehicle safety, lower fuel consumption, lower engine pollution, and higher automotive reliability. An onboard computer, for example, could determine continuously the optimum fuel/air ratio in response to engine temperature, engine speed, and load conditions. Similarly, the computer, in principle, could limit the amounts and kinds of pollution in engine exhausts through appropriate ignition control. Especially, the range of ECU's in luxury cars like the BMW seven series and MERCEDES MAYBACH is from 60-65. Microcontroller's adaptability and flexibility provide high performance and low power consumptions and bring a revolutionary change in all aspects of the automobile industry.

1.5.2 TYPES OF MICROCONTROLLERS USED IN A CAR

Different microcontrollers used in an automobile can communicate with one another through multiplexing. These microcontrollers can manage related systems separately by using a BUS to communicate with other networks when they are required to perform a function. The combination of several linked networks includes the controller area networks. Present controller area networks permit complex interactions that involve sensory systems, car speed, outdoor rainfall interactions, and in-car temperature with performance controls for air conditioning maintenance, audio-visual multimedia systems and braking mechanisms.

- **INFINEON TRI-CORE MICROCONTROLLER**

Tri-core is a 32- bit microcontroller, which is developed by Infineon. These microcontrollers are assembled in over 50 automotive brands which means that every second a vehicle designed today, includes a Tri-core based microcontroller. It is responsible for keeping the exhaust emissions and fuel consumption as low as possible. Tri-core microcontrollers are used in the gearboxes to control the injection and central control units for the combustion engine's ignition. Progressively, they are also being used in electrical and hybrid vehicle drives.

- **ATMEL AVR MICROCONTROLLER**

Atmel AVR microcontrollers distribute the power, performance and flexibility for automobile applications. This microcontroller consists of Harvard architecture. So the device runs very fast with a reduced number of machine-level instructions. The main features of AVR microcontrollers compared to other microcontrollers include inbuilt ADC, 6-sleep modes serial data communication and internal oscillator, etc.

1.5.3 APPLICATIONS OF MICROCONTROLLER IN AUTOMOBILES

The existence of various microcontrollers in automobiles plays a vital role in various aspects respective to the vehicle and the driver.

1.5.3.1 VEHICLE CONTROL

- 16-bit microcontrollers are mostly used
- Computational performance key decision factor
- Design Cycle is 3 to 4 years

1.5.3.2 POWER TRAIN

- 16 to 32-bit Microcontrollers are used mostly
- Key Decision Factor.
- Design Cycle is 4 to 5 years

1.5.3.3 DRIVER INFORMATION

- 8-Bit microcontrollers are used mostly
- High Integration and low EMI Key decision factor
- Design Cycle is 3 to 4 years

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

The various research works attempted in the area of measurement of airflow, effects of car speed on ventilation and concentration of carbon dioxide have been referred to and discussed here.

2.2 LITERATURE REVIEW

[1] Milos Fojtlin et al (2016) interpreted the distribution of air within a passenger vehicle with a focus on airflow through HVAC vents as shown in Figure 2.1 and the calculation of corresponding air distribution coefficients using hot-wire anemometer. They analyzed the measurements by determining the airflow through the vents in the car and by examining the inlet airflow to the HVAC system outside for summer, winter, and spring/fall conditions. As per their analysis, the acceptable difference between the estimation and measure airflow is 8.3% as shown in Figure 2.2.

Figure 2.1 Arrangement of vents in a tested vehicle

flow	inlet	outlet	difference in-out	
			absolute	relative
($g\ s^{-1}$)	68.6 ± 0.3	70.2 ± 3.1	1.6	2.3 %
($l\ s^{-1}$)	59.0 ± 0.2	57.4 ± 2.6	1.6	2.7 %

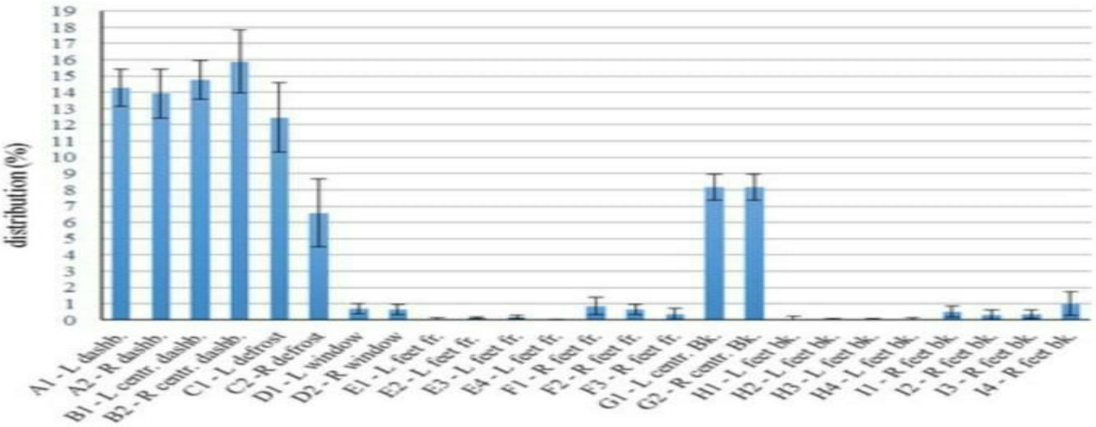

Figure 2.2 Case distribution coefficients with 95% confidence error bar

Their extensive research is crucial as it helps to understand and estimate the opening and closing of the flaps to facilitate the supply of oxygen and also for the placement of sensors because there are different vents for the passage of air to flow into the car cabin.

[2] Jan Fiser and Jan Pokorny (2014) perused that the supply of airflow rate is strongly dependent on the speed of the car when the fan speed is set at the second level in combination with the position of heating mixing flap on 1/2 of full open. In such a case, the increase of the volume flow is from 21.7 l/s at 0 km/h to 31.1 l/s at a car speed of 90 km/h which is equal to the total increase of 43.2%. In case of higher fan speed or a lower mixing ratio of heating, increase of supply air is lower 18.3% for the combination of the Fan speed 2 + mixing flap in position 1/2, and 6.1% for the combination of Fan speed 4 + mixing flap of heating in position ¼.

From their meticulous study, it is acceptable that altering the angles of the flaps lets the airflow into the cabin with respect to the position of the flap angle, i.e. greater the opening angle of the flap, the is more the supply of air and vice-versa.

[3] Maytat Luangprasert et al (2017) investigated the in-vehicle carbon dioxide concentration in commuting cars. They analyzed that there is generally a higher pressure inside the car cabin compared to the outside air and so it is less likely that the outside air enters the cabin compartment through leakages. The airflow rate depends on the HVAC system blower speed, the recirculation door position, the vehicle speed, and to some extent the vane/louver positions of the air vents. They further examined that when the vehicle is stationary, the airflow rate Q (outside air ventilation rate through the vehicle cabin) is more susceptible to prevailing wind around the vehicle, whereas when the vehicle is driven at a steady speed, the relative air velocity is more or less constant. When the vehicles are parked for a long time, the CO2 concentration decays gradually to an acceptable level.

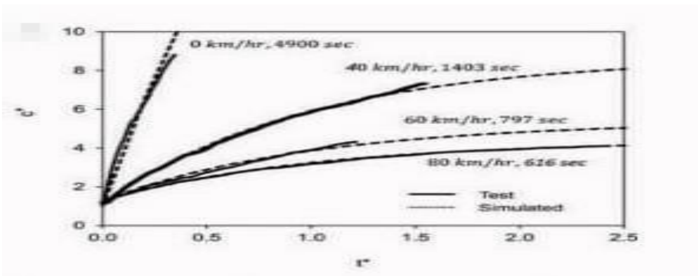

Figure 2.3 Dimension-less CO2 concentration for test runs

From thoroughly scrutinizing their research it is clearly understood that there is slightly higher pressure inside the car cabin rather than the outside atmosphere. So, it is less likely that outside air enters the cabin compartment through leaks. Further, it was understandable that the co2 concentration depletes with respect to time as shown in Figure 2.3. The airflow rate varies with different vehicle speeds and correspondingly the CO2 concentration depletes with respect to time as shown in Figure 2.3.

[4] R A Angelova et al (2019) examined the accumulation of metabolic carbon dioxide in a vehicle cabin. They perused that the increment of CO2 concentration in the cabin was at the expense of oxygen availability. The regular breathing of a single person in the car cabin in recirculation is modeled to a 1.7% decrement in the O2 concentration, because of the very long time of the exposure and the gradual oxygen depletion.

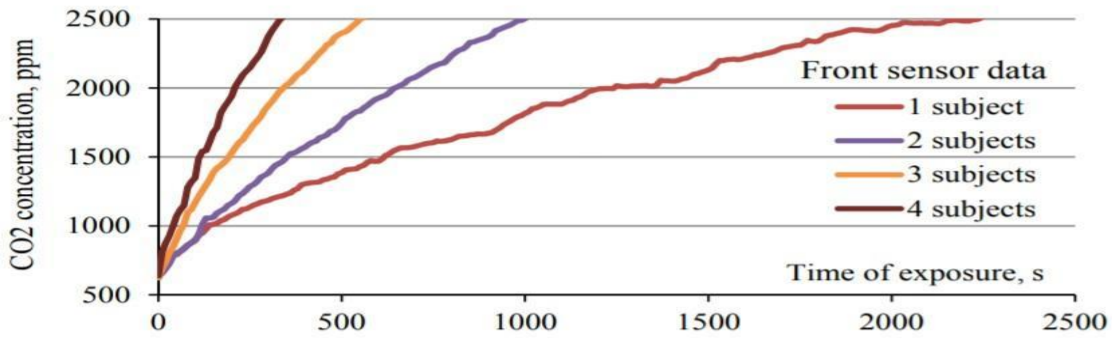

Figure 2.4 CO2 concentration in the car cabin: effect of the number of passengers.

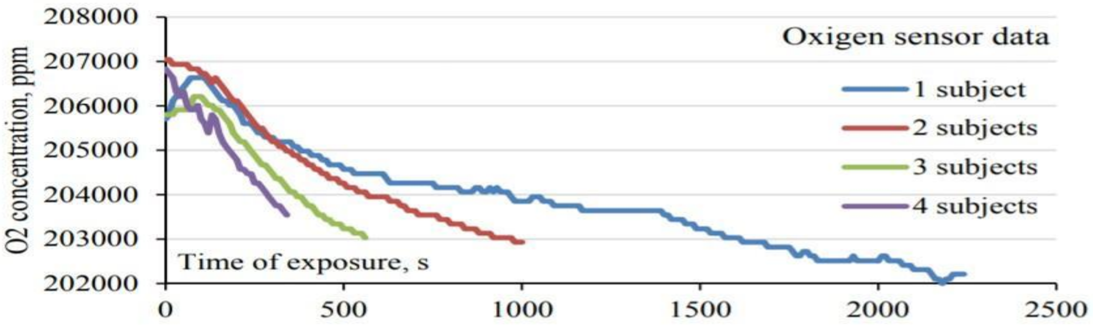

Figure 2.5 O2 concentration in the car cabin: effect of the number of passengers.

From their meticulous research and scrutiny, it is acceptable that the oxygen concentration shown in Figure 2.5 and the CO2 concentration shown in Figure 2.4 in the car cabin changes steadily or rapidly depending upon the number of passengers inside the car cabin.

CHAPTER 3

INTRODUCTION TO HARDWARE AND SOFTWARE COMPONENTS

3.1 INTRODUCTION

This chapter introduces the various components and software used to accomplish the final result. We have used a variety of components with respect to numerous aspects concerning this project and we have specified it as well.

3.2 SENSORS

A sensor is a device that detects a change in the environment and responds to some output on the other system. A sensor converts a physical phenomenon into a measurable analog voltage (or sometimes a digital signal) converted into a human-readable display or transmitted for reading or further processing. The sensors used in this project are as follows:-

3.2.1. MQ 135 GAS SENSOR

MQ135 Gas Sensor is an air quality sensor for detecting a wide range of gases, including NH3, NOx, alcohol, benzene, smoke and CO2. Ideal for use in office or factory.

Figure 3.1 MQ 135 Sensor

MQ135 gas sensor has a high sensitivity to Ammonia, Sulfide and Benzene steam, also sensitive to smoke and other harmful gases. It is cost effective & particularly suitable for Air quality monitoring applications.

3.2.1.1 PIN CONFIGURATION

Pin No	Pin Name	Description
For Module		
1	Vcc	Used to power the sensor. Generally, the operating voltage is +5V.
2	Ground	Used to connect the module to system ground.
3	Analog Out	You can also use this sensor to get digital output from this pin, by setting a threshold value using the potentiometer.
4	Digital Out	This pin outputs a 0-5V analog voltage based on the intensity of the gas.

Table 3.1 Pin Configuration of MQ 135 Sensor

3.2.1.2 MQ-135 Sensor Features

- Wide detecting scope
- Fast response, high sensitivity, stable and long life
- Operating Voltage is +5V
- Detect/Measure NH3, NOx, alcohol, Benzene, smoke, CO2, etc.
- Analog output voltage: 0V to 5V
- Digital output voltage: 0V or 5V (TTL Logic)
- Preheat duration 20 seconds
- Can be used as a digital or analog sensor
- The Sensitivity of the digital pin can be varied using the potentiometer

3.2.2 Pi CAMERA MODULE V2

The Pi camera module is a portable lightweight camera that supports Raspberry Pi. It is normally used in image processing, machine learning, or surveillance projects. It is commonly used in surveillance drones since the payload of cameras is very less. Apart from these modules, Pi can also use normal USB webcams that are used along with computer

Figure 3.2 Pi Camera Module

3.2.2.1 Pi camera Features

- 5 megapixels color camera module without a microphone for Raspberry Pi
- Supports both Raspberry Pi Model A and Model B
- MIPI Camera serial interface
- Omnivision 5647 Camera Module
- Resolution: 2592 * 1944
- Supports: 1080p, 720p and 480p
- Lightweight and portable

3.2.3 HC-SR04 Ultrasonic Sensor

The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package from 2 cm to 400 cm or 1” to 13 feet.

Figure 3.3 HC-SR04 Ultrasonic Sensor

3.2.3.1 Working

As shown above in the Figure 3.3 HC-SR04 Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that distance is equal to speed times time.

Speed of sound at sea level = 343 m/s or 34300 cm/s Eqn: 3.1

Distance =17150 * Time Eqn: 3.2

Where 17150 = ½ * speed at sea level

3.2.3.2 Pin Configuration

Pin No	**Pin Name**	**Description**
1	Vcc	The Vcc pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

Table 3.2 Pin Configuration of HC-SR04 Ultrasonic Sensor

3.2.3.3 HC-SR04 Sensor Features

- Operating voltage: +5V
- Theoretical Measuring Distance: 2cm to 450cm
- Practical Measuring Distance: 2cm to 80cm
- Accuracy: 3mm
- Measuring angle covered: <15°

- Operating Current: <15mA
- Operating Frequency: 40Hz

3.2.3.4 APPLICATIONS

- Used to avoid and detect obstacles with robots like biped robot, obstacle avoider robot, path finding robot etc.
- Used to measure the distance within a wide range of 2cm to 400cm.
- Can be used to map the objects surrounding the sensor by rotating it.

3.3 MICROCONTROLLER

A microcontroller is an integrated circuit (IC) device used for controlling other portions of an electronic system, usually via a microprocessor unit (MPU), memory, and some peripherals. These devices are optimized for embedded applications that require both processing functionality and agile, responsive interaction with digital, analog, or electromechanical components.

3.3.1RASPBERRY Pi 4

The Raspberry Pi 4 Model B is the latest version of the low-cost Raspberry Pi computer. The Raspberry Pi is a low-cost, credit-card sized computer that plugs into a computer monitor or TV.

The Raspberry Pi 4 can surprisingly do a large number of tasks. Amateur tech enthusiasts use Pi boards as media centers, file servers, retro games consoles, routers, and network-level ad-blockers, for starters. With the Pi 4 being faster, able to decode 4K video, benefiting from faster storage via USB 3.0, and faster network connections via true Gigabit Ethernet, the door is open to many new uses. It's also the first Pi that supports two screens at one up to dual 4K@30 displays.

3.3.1.1 SPECIFICATIONS OF RASPBERRY PI

- **System-on-a-chip:** Broadcom BCM2711
- **CPU:** Quad-core 1.5GHz Arm Cortex-A72 based processor
- **GPU:** Video Core VI
- **Memory:** 1/2/4GB LPDDR4 RAM

- **Connectivity:** 802.11ac Wi-Fi / Bluetooth 5.0, Gigabit Ethernet
- **Video and sound:** 2 x micro-HDMI ports supporting 4K@60Hz displays via HDMI 2.0, MIPI DSI display port, MIPI CSI camera port, 4 pole stereo output and composite video port
- **Ports:** 2 x USB 3.0, 2 x USB 2.0
- **Power:** 5V/3A via USB-C, 5V via GPIO header
- **Expandability:** 40-pin GPIO header

3.3.1.2 PIN CONFIGURATION

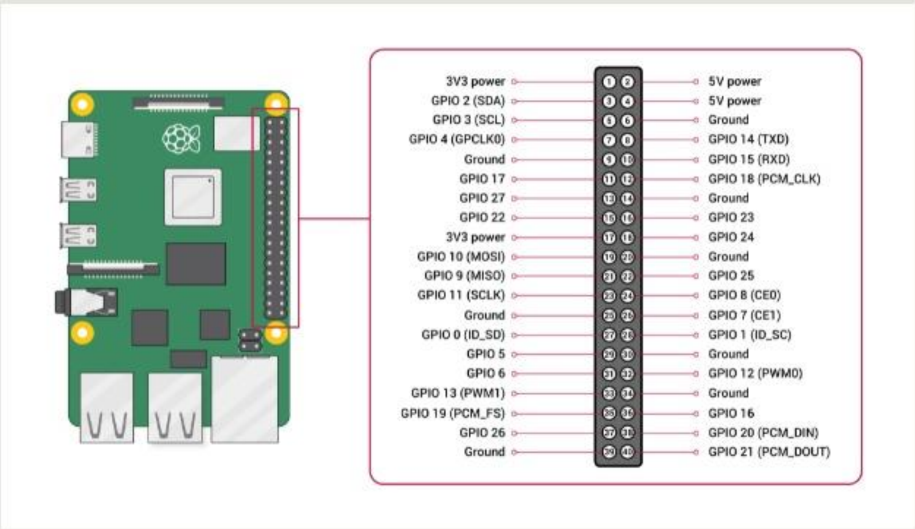

Figure 3.4 Pin Configuration of Raspberry Pie 4

3.4 MOTORS

An electric motor is an electrical machine that converts electrical energy into mechanical energy. Most electric motors operate through the interaction between the motor's magnetic field and electric current in a wire winding to generate force in the form of torque applied on the motor's shaft. Electric motors can be powered by direct current sources, such as from batteries, or rectifiers, or by alternating current sources, such as a power grid, inverters or electrical generators. From all the available motors we have used the SG 90 Servo motor for the purpose of rotating the flaps.

3.4.1 SERVO MOTOR

A servo motor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors. Servomotors are not a specific class of motor, although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system. Servomotors are used in applications such as robotics, CNC machinery, or automated manufacturing.

Figure 3.5 Servo Motor SG-90

3.4.1.1 WIRE CONFIGURATION

Wire Number	Wire Colour	Description
1	BROWN	The ground wire connected to the ground of the system.
2	RED	Powers the motor typically +5V is used
3	ORANGE	PWM signal is given in through this wire to drive the motor

Table 3.2 Wire Configuration of SG-90 Servo Motor

3.4.1.2 SG-90 Servo Motor Features

- Operating Voltage is +5V
- Typically Torque: 2.5kg/cm
- Operating speed is 0.1s/60°
- Gear Type: Plastic Rotation : 0°-180°

- Weight of motor: 9gm Package includes gear horns and screws.

3.4.1.3 DIMENSIONS OF SG-90 SERVO MOTOR

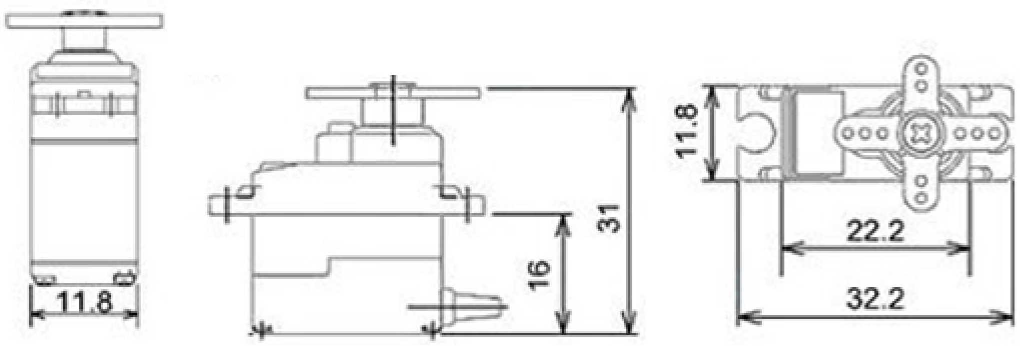

Figure 3.6 Dimensions of servo motor

3.5 MICROPROCESSOR

A microprocessor is an electronic component that is used by a computer to do its work. It is a central processing unit on a single integrated circuit chip containing millions of very small components including transistors, resistors, and diodes that work together. Some microprocessors in the 20th century required several chips. Microprocessors help to do everything from controlling elevators to searching the Web. Everything a computer does is described by instructions of computer programs, and microprocessors carry out these instructions many millions of times a second.

There are various microprocessors available in the market but we used the Arduino Mega 2560.

3.5.1 ARDUINO MEGA 2560

Arduino board is an open-source microcontroller board that is based on Atmega 2560 microcontroller. The growth environment of this board executes the processing or wiring language. These boards have recharged the automation industry with their simple-to-utilize platform wherever everybody with small otherwise no technical backdrop can start by discovering some necessary skills to program as well as run the Arduino board. These boards are used to extend separate interactive objects otherwise we can connect them to a software on a PC like MaxMSP, Processing, and Flash.

It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 hardware serial ports, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila. The Mega 2560 is an update to the Arduino Mega.

3.5.1.1 PIN CONFIGURATION

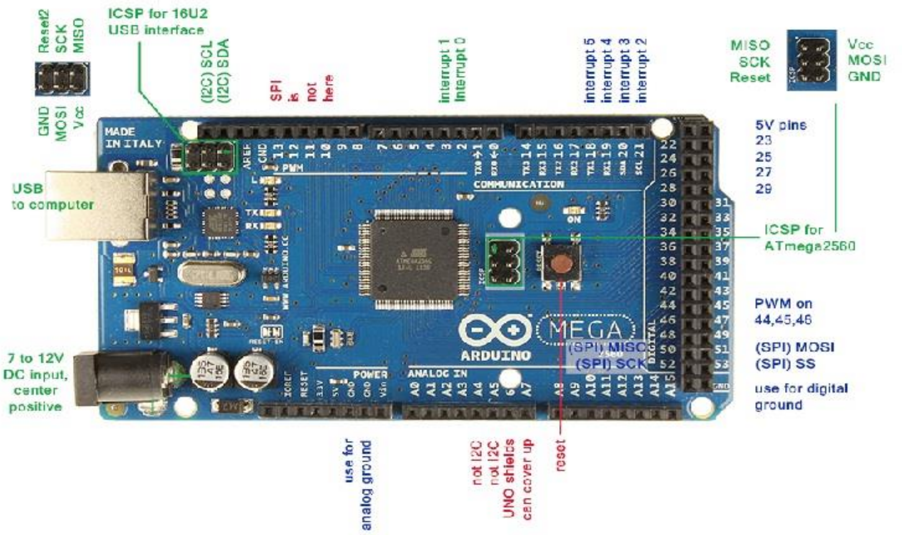

Figure 3.7 Pin Configuration of Mega 2560

3.5.1.2 SPECIFICATIONS OF ARDUINO MEGA 2560

- The input voltage will range from 6V to 20V.
- The clock (CLK) speed is 16 MHz
- The length of this board is 101.52 mm.
- The width of this board is 53.3 mm.
- The weight of this board is 36 g
- **Serial Communication:** The serial pins of this board like TXD and RXD are used to transmit & receive the serial data. TX indicates the transmission

of information whereas the RX indicates receive data. The serial pins of this board have four combinations. For serial 0, it includes TX (1) and RX (0), for serial 1, it includes TX (18) & RX (19), for serial 2 it includes TX (16) & RX (17), and finally for serial 3, it includes TX (14) & Rx (15).

- **Programming:** The programming of an Arduino Mega 2560 can be done with the help of an IDE (Arduino Software), and it supports the C-programming language. Here the sketch is the code in the software which is burned within the software and then moved to the Arduino board using a USB cable.

3.6 SOFTWARE

Computer software, also called software, is a set of instructions and documentation that tells a computer what to do or how to perform a task. The software includes all different programs on a computer, such as applications and the operating system. Applications are programs that are designed to perform a specific operation, such as a game or a word processor. The operating system (e.g. Mac OS, Microsoft Windows, Android and various Linux distributions) is a type of software that is used as a platform for running the applications and controls all user interface tools including the display and the keyboard. The softwares used in this project are Python and OpenCV.

3.6.1 PYTHON 3

Python is an open-source programming language that was made to be easy to read and powerful. Python is an interpreted language. Interpreted languages do not need to be compiled to run. A program called an interpreter runs Python code on almost any kind of computer. This means that a programmer can change the code and quickly see the results. This also means Python is slower than a compiled language like C because it is not running machine code directly. Python is a good programming language for beginners. It is a high-level language, which means a programmer can focus on what to do instead of how to do it. Writing programs in Python takes less time than in some other languages. Python is used by hundreds of thousands of programmers and is used in many places. Sometimes only Python code is used for a program, but most of the time it is used to do simple jobs while another programming language is used to do more complicated tasks. Its standard library is made up of many functions that come with Python when it is installed. On the Internet, there are many libraries available in the internet that makes it possible for Python language to do more things. These libraries make it a powerful language, it can do many different things.

Python is a wonderful and powerful programming language that’s easy to use with Raspberry pi, lets you to the real world. Python syntax is very clean, with an emphasis on readability and uses standard English keywords.

3.6.2 OPENCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

CHAPTER 4

METHODOLOGY

4.1 INTRODUCTION

This chapter depicts the working of sensors under the input conditions and processes the output according to the input data provided. The sensors work one at a time and the working sequence of these sensors is discussed.

4.2 FLOWCHART

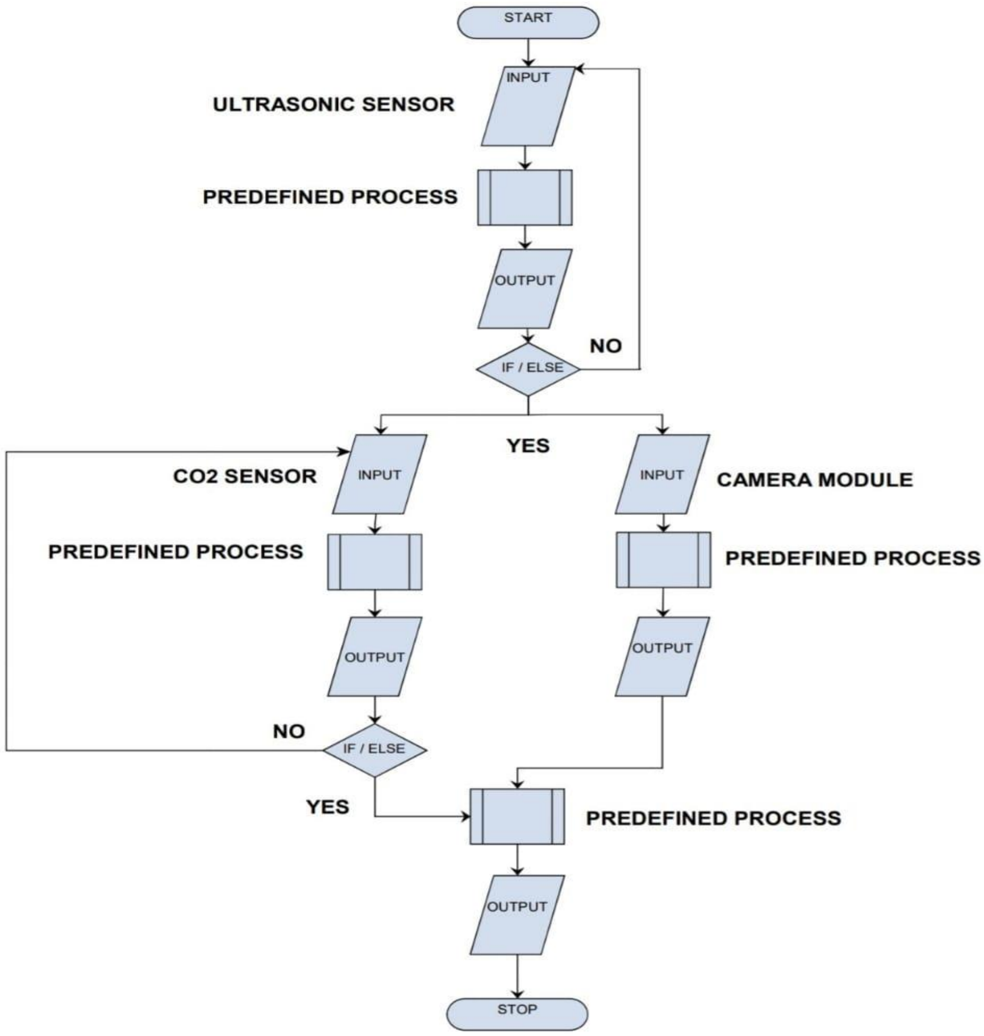

Figure 4.1 Flowchart of working sequence

A flowchart is a “formalized graphic representation” that illustrates a workflow or process or a solution to a given problem. Flow charts can be used to represent a logic sequence, a manufacturing process, or an organizational chart. The common intent behind creating a flowchart is to explain the various stages of a process or a business project to audiences.

4.3 FUNCTIONALITY AND MOUNTING OF SENSORS

4.3.1 ULTRASONIC SENSOR

The ultrasonic sensor is mounted near the door hinges in the door lock at both the B-Pillars. A total of four individual sensors are mounted to each of the doors. The ultrasonic sensor identifies the opening and closing of the doors. This sensor can measure a distance of about 2-400 cm. So this sensor when placed near the door can detect if the door is open or closed.

4.3.2 CAMERA MODULE V2

The camera module is mounted in the center of the roof near the lights or the microphone as it captures all the passengers without any obstruction. This camera identifies the number of passengers inside the vehicle and further sends the processed signal.

4.3.3 GAS SENSOR

The gas sensor is mounted on two locations inside the cabin. Two sensors are mounted at the front and another two sensors are mounted at the rear, a combination of four sensors is used. These sensors sense the quality of the air inside the cabin and the mean value is sent as the output.

4.3.4 SERVO MOTOR

The servo motor is mounted inside the air duct flaps which is not visible from the cabin. This motor is used to alter the angle of the flaps to facilitate the passage of airflow. This motor receives a signal from the microcontroller.

4.3.5 RASPBERRY Pi-4

The raspberry pi microcontroller is mounted inside the cabin cluster. This microcontroller receives input from the sensors and actuates the servo motor.

4.4 WORKING SEQUENCE

The ultrasonic sensor senses the opening and closing of the doors. It is mounted in the doors since a passenger can only enter and exit through the doors. Every time a door is opened and closed the sensor is first triggered. This is the point where the entire system starts to work. The ultrasonic sensor sends the binary signal to the raspberry pi.

The camera sensor receives a signal from the raspberry pi. The camera sensor then recognizes the number of passengers inside the cabin. To ensure the correct count of passengers the camera is placed high in the cabin roof so that it could capture the exact number of passengers. Then the signal is transferred to the raspberry pi.

Simultaneously the CO2 sensors are triggered. These sensors sense the CO2 level inside the cabin and the mean value is sent as output to the raspberry pi. If the CO2 level inside the cabin increases above the optimal level then the sensor senses it and the signal is sent to the raspberry pi and immediately the flaps on the servo motor is opened to facilitate the passage of fine air until the optimal level of CO2 is reached. In case the air inside the cabin is maintained below or at the optimal CO2 level then the flaps don’t open. In this case the sensor predominantly loops back measure the CO2 level. The CO2 sensor senses at a frequency of every second.

The servo motor receives a signal from the microcontroller and according to the input, the flaps are opened or closed based on the result provided by the CO2 sensor. By opening the flaps, fine air is sent into the cabin to neutralize the CO2 level.

The raspberry pi receives a signal from the ultrasonic sensor and sends the signal simultaneously to the camera module and the CO_2 sensor. Based on the signal received from the CO_2 sensor and the camera module, the microcontroller sends the required signal to the servo motor to alter the flap angle of the servo motor.

CHAPTER 5

INTERFACING AND PROGRAMMING OF COMPONENTS

5.1 INTRODUCTION

This chapter depicts the circuit connections of various components connected to the Raspberry Pi along with their programmed code as snippets. The entire coding process for individual components is listed in the appendix for reference.

5.2 CAPTURING NUMBER OF PASSENGERS

The camera module is mounted with the Raspberry Pi in the given mounting slot as shown in Figure 5.1. OpenCV is installed in raspberry pi after successful mounting of the camera module in the respective slot. Using the Haar Classifier algorithm a pre-trained model available in the OpenCV library. This algorithm uses edge or line detection features and the algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them. The model created from this training is available at the OpenCV GitHub repository. The repository has the models stored in extensible markup language files and can be read with the OpenCV methods. After loading the pre-trained model the camera identifies a single face and outputs with the bounding box. A bounding box is an imaginary rectangle that serves as a point of reference for object detection and creates a collision box for that object. The required output from the camera module should be a numeric value of how many passengers available inside the cabin. After creating a function that will look for multiple faces and puts a bounding box on each of them, later those bounding boxes will be counted and the numeric value will be returned as the output from this function. This process is looped for 30 seconds and the numeric value for each second will be stored in an array, later the model of the array is outputted as the number of passengers, and this value is returned to the microcontroller.

Figure 5.1 Raspberry Pi with Camera Module

```
def face_dec():
    global i, a,  i = 0,  a = []
    face_cascade =
cv2.CascadeClassifier('/home/pi/opencv/data/haarcascades/haarcascade_frontalf
ace_default.xml')
    cap = cv2.VideoCapture(0)
    while True:
        ret, frame = cap.read()
        frame=cv2.flip(frame, 1)
        frame=cv2.resize(frame, (0,0), fx=0.2, fy=0.2)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)
        i = 0
```

```
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 5)
            i = i + 1
            cv2.putText(frame, 'face num' + str(i), (x-10, y-10),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
            a.append(i)
        cv2.imshow('Test', frame)
        k = cv2.waitKey(1) & 0xff
        if k==27:
            break
    cap.release()
    cv2.destroyAllWindows()


def passenger_count():
    global passenger_count
    m = stats.mode(a)
    passenger_count = int(m[0])
```

The above two code snippets are the functions that handle capturing faces and calculating them.

5.3 CALCULATING CO_2 ppm

After capturing the number of passengers the next main value to be calculated is the CO_2 ppm level. MQ135 sensor has an operating voltage of 5V whereas the Raspberry Pi works only in 3.3V. Therefore a need for a 5V to 3.3V logic converter arose and unfortunately, the General-Purpose Input/Output (GPIO) pins in Raspberry Pi can only read digital signals whereas the sensor value from MQ135 is analog. This situation can only be handled by using an MCP3008 Integrated circuit (IC). The MCP3008 IC converts the analog signals to digital signals which will be comparable in Raspberry Pi. After mounting the MQ135 gas sensor with the Raspberry Pi with the help of MCP3008 and 5V to 3.3V logic converter, The sensor was ready to capture the CO_2 in the atmosphere, But programming and the wiring for the gas sensor to calculate the CO_2 level was complex because of the addition of MCP3008 and the logic converter as shown in Figure 5.2 and during testing the sensor the accuracy was not up to the mark. Later the CO_2 was mounted with an Arduino because this microprocessor works in 3.3V and can read analog values as shown in Figure 5.3. This eliminates the usage of the components MCP3008 and logic converter. Simultaneously the complexity of the program was reduced by a significant amount. Later the Arduino is connected to Raspberry Pi through the serial connection. Serial connections are used to communicate between the Arduino board and a computer or other devices like Raspberry Pi as shown in Figure 5.4. All Arduino boards have at least one serial port for serial communication known as a UART or USART.

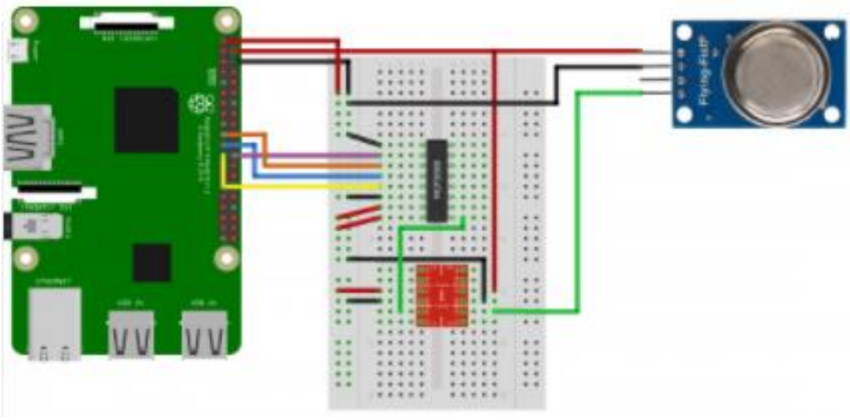

Figure 5.2 Connection of MQ135 Using MCP3008 and Logic Converter in Raspberry Pi

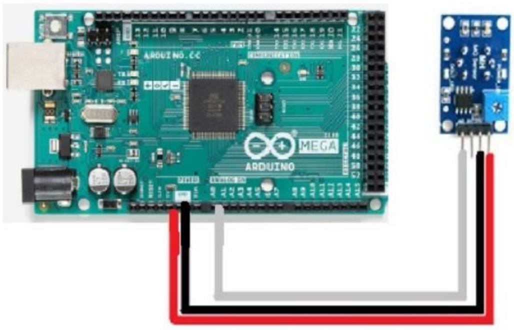

Figure 5.3 Connection of MQ135 Sensor with Arduino

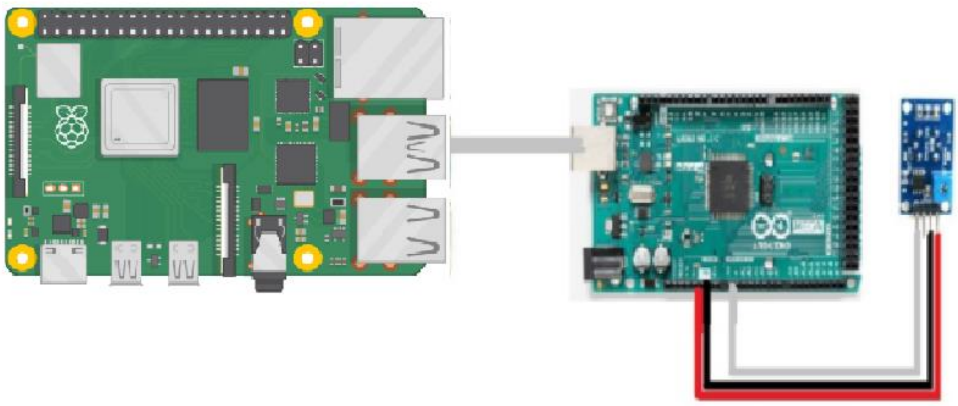

Figure 5.4 Serial connection between Arduino and Raspberry Pi

```
  def call(self):
      ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
      ser.flush()
      while self.i <= 10:
          read_serial = ser.readline().decode('utf-8').rstrip()
          if len(read_serial) > 0:
              self.ppm = int(read_serial)
              if self.max < self.ppm:
                  self.max = self.ppm
          self.i = self.i+1
      return self.max
```

The above is code snippet helps to communicate the MQ135 sensor, whereas the below code snippet is an Aurdino code that calculates the CO2 level in the atmosphere.

```
void loop()
{int co2now[10];                    //int array for co2 readings
 int co2raw = 0;                    //int for raw value of co2
 int co2ppm = 0;                    //int for calculated ppm
 int zzz = 0;                       //int for averaging
 for (int x = 0;x<10;x++)           //sample co2 10x over 2 seconds
 {co2now[x]=analogRead(A0);
```

```
  delay(200)}
for (int x = 0;x<10;x++)              //add samples together
{zzz=zzz + co2now[x];    }
co2raw = zzz/10;                      //divide samples by 10
co2ppm = co2raw - co2Zero;        //get calculated ppm
Serial.print(co2ppm);                // prints the value read
delay(50);}
```

5.4 PROGRAMMING ULTRASONIC SENSOR

The ultrasonic sensor is interfaced with the Raspberry Pi as shown in Figure 5.5. The ultrasonic sensor works on 5V whereas the operating voltage of the Raspberry Pi is 3.3V. Therefore the sensor value of the Echo pin pass-through a logic converter, when the sensor value is connected to the High Voltage pins and it outputs the signal with the lower voltage on the other side which communicates with the GPIO pins in the Raspberry Pi. As shown in the code snippet below a function is written with a functionality to return a binary number whenever the door is closed. This is determined when the distance between the sensor and the door is lesser than the given threshold value.

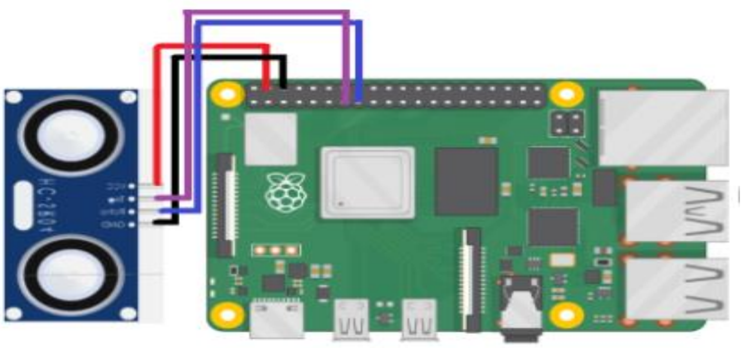

Figure 5.5 Interfacing Ultrasonic sensor with Raspberry Pi

```
def distance (self):
    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)
    StartTime = time.time()
    StopTime = time.time()
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()
    TimeElapsed = StopTime - StartTime
    distance = (TimeElapsed * 34300) / 2
    return distance


def call(self):
    while True:
        self.dist = self.distance()
        print ("Measured Distance = %.1f cm" % self.dist)
        if self.dist < 0.7:
            self.ul = 1
            print("Door is Closed")
            break
```

```
        time.sleep(1)
    return self.ul
```

The above-mentioned code snippets are the functions that calculate the door and opening using the ultrasonic sensor

5.5 ACTUATION OF THE SERVO MOTOR

The actuator is interfaced directly with the Raspberry Pi as shown in Figure5.6. The working of the servo motor depends on the parameters which are provided by other sensors. The positive terminal of the servo motor is connected to 5V, the negative terminal is grounded with Raspberry Pi, and the Pulse Width Modulation (PWM) is connected to GPIO pin 11. A channel is created with PWM on the servo pin with a frequency of 50Hz. After successfully interfacing the actuator with the microcontroller, an algorithm is framed to calculate the degree of rotation for each duty cycle. The duty cycle is the amount of time a digital signal is in the active state relative to the period of the signal. The range of duty cycle which the servo motor works are between 2 and 12. The degree of rotation mapped along with the duty cycle ranges from 0 degree to 180 degrees. The algorithm calculates 18 degrees of increase in the rotation for the consecutive duty cycle.

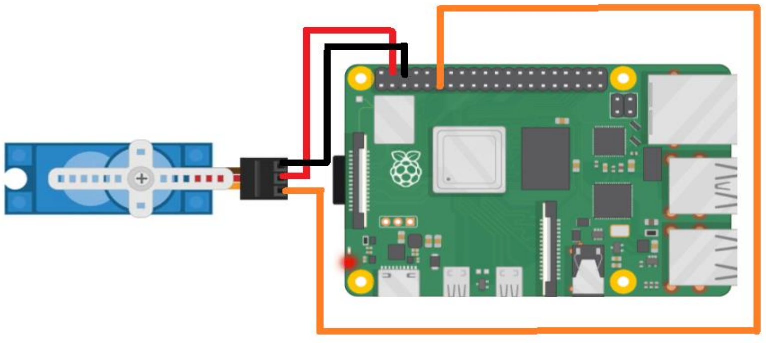

Figure 5.6 Interfacing Servo Motor with Raspberry Pi

```
def motor(duty):
    if duty <= 7:
        servo1.ChangeDutyCycle(duty)
        time.sleep(1)
  def flap():
    if co2_ >= 1000:
        if passenger_count == 0:
            print("No passenger detected \nFlap is closed")
        elif passenger_count == 1:
            print("One passenger detected \nFlap Opening for 18 degree")
            duty = 3
            motor(duty)
        elif passenger_count == 2:
            print("Two passenger detected \nFlap Opening for 36 degree")
            duty = 4
            motor(duty)
        elif passenger_count == 3:
            print("Three passenger detected \nFlap Opening for 54 degree")
            duty = 5
            motor(duty)
        elif passenger_count == 4:
            print("Four passenger detected \nFlap Opening for 72 degree")
```

```
        duty = 6
        motor(duty)
    elif passenger_count == 5:
        print("Five passenger detected \nFlap Opening for 90 degree")
        duty = 7
        motor(duty)
    else:
        print("Passenger Overloading")
```

The above code snippet is the function that controls the flaps with the value from the camera module

CHAPTER 6

SUMMARY

6.1. CONCLUSION

This project has achieved to maintain the optimal CO2 level within the car cabin with the help of a microcontroller and various sensors. It involves determining the number of passengers inside the cabin and then maintaining their optimal CO2 level by using the formulas and values for the respective sensors. According to the values obtained from all the components, the flap which is connected to the servo motor opens and closes to maintain the optimal CO2 level inside the cabin. This project can be used in all modern day automobile vehicles where there is a need for further improvement.

6.2. LIMITATIONS

- In modern day automobiles, automatic air temperature control feature is used to maintain the desired temperature by using the flaps. While adding this project to this existing feature, the flap needs to work on two different parameters, one is to control the desired temperature and another is to maintain the optimal CO2 level within the cabin of the car. Now, this is not possible for the flap to work on two different parameters.
- The microcontroller needs a constant high power supply throughout the working of this project. Due to this different sensors gets heated up quickly.
- The MQ135-Air quality gas sensor is used in this project which has an inbuilt heater to warm up air near the sensitive part for oxidation or reduction to take place .It has been advised not to use this with a small battery source as it will quickly drain your battery.

- The economically effective sensors used in this project cannot achieve higher precision and accuracy which are achieved by manufactured sensors which is used in real time vehicles. The manufactured sensors were not used as they are expensive and their complexity in interfacing is high.
- The sensors used in this project will be exposed to atmospheric air, due to this the sensor gets heated while working and it is exposed to dust particles which might affect the sensitivity of the sensors in a prolonged usage.

6.3. SHORTCOMINGS

- Initially this project was planned to use four gas sensors and ultrasonic sensors. Due to the pandemic, procurement of all four gas sensors couldn’t be achieved due to the unavailability. Therefore a single CO2 sensor and ultrasonic sensor is used in this project.
- Due to the pandemic, the mounting of sensors in a real world scenario was complicated, however this project shows successful results for all the tests conducted by creating a real time scenario inside a closed environment.
- It was planned to design and manufacture a 3D model for the flap as shown in Figure 6.1 and Figure 6.2. However the manufacturing of this prototype couldn’t be achieved due to the pandemic.

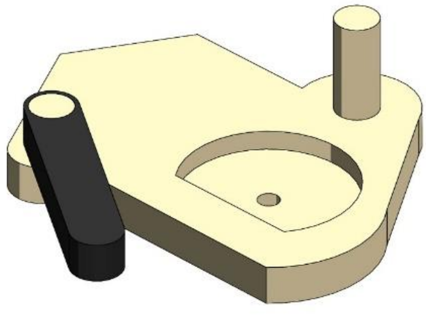

Figure 6.1 3-D Model of Flap

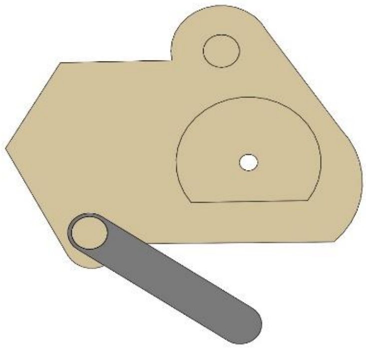

Figure 6.2 Top view of Flap

6.4. FUTURE SCOPE

This project can be implemented with the existing automatic air temperature control feature to control both the air temperature as well as optimal oxygen level inside the car cabin. The currently used gas sensor can detect various harmful gases, but it is programmed to detect only CO2 for this project. Hence if this gas sensor is programmed accordingly it can also detect various other harmful gases like NOx, NH3, methane, benzene etc. for effectively maintaining precise optimal oxygen level inside the cabin of a car. In a real time scenario the components used in this project can be mounted at an appropriate location to provide further precision.

CHAPTER 7

REFERENCE

1. Milos Fojtlin "Airflow Measurement Of The Car HVAC Unit Using Hot-wire Anemometry".EPJ web of conferences by EDP SCIENCE (2016) Pg. 2-6
https://doi.org/10.1051/epjconf/201611402023 .

2. Jan Fiser "Effect of car speed on amount of air supplied by ventilation system to the space of car cabin". EPJ web of conferences by EDP SCIENCES(2014) Pg. 2-3
http://dx.doi.org/10.1051/epjconf/20146702027

3. Maytat Luangprasert "In-vehicle carbon dioxide concentration in commuting cars". Journal of the Air & Waste Management Association(2017) Pg. 6-7
https://doi.org/10.1080/10962247.2016.1268983

4. R A Angelova,"Accumulation of metabolic carbon dioxide CO2 in a vehicle cabin".IOP Conf. Series: Materials Science And Engineering, London(2019) Pg. 4-5
https://iopscience.iop.org/article/10.1088/1757-899X/664/1/012010

APPENDIX A

Main python code

```
import time
import math
import cv2
import numpy as np
from scipy import stats
from ard_mer import ard_mer
from ultra_sonic import ultra_sonic
def face_dec():
    global i, a
    i = 0
    a = []
    face_cascade = cv2.CascadeClassifier('/home/pi/opencv/data/haarcascades/haarcascade_frontalface_default.xml')
    cap = cv2.VideoCapture(0)
    while True:
        ret, frame = cap.read()
        frame=cv2.flip(frame, 1)
```

```
        frame=cv2.resize(frame, (0,0), fx=0.2, fy=0.2)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)


        i = 0
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 5)
            #Increment iterator for each faces in faces
            i = i + 1
            #Display the box and faces
            cv2.putText(frame, 'face num' + str(i), (x-10, y-10),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
            a.append(i)
        cv2.imshow('Test', frame)
        k = cv2.waitKey(1) & 0xff
        if k==27:
            break
    cap.release()
    cv2.destroyAllWindows()


def passenger_count():
    global passenger_count
```

```
    m = stats.mode(a)
    passenger_count = int(m[0])
    #return ("Total Passenger Count: ", passenger_count)

def servo():
    # import lib
    import RPi.GPIO as GPIO
    # Setiing GPIOP numbering mode
    GPIO.setmode(GPIO.BOARD)
    # Set pin 11 as an output , and set servo1 as pin 11 as PWM
    GPIO.setup(11, GPIO.OUT)
    servo1 = GPIO.PWM(11, 50) # Note 11 is pin, 50=50Hz pluse
    #start PWM running, but with value of 0 (pulse off)
    servo1.start(0)
    #print("Starting")
    time.sleep(2)

    def motor(duty):
        if duty <= 7:
            servo1.ChangeDutyCycle(duty)
            time.sleep(1)
```

```
def flap():
    if co2_ >= 1000:
        if passenger_count == 0:
            print("No passenger detected \nFlap is closed")
        elif passenger_count == 1:
            print("One passenger detected \nFlap Opening for 18 degree")
            duty = 3
            motor(duty)
        elif passenger_count == 2:
            print("Two passenger detected \nFlap Opening for 36 degree")
            duty = 4
            motor(duty)
        elif passenger_count == 3:
            print("Three passenger detected \nFlap Opening for 54 degree")
            duty = 5
            motor(duty)
        elif passenger_count == 4:
            print("Four passenger detected \nFlap Opening for 72 degree")
            duty = 6
            motor(duty)
        elif passenger_count == 5:
            print("Five passenger detected \nFlap Opening for 90 degree")
```

```
            duty = 7
            motor(duty)
        else:
            print("Passenger Overloading")
flap()
print('\n')
print('Waiting for fresh air to fill in')
time.sleep(2)
max = ard_mer()
co2ppm = max.call()
print('\n')
print('Air refilled')
print('\n')
print("New CO2 level is {} ppm".format(co2ppm))
time.sleep(2)
# Turn back to 0 degrees
print('\n')
print("CO2 under Optimal level")
print("Flap Closing..")
servo1.ChangeDutyCycle(2)
time.sleep(0.5)
servo1.ChangeDutyCycle(0)
```

```
    # Clean things up at the end
    servo1.stop()
    GPIO.cleanup()
    #print("Success..")


print('Starting...System')
print('\n')
print('Starting...Ultrasonic scan')
print('\n')


uls = ultra_sonic()
ul = uls.call()
print("UL:",ul)
if ul == 1:
    print('Opening Camera..')
    face_dec()
    print('\n')
    print('Detecting the passengers count..')
    print('\n')
    passenger_count()
    print('Detected the passengers count Successlly..')
    print('\n')
```

```
print("Starting CO2 Sensor..")

print('\n')

max = ard_mer()

co2ppm = max.call()

print("CO2:{} ppm ".format(co2ppm))

print('\n')

print("Assuming CO2 level is above optimal level for test case")

co2_ = 1000

print('\n')

print("Starting Servo Motor")

print('\n')

servo()

print('\n')

print('Cycle Completed Successfully..')
```

APPENDIX B

Python code for Ultrasonic Sensor

```
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BOARD)

GPIO_TRIGGER = 13

GPIO_ECHO = 15

GPIO.setup(GPIO_TRIGGER, GPIO.OUT)

GPIO.setup(GPIO_ECHO, GPIO.IN)

GPIO.setwarnings(False)

class ultra_sonic():

    def __init__(self):

        self.ul = 0

    def distance(self):

        # set Trigger to HIGH

        GPIO.output(GPIO_TRIGGER, True)

        # set Trigger after 0.01ms to LOW

        time.sleep(0.00001)

        GPIO.output(GPIO_TRIGGER, False)

        StartTime = time.time()

        StopTime = time.time()

        # save StartTime
```

```
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()
    # save time of arrival
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()
    # time difference between start and arrival
    TimeElapsed = StopTime - StartTime
    # multiply with the sonic speed (34300 cm/s)
    # and divide by 2, because there and back
    distance = (TimeElapsed * 34300) / 2
    return distance

def call(self):
    while True:
        self.dist = self.distance()
        print ("Measured Distance = %.1f cm" % self.dist)
        if self.dist < 0.7:
            self.ul = 1
            print("Door is Closed")
            break
        time.sleep(1)
    return self.ul
```

APPENDIX C

Python code for Interfacing Arduino with Raspberry Pi

```
import serial
class ard_mer():
    def __init__(self, i = 0, max = 0, ppm =0):
        self.i = i
        self.max = max
        self.ppm = ppm
    def call(self):
        ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
        ser.flush()
        while self.i <= 10:
            read_serial = ser.readline().decode('utf-8').rstrip()
            #print(read_serial)
            if len(read_serial) > 0:
                self.ppm = int(read_serial)
                if self.max < self.ppm:
                    self.max = self.ppm
            self.i = self.i+1
        return self.max
```

APPENDIX D

Arduino code for MQ135 Sensor

```
#define anInput     A0
#define co2Zero     0
void setup()
{
  pinMode(anInput,INPUT);
  Serial.begin(9600);
}
void loop()
{
  int co2now[10];
  int co2raw = 0;
  int co2ppm = 0;
  int zzz = 0;
  for (int x = 0;x<10;x++)
  {
    co2now[x]=analogRead(A0);
    delay(200);
  }
```

```
  for (int x = 0;x<10;x++)
  {
    zzz=zzz + co2now[x];
  }
  co2raw = zzz/10;
  co2ppm = co2raw - co2Zero;
  Serial.print(co2ppm);
  delay(50);
}
```