



HAND GESTURE CONTROLLED MACHINES

Submitted by

Dua Tanveer

Sara Sajid

Asad Mughal

Hamza Khurshid

Faculty of Computer Science, Military College of Signals National University of Sciences and
Technology, Rawalpindi in partial fulfillment for the requirements of a B.E Degree in Computer
Software Engineering

ABSTRACT

With the discovery of electronics, it has become an essential part of everyone's life. In the recent times, control of electronic devices such as smartphones, robots and machinery have grabbed the attention of many people. People have started to move towards automation where electrical machines can take decision on its own. However, in some cases a volunteer is required to control the robot/smartphone/machine instead of relying on a piece of code. Hence, the appearing advancement in technology has developed a need for natural user interface very much required. To fulfill this need, we have come up with the idea of Hand Gesture Controlled Machines. Controlling machines via Hand Gesture Recognition will be implemented to improve the interaction between human and machines. The aim of this Hand Glove (consisting of a raspberry pi kit, flex sensors and HC-12) is to sense the movement of the fingers and control the movement of the machine according to its movement wirelessly. The reason to do so is that movement of fingers are natural which compose a gesture, so it does not require any extra effort or learning for the user to interact. The idea behind the project is to develop a device that will control a machine by doing some gestures. For this purpose, flex sensors were mounted at every fingers. The bend in a finger causes a change in its resistance. This signals are detected by the raspberry pi which passes a command to the machine which in turns causes a movement in it. All the processes are done in a controlled environment.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

It is certified that work present in the thesis –Hand Gesture Controlled Machines is completed by Dua Tanveer, Sara Sajid, Asad Mughal and Hamza Khurshid under supervision of Maám Aimen Aakif for partial satisfaction of Degree of Bachelor of Computer Software Engineering is right and affirmed. It is fully ample, in scope and excellence, for the degree of Bachelor of Computer Software Engineering from Military College of Signals, National University of Sciences and Technology (NUST). It is original with 14% plagiarism.

Approved By:

Signature:

**Asst. Prof Aimen Aakif
(Supervisor)**

This page is left intentionally blank.

DEDICATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

ACKNOWLEDGEMENT

We are thankful to ALLAH, who has given us this opportunity and empowered us to achieve this undertaking. We are likewise appreciative of our folks and family and well-wishers for supporting us, motivating us in the time of COVID. Thanks for their basic audits. We would like to thank our supervisor Maam Aimen Aakif, for her consistent support and inspiration throughout our venture. This would not have been possible without her efforts.

This page is left intentionally blank.

Contents

1. Introduction	11
1.1 Problem Statement	11
1.2 Approach	11
1.3 Objectives	11
1.4 Scope	12
1.5 Deliverables	12
1.6 Document convention	12
1.7 Intended Audience	13
1.7.1 Project Supervisor	13
1.7.2 BESE 23 and BETE 54 FYP group (developers, testers, and documentation writers) ...	13
1.7.3 UG Project Evaluation Team	14
1.7.4 Reading suggestions	14
2. Literature Review	15
2.1 Introduction:	15
2.2 Vision based hand gesture recognition technique:	15
2.3 RGB and RGB-D sensors for real time hand gesture recognition:	16
2.4 Hand Gestures Based on Instrumented Glove Approach:	17
2.5 Modern Glove based approach:	17
2.6 Hand Gesture Recognition with Skin Detection and Deep Learning Method	19
2.7 Microsoft Kinect Technology	20
2.8 Results:	21
2.9 Conclusion:	21
3. Software Req. Specification (SRS)	22
3.1 Introduction	22
3.2 Purpose	22
3.3 Overall Description	22
3.3.1 Product Perspective	22
3.4 Product Functions	23
3.4.1 User Classes and Characteristics	23
3.4.2 Operating Environment	24
3.4.3 Design and Implementation Constraints	24
3.5 External Interface Requirements	24
3.6 Software Interfaces	25
3.7 Functional Requirements	25
3.7.1 Use Case Diagram	25
3.8 Other Nonfunctional Requirements	32
3.8.1 Performance Requirements	32

3.8.2	Safety Requirements.....	32
3.9	Software Quality Attributes	32
3.9.1	Usability.....	32
3.9.2	Legal	32
3.9.3	Reliability.....	32
4.	Design and Specifications	33
4.1	INTRODUCTION.....	33
4.2	Scope.....	33
4.3	System Overview.....	33
4.4	Work Breakdown Structure	34
4.5	System Architecture.....	34
4.6	Decomposition Description.....	36
4.7	Design Rationale	37
4.8	DATA DESIGN	38
4.8.1	Data Description.....	38
4.9	COMPONENT DESIGN	38
4.9.1	Overview of modules/components	38
4.9.2	Input Component.....	38
4.9.3	Application Component	39
4.9.4	Output Component.....	39
4.10	Activity Diagram:.....	39
4.11	Sequence Diagram	40
4.12	Human interface design.....	40
4.12.1	Overview of user interface.....	40
4.12.2	Screen images	41
4.13	Requirements Matrix	42
5.	Methodology.....	43
5.1	Why not NRF module?	46
5.2	AT SENDER SIDE:	51
5.3	How we are interfacing:.....	51
5.4	Code:	53
5.5	HC12.....	56
5.6	UART Protocol.....	57
5.7	RC car Arduino	59
5.8	Final Diagram:	60
6.	Results and Analyses.....	61
7.	Future work and Recommendations	66
8.	Bibliography	70

8.1	Gesture Recognition (Pang, Ismail, & Gilbert, 2010)	70
-----	---	----

Table of Figures

Figure 1	Classifications method	16
Figure 2	vision based recognition technique	17
Figure 3	Sensor based data glove	18
Figure 4	adapted from jimaging-06-00073%20(3).pdf	18
Figure 5	Gesture Recognition System Based on RGB video	20
Figure 6	flowchart HGCM	23
Figure 7	Use case diagram	25
Figure 8	Arduino based RC car	35
Figure 9	Sender (Glove)	36
Figure 10	Receiver side	37
Figure 11	Activity diagram	39
Figure 12	Sequence Diagram	40
Figure 13		43
Figure 14	Sender side	44
Figure 15	Receiver side	45
Figure 16	interfacing	52
Figure 17	H-12 configuration	57
Figure 18	RC car	Error! Bookmark not defined.
Figure 19	receiver side code	61
Figure 20	thresholds set in if -else statements	63
Figure 21	thresholds set	63
Figure 22	notice channel values for gesture 4 meeting the if conditions	64
Figure 23	Hand Glove	59

Chapter 1

1. Introduction

The main idea behind the project is to provide a hand glove to people so that they can wear it and control a device with only a small amount of instruction. A flex sensor is a sensor used to measure the bending of the finger. When flex sensor bends, it results in a change of resistance, which can be detected by the raspberry pi. We will be using flex sensors with raspberry pi to detect hand gestures.

1.1 Problem Statement

The technique used in hand gesture recognition up till now is mainly visual based hand gesture recognition. It has a lot of complexities and is not cost effective at all. Our aim is to develop a cost effective, with less complexities and safe for the users' hand glove that can control a device without touching it.

1.2 Approach

To create a cost-effective hand glove, we will be using HC12 module which is a half- duplex wireless serial communication. The hand glove that we are building recognize gestures and perform a specific action according to it. Flex sensors detect the movement of the fingers and pass the data to the raspberry pi. An ADC is attached in between this two as Raspberry pi only reads digital input. Raspberry pi perform computation and with the different inputs values recognizes which gesture has been implemented. A specific threshold is set to differentiate movements of the fingers. Each gesture is mapped by a specific command, which is forwarded to the transceiver. This command is then transmitted to the machine.

1.3 Objectives

The aim of this HGCM is to remove the barrier between human natural gestures and devices. The main objective of HGCM is to make a cost-effective and safe to use hand glove using wireless communication between two raspberry pi.

1.4 Scope

As the world is moving towards automation, where electronic devices now can think of their own. However sometimes we need a human to control robot/smartphone/machine instead of relying on a piece of code.

1.5 Deliverables

Table 1 Deliverables.

Sr	Tasks	Deliverables
1	Literature review	Literature survey
2	Requirements gathering	SRS Document
3	Application Design	SDS Document
4	Implementation	Live demo

1.6 Document convention

Heading is focused on in a numbered design, the most noteworthy need heading having a solitary digit and ensuing headings having more numbers, per their level.

Every one of the primary headings are named as follows: single-digit number followed by a dab and the name of the segment (All strong Times New Roman, size 18, Centred).

The entire second-level subheadings for each subsection have a similar number as their individual principal heading, trailed by one dab and resulting subheading number followed by name of the subsection (All striking Times New Roman, size 16).

Further subheadings, i.e., level three and underneath, adhere to similar standards as above for numbering and naming, however extraordinary for the text style (All the further sub-headings are of size 14 and bold.)

1.7 Intended Audience

The intended audiences for the Hand Gesture Controlled machine include the project supervisor, the BESE 23 and BETE 54 FYP group (developers), the UG project evaluation team, and other persons at MCS CSE Department and EE Department

1.7.1 Project Supervisor

- It will help the supervisor to supervise the project and guide the team in a better way. This document will be used by her to check whether all the requirements have been understood and, in the end, whether the requirements have been properly implemented or not.

1.7.2 BESE 23 and BETE 54 FYP group (developers, testers, and documentation writers)

- For FYP group members, this document will provide the guideline for developing and testing the project.

1.7.3 UG Project Evaluation Team

- It will help the evaluation team to evaluate the progress of the FYP project. The document will provide the evaluators with the scope, requirements, and details of the project to be built. It will also be used as a basis for the evaluation of the implementation and final project.

1.7.4 Reading suggestions

- The SRS begins with the title and table of contents. All level 1 and level 2 headings are given in the table of contents, but the lower subheadings are not included. Each main heading is succeeded by several subheadings, which are all in bold format.

CHAPTER 2

2. Literature Review

2.1 Introduction:

Hand gesture recognition is a way to communicate with devices using hand gestures. It is an alternate to a user interface which is more enjoyable and convenient. A user has to simply use natural gestures to control a device. In this, you do not need to type anything to perform an action to do a particular task. Currently the traditional input devices are keyboards, mice and touch input. Hand gestures can also be used as an input. For example, in Sign language translation, virtual reality applications, smart homes, smart TVs, vending machines and Virtual instore displays etc. Covid-19 also motivated us to work on HMI to deal with pandemic like situation in future.

In a study by Watson et al, participants were divided into two groups. The first group used touch input for a task and the other group used a mouse instead. The results showed that participants using touch input enjoyed more than the people using mouse for the input. Also touch input's performance was better in terms of speed and accuracy. (Watson, Hancock, Birk and Mandryk 2013)

In a research by Cao and Vronay [1], it was observed that gesture-controlled presentations were not only enjoyed by the presenters itself but also the audience.

HGCM uses two raspberry pi zero w which are wirelessly communicated (half-duplex) using HC 12. The flex sensors or bend sensors measure the deflection, the resistance is varied when bending the sensor. These flex sensors are attached to the fingers in HGCM. The data is sent to the HC12. HC12 forwards the signal to the second HC12 attached with second pi. The raspberry pi reads the command and pass output to GPIO pin. This command will then be showed by the RC car being operated with Arduino.

2.2 Vision based hand gesture recognition technique:

The method includes three major steps detection, tracking and recognition. According to a research by [2] Lei Shi, Yangsheng Wang and Jituo Li, they trained the hand detector using AdaBoost Algorithm with HOG features. Then the tracker tracks the hand contour. Using hidden Markov models, they recognized the gestures. The system proposed has potential in many applications. But a major drawback of many VGR techniques is that it has complexities when it comes to extracting gestures from video sequences. VGR

techniques are expensive and require high power. Implementation of real time VGR-based recognition system is difficult.

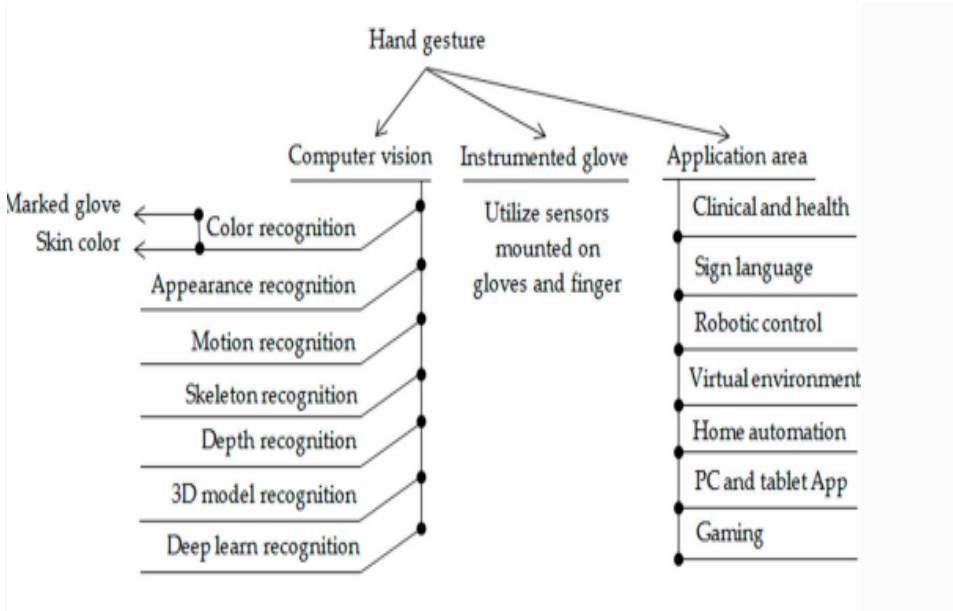


Figure 1 Classifications method conducted by this review (adapted from jimaging-06-00073%20(3).pdf)

2.3 RGB and RGB-D sensors for real time hand gesture recognition:

Yuan Yao (Shanghai University) and Fan Zhang (Zhejiang University) used RGB-D sensors for real time hand gesture recognition. They proposed a procedure for capturing hand gesture using real gesture data set. They designed a hand partition scheme for color-based semi-automatic labeling. This method was integrated with a framework (for development of desktop applications). This hand contour model was used to recognize and match gestures in 3D space for complex real time interactions. With the help of Kinect sensors, more accurate tracking was achieved in desktop environment. RGB sensor (Red, green, blue) assists camera in capturing in a way that it tells the amount of light required to produce a well exposed image. TOF (Time of flight) camera uses infrared light, lasers invisible to human, to determine depth info. Thermal stereo vision is beneficial for partial autonomous systems by making them more reliable and safer.

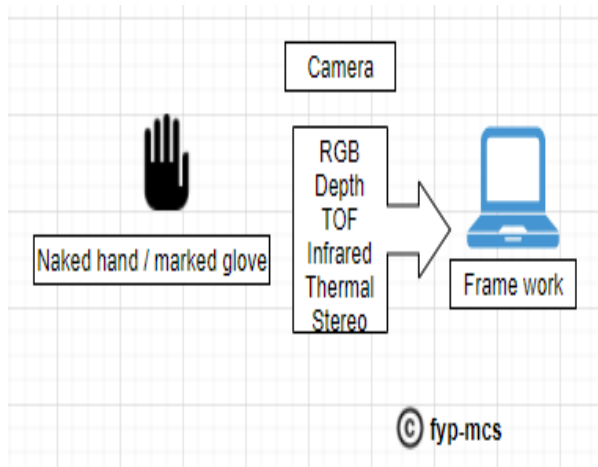


Figure 2 vision based recognition technique

2.4 Hand Gestures Based on Instrumented Glove Approach:

The wearable glove-based sensors are used to detect hand motion and its position. They also provide exact coordinates of the fingers. This is possible by orientations and configurations by sensors used in the glove. But the restriction in this approach is that the user must be connected to the computer physically. It affects the easiness of the user. Then again, the price of the devices used is high.

2.5 Modern Glove based approach:

The innovation of touch is being utilized in present day glove approach. It is viewed as Industrial-grade haptic innovation. The glove for this situation gives haptic input. Because of which client can detect the shape, surface, development and weight of a virtual article by utilizing microfluidic innovation. The wearable glove-based sensors can be utilized to catch hand movement and position. Furthermore, they can without much of a stretch give the specific directions of palm and finger areas, direction, and arrangements by utilizing sensors connected to the gloves. Notwithstanding, this approach requires the client to be associated with the computer truly, which hinders the simplicity of cooperation among client and computer. Also, the cost of these gadgets is very high . Be that as it may, the advanced glove-based approach utilizes the innovation of touch, which really encouraging innovation and it is viewed as Industrial-grade haptic innovation. Where the glove gives haptic input that bodes well the shape, surface, development, and weight of a virtual item by

utilizing microfluidic innovation. Figure 3 shows an illustration of a sensor glove utilized in gesture-based communication. Hand Gestures Based on Computer Vision Approach The camera vision based sensor is a typical, reasonable and appropriate procedure since it gives contactless correspondence among people and computers. Various designs of cameras can be used, for example, monocular, fisheye, TOF and IR. Be that as it may, this strategy includes a few difficulties, including lighting variety, foundation issues, the impact of impediments, complex foundation, preparing time exchanged against goal and casing rate and forefront or foundation objects introducing a similar skin shading tone or in any case showing up as hands.

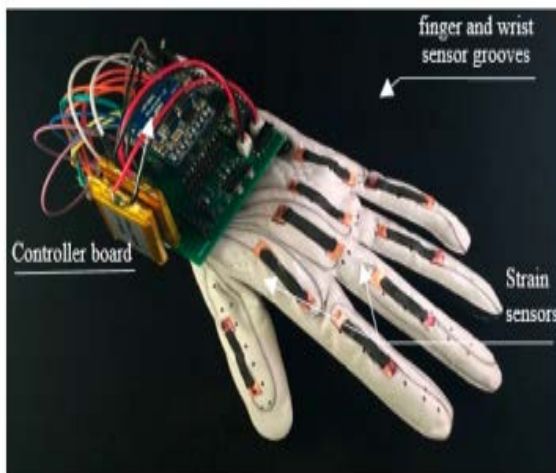


Figure 3 Sensor based data glove (adapted from website: <https://physicsworld.com/a/smart-glove-translates-sign-language-into-di>)

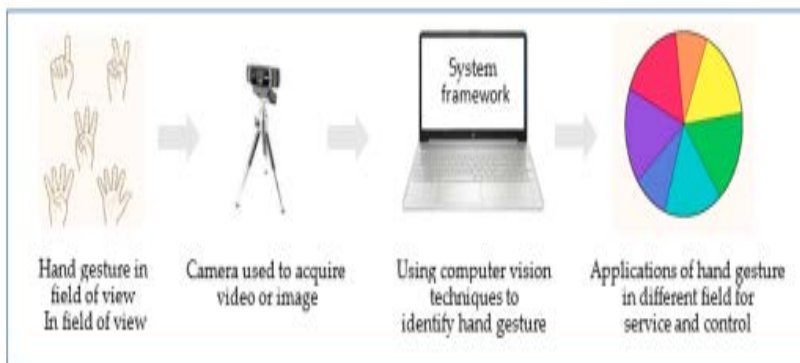


Figure 4 adapted from [jimaging-06-00073%20\(3\).pdf](http://jimaging-06-00073%20(3).pdf)

2.6 Hand Gesture Recognition with Skin Detection and Deep Learning Method

This framework has three principal parts in the initial segment the framework catches video of the client then it completes limit division, and it is changed over into a double video. Some preprocessing strategies like development and consumption are required every one of the shapes of hand depends on a few records. Pyramidal pooling module component is utilized to perceive motion. The stream graph of occasions is given underneath.

There has been incredible accentuation on Human-Computer-Interaction examination to make simple to-utilize interfaces by straightforwardly utilizing normal correspondence and control abilities of humans. As a significant piece of the body, perceiving hand motion is vital for Human-Computer-Interaction. In later a long time, there has been an enormous measure of examination available motion acknowledgment. While there is various exploration centered around this subject, there are as yet a few issues to be settled. The speed and exactness are two fundamental attributes of the calculation; hence, a powerful and quick strategy is expected to improve client encounters. A contour-based technique for perceiving hand signal utilizing profundity picture information is displayed in. Utilizing profundity information, this technique can recognize the hand from foundation effectively without getting confounded by foundation tone. Notwithstanding, profundity information are not normal and effectively accessible while RGB information tackles the issue. In a progressive technique for static hand signal acknowledgment that consolidates finger identification and histogram of arranged slope (HOG) highlight is proposed. A calculation applied for finding. Fingertips close by district extricated by Bayesian guideline-based skin shading division is proposed in. In the consistent motion acknowledgment issue is handled with a two streams Recurrent Neural Networks (2S-RNN) for the RGB-D information input. These strategies all get great impact, yet they are not so productive. This paper gives a more effective path dependent on profound learning.

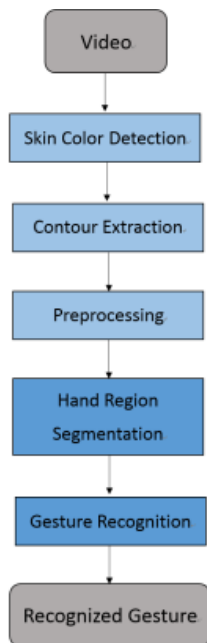


Figure 5 Gesture Recognition System Based on RGB video

2.7 Microsoft Kinect Technology

Kinect (codenamed Project Natal during progress) is a line of development recognizing input devices made by Microsoft and first conveyed in 2010. The advancement fuses a lot of gear at first made by PrimeSense, joining RGB cameras, infrared projectors and identifiers that arranged significance through either coordinated light or period of flight assessments, and a mouthpiece show, close by programming and man-made cognizance from Microsoft to allow the contraption to perform progressing movement affirmation, talk affirmation and body skeletal area for up to four people, among various limits. This engages Kinect to be used as a without hands ordinary UI contraption to speak with a PC structure. Kinect is a periphery that sits on the customer's exhibit like a webcam. Microsoft is driving the blame for Kinect, a sign affirmation stage that grants individuals to talk with PCs totally through talking and motioning. Kinect gives PCs, "eyes, ears, and a frontal cortex." There a few unique parts in the space, for instance, Soft Kinect, Gesture Tek, Point Grab, Eyesight and PrimeSense, an Israeli association acquired by Apple. Emerging advancements from associations, for instance, Eyesight goes far past

gaming to think about another level of little motor precision and significance knowledge.

2.8 Results:

The above method provided good results but have few limitations that makes it unsuitable. The sensors being used in this technique make a skin damage or severe burns. The sensors itself are quite expensive. The quality of camera matters a lot, it should have a frame rate that is enough to capture hand movements accurately. For high resolution cameras, you will have to invest a lot. Skin color and hand size variations could affect the results. Changing light conditions can cause ambiguity in calculating depth. It is hard for the camera for tracking motion, if a person rapidly moves hand/glove, and can cause it to be blur.

2.9 Conclusion:

HGCM is a cost-effective system. The sensors used are totally safe for the user. It is not sensitive to light so results will not be varied. There is less complexity as there is no need of high resolution of cameras. It has no major calculations. Whereas in computer vision methods, you need to detect hands using different types of cameras. Algorithms are needed for the purpose. The algorithms are meant to detect hand features such as skin color, appearance, motion, depth, 3D model, deep learn detection etc. These methods make it complex and quite challenging. Sensor based recognition is more effective.

Chapter 3

3. Software Req. Specification (SRS)

3.1 Introduction

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references, and overview of the SRS. The aim of this document is to present a detailed description of the project Hand Gesture Controlled Machine which aims to develop a hand glove which will control the motion of any device. The detailed requirements of this project are provided in this document.

3.2 Purpose

The SRS covers the requirements (SRS) for HGCM. Controlling machines through Hand Gesture Recognition will be carried out to improve the communication between human and machines. The point of this Hand Glove (comprising of a raspberry pi unit, flex sensors and HC-12 module) control a device/gadget without touching it. The purpose for it being that human hand gestures are natural. For instance, controlling cranes with hand gestures. It will reduce the training cost of crane operator and provides safety in complex operating environments.

4. Overall Description

4.1 Product Perspective

The main idea behind the project is to provide a hand glove to people so that they can wear it and control the car with only a small amount of instruction. A flex sensor mounted on the index finger is used to sense the curvature of the finger. Bending the sensor results in a change of resistance, which can be detected by the raspberry pi. The distance between the transistor and receiver (which is RC car in this case) should not be too large and there should be no obstruction between the transmitter and receiver.

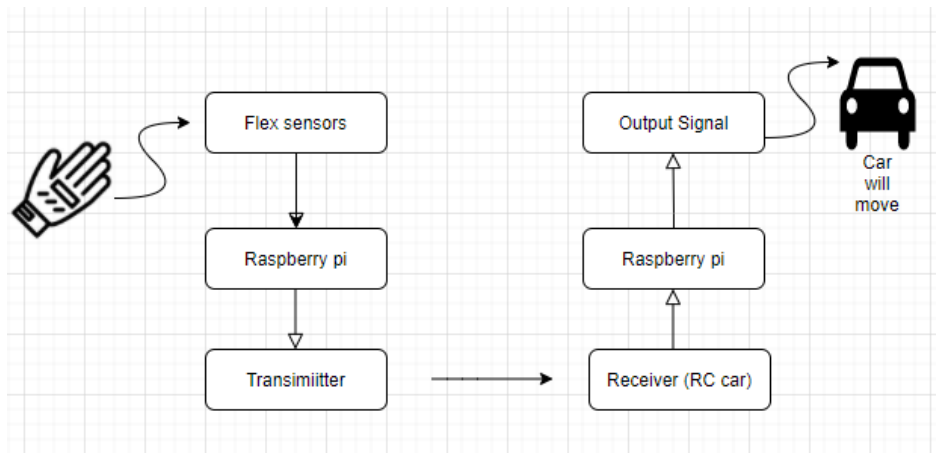


Figure 6 flowchart HGCM

In the above figure, it is shown that flex sensors attached in the glove send a signal to raspberry pi when they are bent. The raspberry pi on processing the data sends it to the HC 12 module which communicates wirelessly with the second raspberry pi. The second raspberry pi receive the output on a specified GPIO pin.

4.2 Product Functions

4.2.1 User Classes and Characteristics

With hand gesture glove you can handle your robot with movement gestures for simple gymnastic deceives, and control. Maestro Gesture Glove permits clients of any expertise level more precise and stable control of UAVs. The Hand Gesture Glove gives the customer a more pleasant and "effortlessness of control" understanding while at the same time working a PC. On the off chance that you from time to time use web, overview or change reports, and peruse your email, the Air-Keyboard and Air-Mouse feature will give a 3-D unprecedented control knowledge. Game players can even program their own custom movements to make modified and life-like experiences that are reasonable with all PC games. Glove can fill in as a regulator, control centre, mouse, or other gaming contraption for more careful and dynamic control. The officials can give centres in scoring clearly to the scoreboard while utilizing the Hand Gesture Glove with Sports the chief's circumstance on the right or left hand or both!

This undertaking is an optimal base for the engineers who wish to manage the hand glove and make changes in the current errand to extend it to various applications, including hardware and programming. This venture can help the auto business which contains a wide extent of associations and affiliations drew in with the arrangement, progression, amassing, publicizing, and selling of motor vehicles.

4.2.2 Operating Environment

The environment will be controlled there will not be any obstacle between the glove and RC Car. Also, both the glove and RC car will be in specific range to each other, and the gestures outputs will be specific.

4.2.3 Design and Implementation Constraints

The following design and implement constraints must be kept in mind. Operating voltage of raspberry pie should not exceed 5V. The distance between the hand glove and end device has its own limitations due to transmitting range of the sensors.

4.3 External Interface Requirements

Flex sensors are thin strips of film-like material, whose resistance changes with bending. We use 9 of these to measure the bending of the important finger joints. HC12 module allows half duplex serial communication. It will detect obstacles around the machine. If object comes close to the RC car, it will stop.

4.4 Software Interfaces

This Raspberry Pi in hand glove helps you to control the appliances using our palm movement, by placing 5 flex sensors in the glove wore in our hand, based on bending the flex sensor value gets varied. According to the convergence robotic arm which connected to the GPIO pins of the Raspberry Pi. The application can be anything like controlling appliances based on stretching the fingers and closing the fingers for each application. The raspberry pi code on the microcontroller transmits the sensor data, after minimal processing, over a serial port to the computer. The date is picked up by using the library in Python.

4.5 Functional Requirements

4.5.1 Use Case Diagram

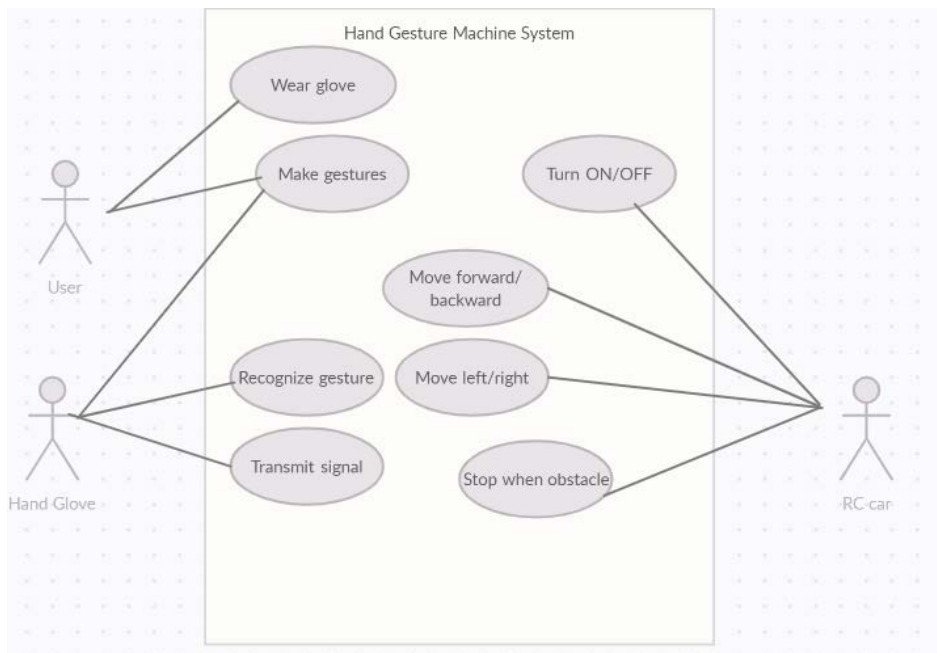


Figure 7 Use case diagram

USE CASE # 1
Description: The RC car will turn on a specific gesture.
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: RC car will be in static position meaning it will be off. LED will be turned off.
Main Flow: 1. The user will wear the hand glove 2. The user will make specified gesture to turn on the machine (RC car) 3. The LED on RC car will turn on 4. the use case ends
Postconditions: RC car is ready to move
Alternative Flows: Connect RC car directly with raspberry pi

USE CASE # 2
Description: The RC car will turn off on a specific gesture.
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: RC car will be in static position.
Main Flow: 1. The RC car is either moving/ stationary. 2. The user will make specified gesture to stop on the RC car from moving (if it was already moving) 3. The user will make gesture to turn off the machine. 3. The LED on RC car will turn off 4. the use case ends
Postconditions: RC car is not ready to receive signals
Alternative Flows: Connect RC car directly with raspberry pi

USE CASE # 3
Description: The RC car will move forward
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: RC car will be in static position.
Main Flow: 1. The user will wear the hand glove . 2. The user will make specified gesture using hand glove 3. The RC car will move forward. 4. the use case ends
Postconditions: RC car moves forward
Alternative Flows: Connect RC car directly with raspberry pi

USE CASE # 4
Description: The RC car will turn left
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: RC car is stationary
Main Flow: <ol style="list-style-type: none">1. The user will wear the hand glove2. The user will make specified gesture with hand glove3. The RC car will turn left4. the use case ends
Postconditions: RC car turns left
Alternative Flows: Connect RC car directly with raspberry pi

USE CASE # 5
Description: The RC car will turn right
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: The RC car is stationary
Main Flow: 1. The user will wear the hand glove 2. The user will make specified gesture with hand glove 3. The RC car will turn right 4. the use case ends
Postconditions: RC car turns right
Alternative Flows: Connect RC car directly with raspberry pi

USE CASE # 6
Description: The RC car will reverse
Primary Actors: User Hand glove
Secondary Actors: RC car
Preconditions: The RC car is stationary
Main Flow: 1. The user will wear the hand glove 2. The user will make specified gesture with hand glove 3. The RC car will reverse 4. the use case ends
Postconditions: RC car reverse
Alternative Flows: Connect RC car directly with raspberry pi

4.6 Other Nonfunctional Requirements

4.6.1 Performance Requirements

Accuracy: a threshold will be set for a specific gesture so that the sensor can recognize the movement of finger.

Delay: there will be about 5 second delay between the input and the output.

Distance: there is a specified range between transmitter and the receiver. Also, as we are working in controlled environment so there will be no obstacle.

4.6.2 Safety Requirements

The glove should not encounter water or else it gives you a shock. Sensors must not be damaged.

4.7 Software Quality Attributes

4.7.1 Usability

Our target customers mostly include programmers and users from technical field, but we will still create simple interface which will comply with design interface standards that make the user's interaction as simple and efficient as possible.

4.7.2 Legal

The Hand Gesture Controlled Machine Project should follow customer privacy policy strictly.

4.7.3 Reliability

Our system is mostly depending on hardware. As a result, the reliability of our project highly depends on the reliability of the singular hardware components and their interfaces. The system **is not expected to handle any kind of failure except if batteries** stop working.

Chapter 4

5. Design and Specifications

5.1 INTRODUCTION

This document covers the software design specifications for our project. The purpose of this document is to understand each component and module of the project. It will provide information about the relationship between each module and how they are interconnected. The document is intended to inform stakeholders of the details of the design and the design process. It is meant to outline the features, structure, and architecture of HGCM, to serve as a guide to developers and the intended audience.

5.2 Scope

The aim of this Hand Glove is to dissolve the interaction of human with their device by recognizing the hand movements of the user without touching. The Hand Glove takes advantage of a multitude of sensors to capture finger movements and uses this information to control a device – in this case, a modified RC Car. HGCM in which a hand glove will be programmed in such a way that it will control basic six motions of a RC Car i.e. (switch on, move forward, backward, turn left, right, and switch off).

5.3 System Overview

Fingers in different positions constitute a specific gesture. These gestures have their own meaning and are used on daily purposes. The hand glove that we are building recognize gestures and perform a specific action according to it.

Flex sensors detect the movement of the fingers and pass the data to the raspberry pi. An ADC is attached in between this two as Raspberry pi only reads digital input. Raspberry pi perform computation and with the different inputs values recognizes which gesture has been implemented. A specific threshold is set to differentiate movements of the fingers. Each gesture is mapped by a specific command, which is forwarded to the transceiver. This command is then transmitted to the machine.

5.4 Work Breakdown Structure

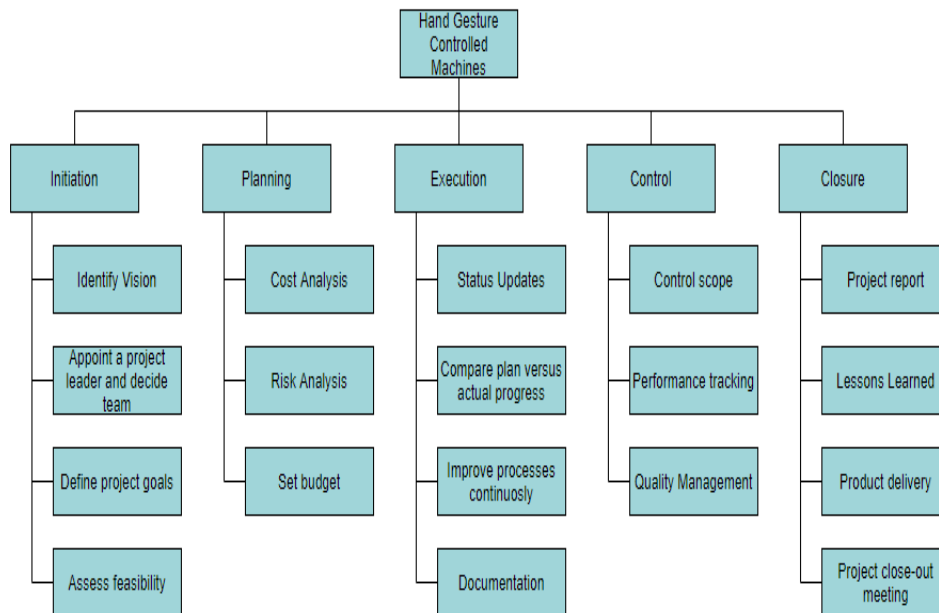


Figure 8 WBS

5.5 System Architecture

HGCM is an embedded system architecture that is a system which we will be integrating hardware system with software that is designed to perform a specified function, either as an independent system or as a part of a large system. We chose raspberry pi zero w which is ideal for an embedded system. Typical embedded system architecture consists of two main parts: embedded hardware and embedded software.

The flex sensors can be viewed as variable resistors whose values are dependent on the degree of deflection of the sensors. Flex sensors detect the movement of the fingers and pass the data to the raspberry pi.

Coding is being done in python on Thonny IDE using library gpiozero. This library supports a number of different pin implementations. One of the pin libraries supported, pigpio, provides the ability to control GPIO pins remotely over the network which means you can use GPIO Zero to control devices connected to Raspberry pi on the network. The code is being written in an event driven style because event-driven programming is in which the flow of the program is determined by events such as user actions, sensor outputs, or message passing from other programs or threads.

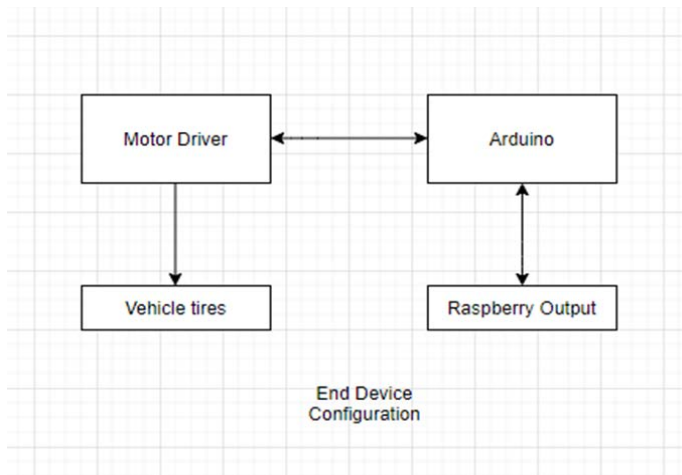


Figure 9 Arduino based RC car.

5.6 Decomposition Description

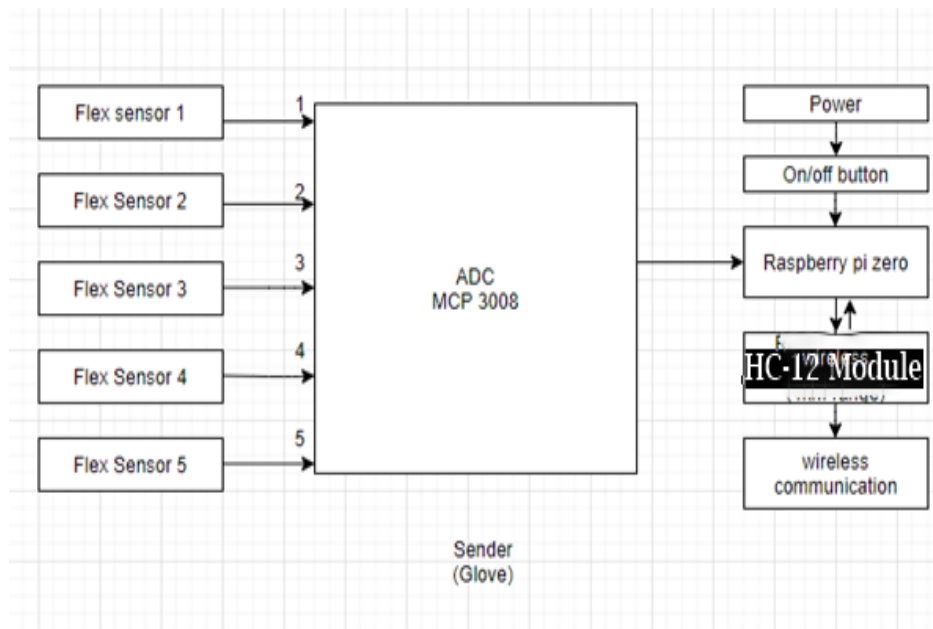


Figure 10 Sender (Glove)

The flex sensors are attached at the glove for every finger. When a finger is bent, it causes a bend in the flex sensor. This bend in flex sensor causes a change in its voltage. This change in voltage is read by mcp3008 and passes all the channel inputs to the raspberry pi. The raspberry pi processes all the voltages and checks for its conditions. When any of the specified condition is fulfilled, it passes a specific command according to the condition to the HC-12 module. HC-12 then wirelessly transfers this command to the receiver.

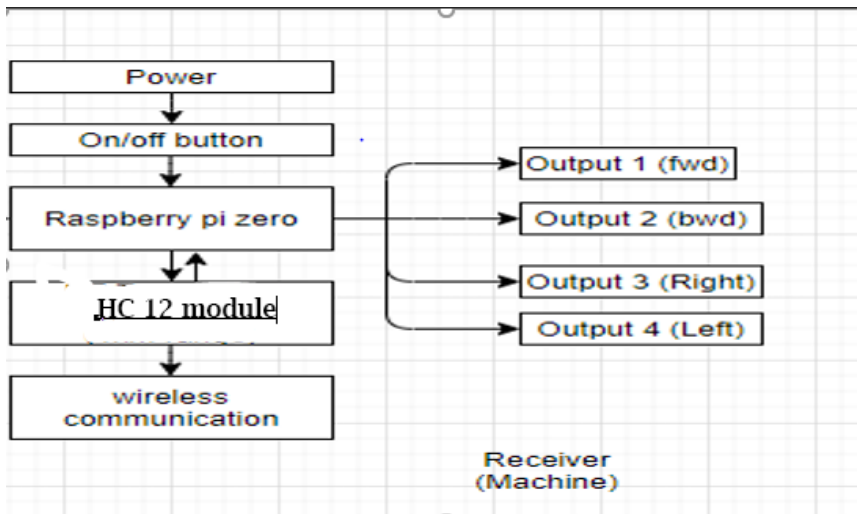


Figure 11 Receiver side

At the receiver side, the HC-12 module, when it receives a command at the specified baud rate, it passes it to the raspberry pi. The raspberry pi processes this command and checks the conditions against it. If a given condition is true, it will make the specific GPIO against it high and the others zero.

5.7 Design Rationale

1. **Performs specific task**
Embedded systems performs some specific function or tasks.
2. **Low Cost**
The price of embedded system is not so expensive.
3. **Time Specific**
It performs the tasks with in a certain time frame.
4. **Low Power**
Embedded Systems don't require much power to operate.
5. **High Efficiency**
The efficiency level of embedded systems are so high.
6. **Minimal User interface**
These systems require less user interface and easy to use.

7. Less Human intervention

These systems require no human intervention or very less human intervention.

5.8 DATA DESIGN

5.8.1 Data Description

This undertaking concerns the improvement of an answer for imagining and putting away various sorts of information utilizing a solitary board PC. To do this, an appropriate single board PC has been chosen, considering cost and execution.

The gathering has picked the Linux appropriation called Raspbian for the venture. This is the authoritatively upheld working framework for the Raspberry Pi. Raspbian is a changed variant of Debian, which is a working framework that utilizes the Linux part. The primary purposes behind utilizing Raspbian is triple: it accompanies most or the entirety of the product bundles required, worked in help for the Raspberry contact screen, and in the gathering's assessment a bigger possibility of long haul support.

The Raspberry Pi **REQUIRES** that you utilize a microSD card, and it loads programming from the card as it boots up.

5.9 COMPONENT DESIGN

Conceptual Architecture is “Context” for the system’s use.

5.9.1 Overview of modules/components

This subsection will introduce the various components and subsystems. To capture the sensory data for the finger gesture recognition, a wireless smart glove equipped with flex sensors will be implemented. Then raspberry pi will compute this data. The transceiver will act as a broker between the glove and end device (RC car). Then RC car will rotate according to receiving data.

5.9.2 Input Component

Input signal will be in the form of mechanical physical movement of fingers which will be converted into electrical signal.

5.9.3 Application Component

This electrical signal will be sent to the raspberry pi. It will process this electrical signal and produce a corresponding Rf signal.

5.9.4 Output Component

This rf signal will be received on the machine and after processing it will give out a specific electrical signal at the corresponding output port.

5.10 Activity Diagram:

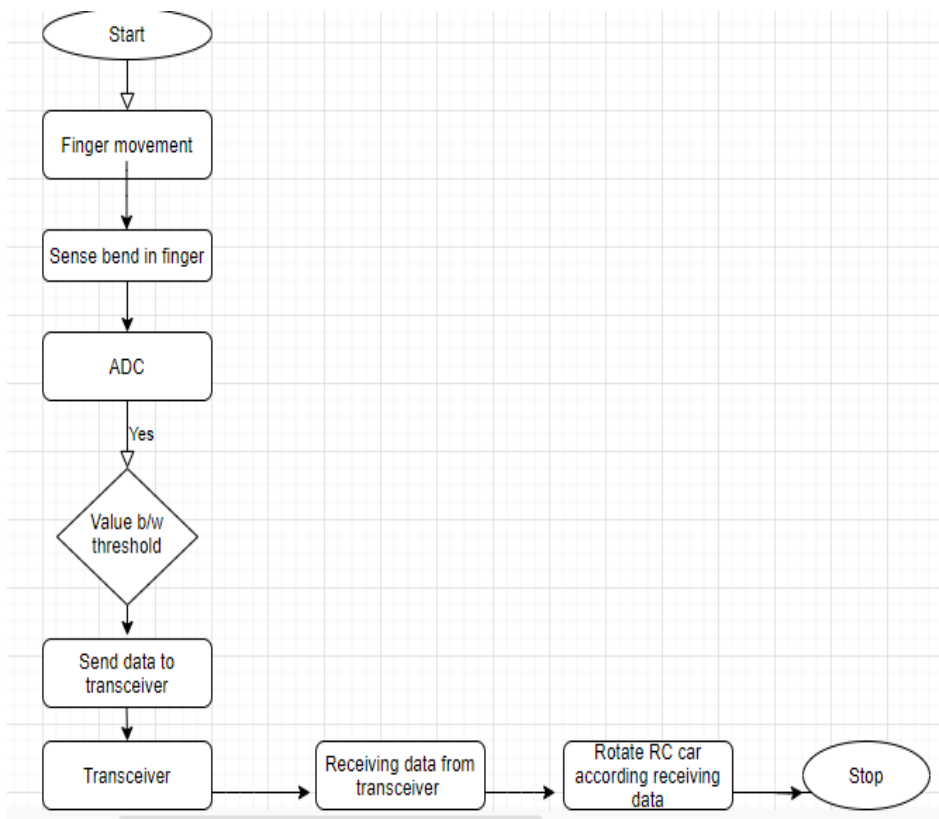


Figure 12 Activity diagram

5.11 Sequence Diagram

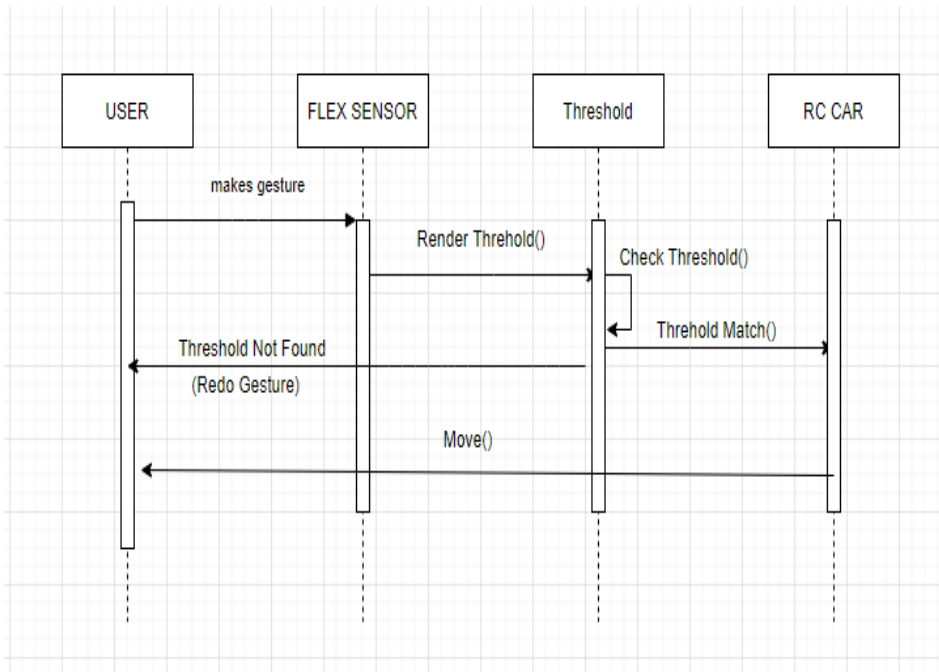


Figure 13 Sequence Diagram.

5.12 Human interface design

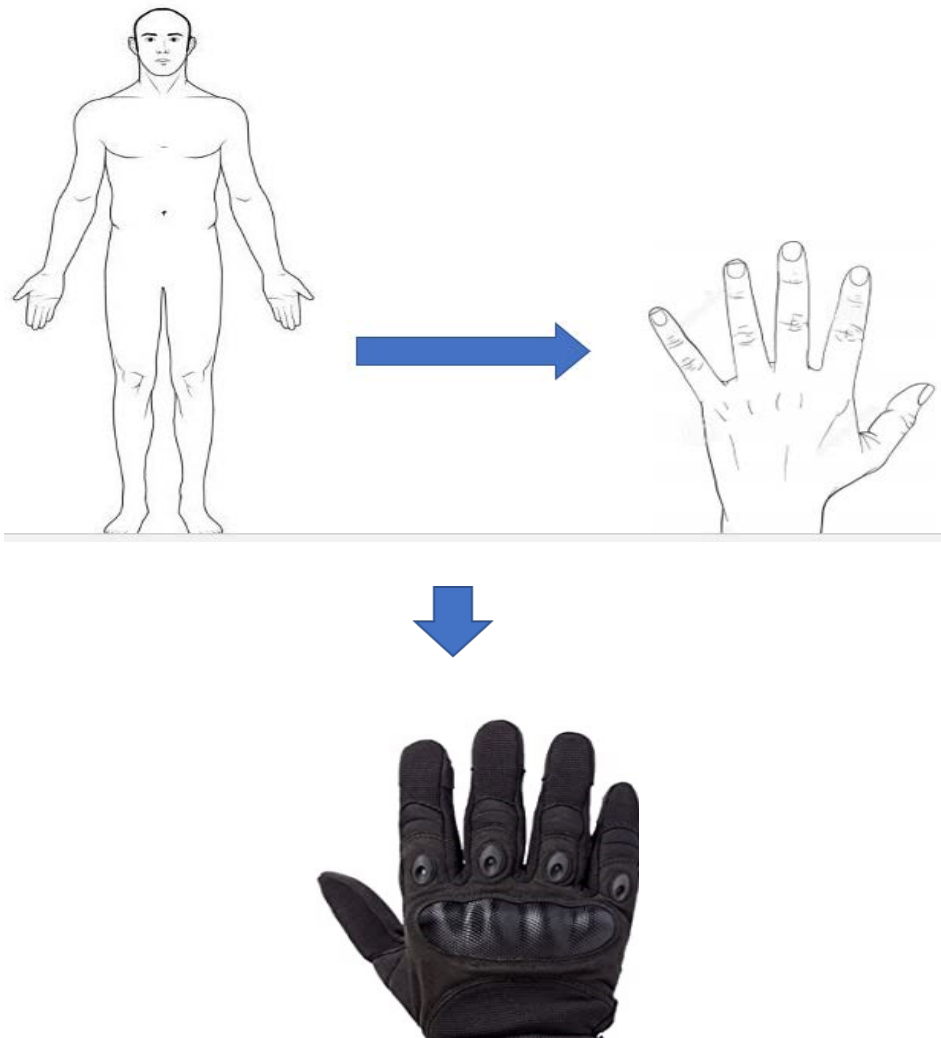
5.12.1 Overview of user interface

In this project, the human user interface is hardware based. Meaning that the point where the user will interact with our device is totally based on hardware. So, to easily understand the user interface design, following points will help a lot.

1. The raspberry pi module will be mounted on the back side of the glove.
2. The side of the glove will have the transmitter and the power source of the raspberry pi.
3. The flex sensors will be fixed along the finger of the glove.

So, in this way, after powering up these components from the raspberry pi, our user interface will be ready to go.

5.12.2 Screen images



5.13 Requirements Matrix

REQUIREMENTS TRACEABILITY MATRIX						
PROJECT MANAGER:	Dua Tanveer			PROJECT SUPERVISOR:		
PROJECT Name:	Hand Gesture Controlled Machine			Maám Aimen Aakif		
REQUIREMENT INFORMATION						
ID	CATEGORY	REQUIREMENT	PRIORITY	COMPONENTS		
REQ-001	Should have	RC Car should move forward on Gesture 1	High	threshold set for gesture 1		
REQ-002	Should have	RC Car should reverse on Gesture 2	High	threshold set for gesture 2		
REQ-003	Should have	RC Car should move left on Gesture 3	High	threshold set for gesture 3		
REQ-004	Should have	RC Car should move right on Gesture 4	High	threshold set for gesture 4		
REQ-005	Might have	ON/OFF signal indicated by LED	Medium	LED attached to glove		
REQ-006	Should have	RC Car should avoid obstacles	Medium	Ultrasonic sensor		

CHAPTER 5

6. Methodology

To make this hand glove, we are using the technique of wirelessly connecting two raspberry pi zero w using HC 12 module. For this we require at least two working Pis. With the addition of wireless LAN and Bluetooth, the Raspberry Pi Zero w, which is perfect for embedded Internet of Things (IoT) projects. The Pi Zero is designed to be flexible and compact as possible with mini connectors and an unpopulated 40-pin GPIO connector, allowing you to use just what you need for your project. To create a cost-effective hand glove, we will be using HC12 module which is a half- duplex wireless serial communication. The hand glove that we are building recognize gestures and perform a specific action according to it. Flex sensors detect the movement of the fingers and pass the data to the raspberry pi. An ADC is attached in between this two as Raspberry pi only reads digital input. Raspberry pi perform computation and with the different inputs values recognizes which gesture has been implemented. A specific threshold is set to differentiate movements of the fingers. Each gesture is mapped by a specific command, which is forwarded to the transceiver. This command is then transmitted to the machine.

GPIO full form is general-purpose input/output. It is a digital signal pin on an integrated circuit or electronic circuit board which can be programmed using a software. It is used to perform digital input and output functions. User can control the input output at runtime.

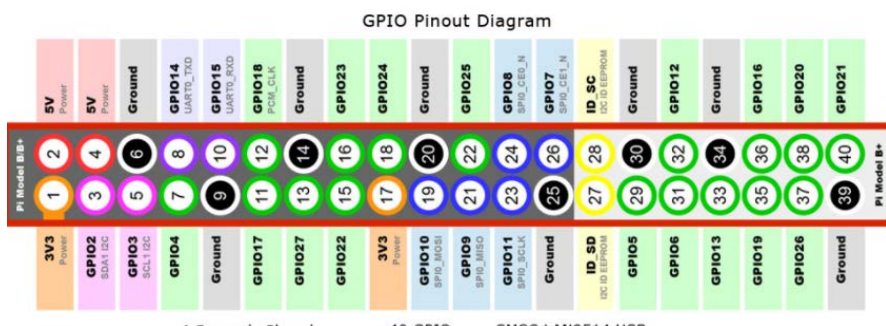


Figure 14 adapted from <https://dev.drun.net/2017/05/10/reading-hc-12-from-rpi-and-sending-to-domoticz/>.

Naturally, GPIOs have no predefined reason. By plan it has no predefined rationale and can be utilized by the equipment or programming engineer to play out the assignments they require. GPIO pins can be arranged for various purposes and can work with a few kinds of parts. GPIO pins go about as switches. At the point when the yield is set to HIGH, it gives 3.3 volts and no voltage when set to LOW. GPIO pins can be constrained by utilizing a product program to run any gadget. Each of the 17 GPIO pins are computerized. They can give high and low levels yield or read high and low levels yield. It is appropriate for sensors that give advanced contribution to Pi in any case it is anything but so useful for simple sensors. A GPIO yield, it is anything but an all-inclusive pin, which is the worth of one of the two voltage boundaries (high and low), and the conduct can be modified by utilizing the add or eliminate programs. A GPIO port is a gadget characterized gathering of GPIO pins (regularly at least 4 pins).

The flowcharts below show the fundamental model of the proposed approach.

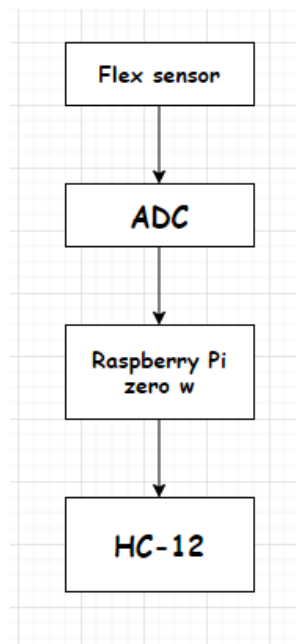


Figure 15 Sender side

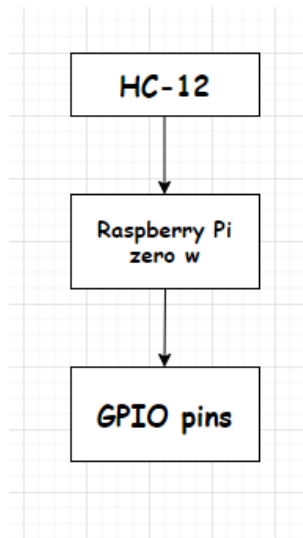


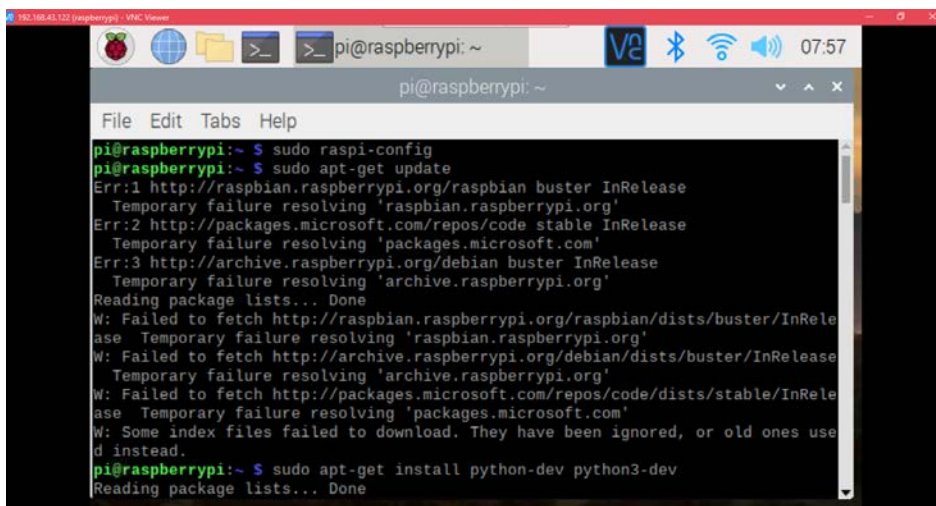
Figure 16 Receiver side

To create a cost-effective hand glove, we will be using HC12 module which is a half- duplex wireless serial communication. The hand glove that we are building recognize gestures and perform a specific action according to it. Flex sensors detect the movement of the fingers and pass the data to the raspberry pi. An ADC is attached in between this two as Raspberry pi only reads digital input. Raspberry pi perform computation and with the different inputs values recognizes which gesture has been implemented. A specific threshold is set to differentiate movements of the fingers. Each gesture is mapped by a specific command, which is forwarded to the transceiver. This command is then transmitted to the machine.

6.1 Why not NRF module?

Initially we decided to use NRF module to connect raspberries wirelessly, but we were unable to do so. We installed the required libraries from GitHub. We thought that the error is in the code. The code we wrote had no syntax errors. But the channel in use had noise. In return acknowledgment we were getting 128 128 128... which is not correct. It meant that sender and receiver sides are not connected. To remove the noise, we tried using capacitor of 10 μ F. It still gave us the same error.

After detailed study we found out that nrf24l01 in real time was not a preferred options for wireless communication. It was liable to errors or noise. We decided to search for another option. Hence HC-12 was found. NRF is full duplex due to which we are receiving noise at both sides. Whereas HC-12 is 1/2 duplex serial communication that can either send or receive at a moment resulting in no noise. NRF had 1km range and HC-12 had 500 m range. Communications depends on baud rate and HC-12 has these. The lesser the baud rate, the more the range. We can customize its range.



```
pi@raspberrypi:~$ sudo raspi-config
pi@raspberrypi:~$ sudo apt-get update
Err:1 http://raspbian.raspberrypi.org/raspbian buster InRelease
  Temporary failure resolving 'raspbian.raspberrypi.org'
Err:2 http://packages.microsoft.com/repos/code stable InRelease
  Temporary failure resolving 'packages.microsoft.com'
Err:3 http://archive.raspberrypi.org/debian buster InRelease
  Temporary failure resolving 'archive.raspberrypi.org'
Reading package lists... Done
W: Failed to fetch http://raspbian.raspberrypi.org/raspbian/dists/buster/InRelease
  Temporary failure resolving 'raspbian.raspberrypi.org'
W: Failed to fetch http://archive.raspberrypi.org/debian/dists/buster/InRelease
  Temporary failure resolving 'archive.raspberrypi.org'
W: Failed to fetch http://packages.microsoft.com/repos/code/dists/stable/InRelease
  Temporary failure resolving 'packages.microsoft.com'
W: Some index files failed to download. They have been ignored, or old ones used instead.
pi@raspberrypi:~$ sudo apt-get install python-dev python3-dev
Reading package lists... Done
```

```
W: Failed to fetch http://packages.microsoft.com/repos/code/dists/stable/InRe
ase Temporary failure resolving 'packages.microsoft.com'
W: Some index files failed to download. They have been ignored, or old ones u
d instead.
pi@raspberrypi:~ $ sudo apt-get install python-dev python3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-dev is already the newest version (3.7.3-1).
python3-dev set to manually installed.
The following additional packages will be installed:
  libpython-dev libpython-stdlib libpython2-dev libpython2-stdlib
  libpython2.7 libpython2.7-dev python python-minimal python2 python2-dev
  python2-minimal python2.7-dev
Suggested packages:
  python-doc python-tk python2-doc
```

```
pi@raspberrypi:~ $ wget https://github.com/Gadgetoid/py-spidev/archive/master.z
ip
--2021-04-29 07:53:15-- https://github.com/Gadgetoid/py-spidev/archive/master.
zip
Resolving github.com (github.com)... 13.234.210.38
Connecting to github.com (github.com)|13.234.210.38|:443... connected.
HTTP request sent, awaiting response...
302 Found
Location: https://codeload.github.com/Gadgetoid/py-spidev/zip/master [following
]
--2021-04-29 07:53:16-- https://codeload.github.com/Gadgetoid/py-spidev/zip/ma
ster
Resolving codeload.github.com (codeload.github.com)... 13.127.152.42
Connecting to codeload.github.com (codeload.github.com)|13.127.152.42|:443... c
onected.
```

```
192.168.43.122 (raspberrypi) - VNC Viewer
pi@raspberrypi: ~
File Edit Tabs Help
Location: https://codeload.github.com/Gadgetoid/py-spidev/zip/master [following
]
--2021-04-29 07:53:16-- https://codeload.github.com/Gadgetoid/py-spidev/zip/ma
ster
Resolving codeload.github.com (codeload.github.com)... 13.127.152.42
Connecting to codeload.github.com (codeload.github.com)|13.127.152.42|:443... c
onected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'master.zip'

master.zip      [ <=>          ] 21.66K  111KB/s  in 0.2s

2021-04-29 07:53:18 (111 KB/s) - 'master.zip' saved [22178]

pi@raspberrypi:~ $
pi@raspberrypi:~ $ ls
Adafruit_Python_MCP3008  dd.py  Documents  master.zip  Pictures  Templates
Bookshelf                Desktop  Downloads  Music       Public    Videos
```

```
122 [raspberrypi] - VNC Viewer
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ ls
Adafruit_Python_MCP3008  dd.py  Documents  master.zip  Pictures  Templates
Bookshelf                Desktop Downloads  Music       Public     Videos
pi@raspberrypi:~ $ unzip master.zip
Archive:  master.zip
7412eeb871ba3a73975d27e500890724218ded33
  creating: py-spidev-master/
  inflating: py-spidev-master/.gitignore
  inflating: py-spidev-master/CHANGELOG.md
  inflating: py-spidev-master/LICENSE
  inflating: py-spidev-master/LICENSE.md
  inflating: py-spidev-master/MANIFEST.in
  inflating: py-spidev-master/Makefile
  inflating: py-spidev-master/README.md
  extracting: py-spidev-master/setup.cfg
  inflating: py-spidev-master/setup.py
  inflating: py-spidev-master/spidev_module.c
pi@raspberrypi:~ $ ls
Adafruit_Python_MCP3008  Desktop  master.zip  Public  Videos
```

```
pi@raspberrypi:~ $ cd py-spidev-masters
bash: cd: py-spidev-masters: No such file or directory
pi@raspberrypi:~ $ cd py-spidev-master
pi@raspberrypi:~/py-spidev-master $ sudo python3 setup.py install
running install
running bdist_egg
running egg_info
creating spidev.egg-info
writing spidev.egg-info/PKG-INFO
writing dependency_links to spidev.egg-info/dependency_links.txt
writing top-level names to spidev.egg-info/top_level.txt
writing manifest file 'spidev.egg-info/SOURCES.txt'
reading manifest file 'spidev.egg-info/SOURCES.txt'
reading manifest template 'MANIFEST.in'
writing manifest file 'spidev.egg-info/SOURCES.txt'
installing library code to build/bdist.linux-armv6l/egg
running install_lib
running build_ext
building 'spidev' extension
```



```
pi@raspberrypi: ~/De...
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
creating build/temp.linux-armv6l-3.7
arm-linux-gnueabi-gcc -pthread -DNDEBUG -g -fwrapv -O2 -Wall -g -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 -fPIC -I/usr/include/python3.7m -c spidev_module.c -o build/temp.linux-armv6l-3.7/spidev_module.o
creating build/lib.linux-armv6l-3.7
arm-linux-gnueabi-gcc -pthread -shared -Wl,-O1 -Wl,-Bsymbolic-functions -Wl,-z,relro -Wl,-z,relro -g -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 build/temp.linux-armv6l-3.7/spidev_module.o -o build/lib.linux-armv6l-3.7/spidev.cpython-37m-arm-linux-gnueabi.so
creating build/bdist.linux-armv6l
creating build/bdist.linux-armv6l/egg
copying build/lib.linux-armv6l-3.7/spidev.cpython-37m-arm-linux-gnueabi.so -> build/bdist.linux-armv6l/egg
creating stub loader for spidev.cpython-37m-arm-linux-gnueabi.so
byte-compiling build/bdist.linux-armv6l/egg/spidev.py to spidev.cpython-37.pyc
creating build/bdist.linux-armv6l/egg/EGG-INFO
copying spidev.egg-info/PKG-INFO -> build/bdist.linux-armv6l/egg/EGG-INFO
copying spidev.egg-info/SOURCES.txt -> build/bdist.linux-armv6l/egg/EGG-INFO
```

```
Processing spidev-3.3-py3.7-linux-armv6l.egg
creating /usr/local/lib/python3.7/dist-packages/spidev-3.3-py3.7-linux-armv6l.egg
Extracting spidev-3.3-py3.7-linux-armv6l.egg to /usr/local/lib/python3.7/dist-packages
Adding spidev 3.3 to easy-install.pth file

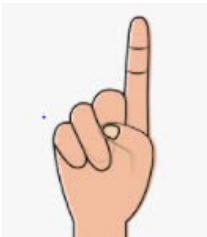
Installed /usr/local/lib/python3.7/dist-packages/spidev-3.3-py3.7-linux-armv6l.egg
Processing dependencies for spidev==3.3
Finished processing dependencies for spidev==3.3
pi@raspberrypi:~/py-spidev-master $ cd
pi@raspberrypi:~ $ cd Desktop/
pi@raspberrypi:~/Desktop $
```

```
pi@raspberrypi: ~/Desktop/NRF24L01
File Edit Tabs Help
Processing spidev-3.3-py3.7-linux-armv6l.egg
creating /usr/local/lib/python3.7/dist-packages/spidev-3.3-py3.7-linux-armv6l.egg
Extracting spidev-3.3-py3.7-linux-armv6l.egg to /usr/local/lib/python3.7/dist-packages
Adding spidev 3.3 to easy-install.pth file

Installed /usr/local/lib/python3.7/dist-packages/spidev-3.3-py3.7-linux-armv6l.egg
Processing dependencies for spidev==3.3
Finished processing dependencies for spidev==3.3
pi@raspberrypi:~/py-spidev-master $ cd
pi@raspberrypi:~ $ cd Desktop/
pi@raspberrypi:~/Desktop $ ls
1
pi@raspberrypi:~/Desktop $ mkdir NRF24L01
pi@raspberrypi:~/Desktop $ cd NRF24L01/
pi@raspberrypi:~/Desktop/NRF24L01 $ ls
pi@raspberrypi:~/Desktop/NRF24L01 $
```

```
pi@raspberrypi:~/Desktop/NRF24L01/lib_nrf24 $ cd ..
pi@raspberrypi:~/Desktop/NRF24L01 $ ls
lib_nrf24  lib_nrf24.py
pi@raspberrypi:~/Desktop/NRF24L01 $ ls
lib_nrf24  lib_nrf24.py
pi@raspberrypi:~/Desktop/NRF24L01 $
```

The above method was used to install the libraries for running NRF module for establishing wireless communication. The following gestures are being used.



Forward



Reverse



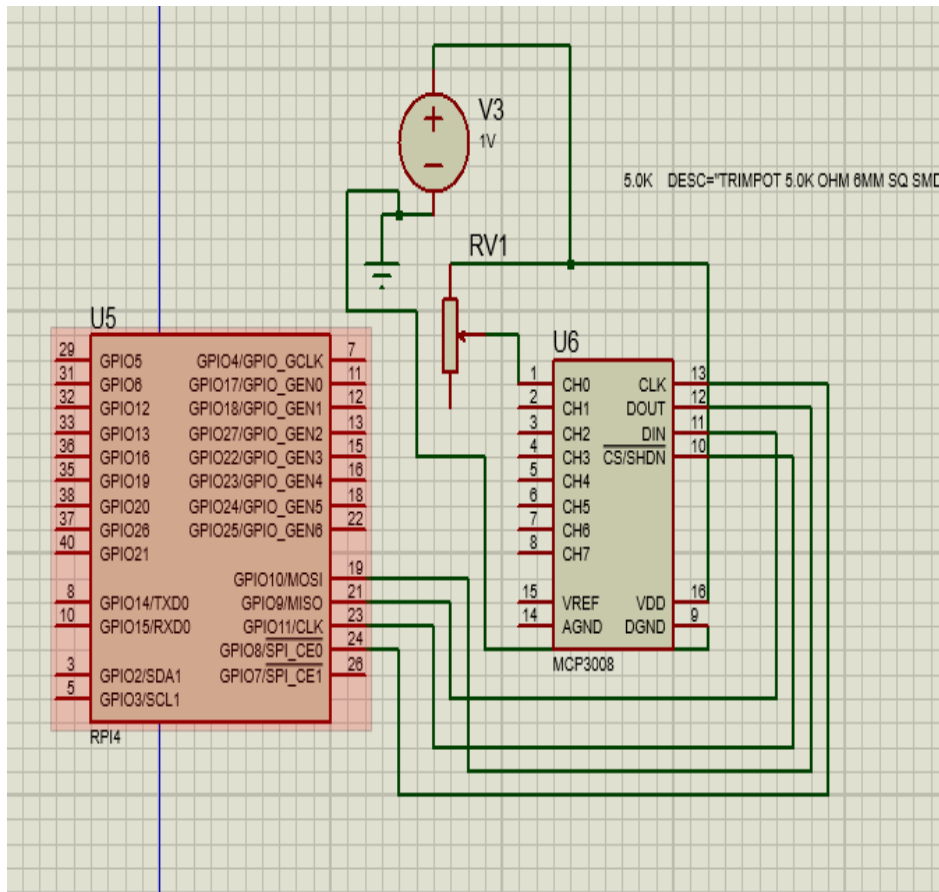
Right



Left

6.2 AT SENDER SIDE:

Flex sensor



6.3 How we are interfacing:

Mcp3008 is an ADC converter. It reads analogue input at its channels into and pass this value/signal to the high-level ports related against its pins. In this circuit diagram, MCP is related with the raspberry pi using SPI consecutive affiliation. SPI is serial communication interface of raspberry pi which enables it to communicate with other peripheral devices. If SPI pins are not available GPIO pins can in like manner be used as opposed to them. To use SPI pins, first they ought to be enabled. The potential gain of using SPI pins is that they have speedier response tan various pins. The DIN and DOUT pins of MCP3008 are

related with the MOSI and MISO pins of the raspberry pi separately. The CS pin of the ADC is related with the CE0 pin and the CLK pin is related with the SCLK pin of the raspberry pi. Next the basic sensor is related at any of the available 8 channels and worth is scrutinized on the redirects in this way.

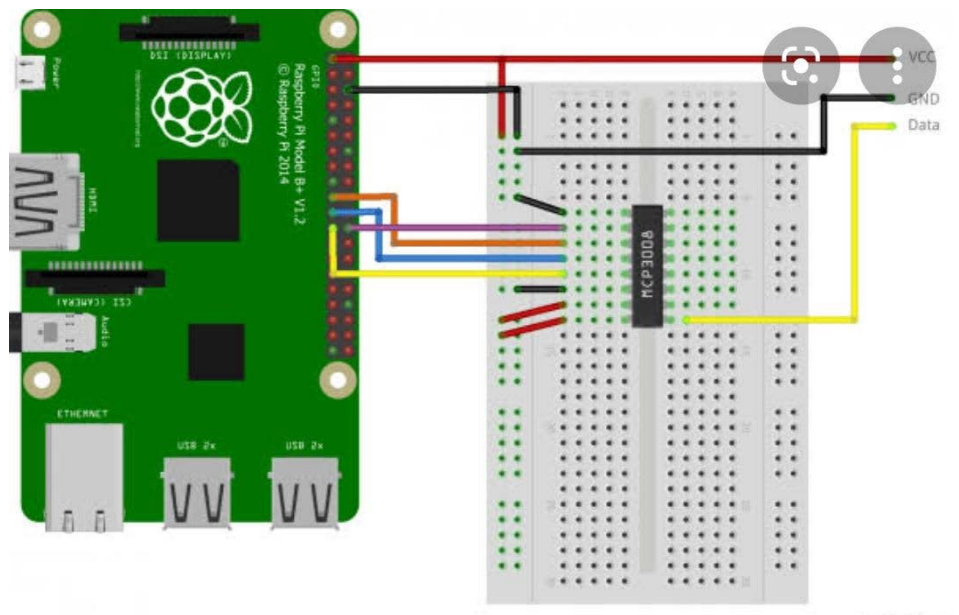


Figure 17 interfacing

6.4 Code:

```
import RPi.GPIO as GPIO
from time import sleep

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)
GPIO.setwarnings(False)

while True:
    GPIO.output(17,0) #left
    GPIO.output(27,0) #right movement
    GPIO.output(23,0) #back
    GPIO.output(24,0) #frwd movement
    sleep (1)
```

We have used GPIO pin (23,17,27 & 24) for producing output.

```
receiver.py ✕
1 import serial
2 import time
3 import RPi.GPIO as GPIO
4 from time import sleep
5 GPIO.setmode(GPIO.BCM)
6 GPIO.setup(23, GPIO.OUT)
7 GPIO.setup(17, GPIO.OUT)
8 GPIO.setup(27, GPIO.OUT)
9 GPIO.setup(24, GPIO.OUT)
10 GPIO.setwarnings(False)
11
12 while(1):
13
14     # ser=serial.Serial('/dev/ttyS0', 9600)
15     ser = serial.Serial ("/dev/ttyS0")

Shell ✕
5
1
2
3
4
```

```
send.py x FYP.py x flex.py x
42     ser.write("3".encode('utf-8'))
43     x=ser.read(9)
44     print(x)
45     ser.close()
46     elif adc.value>=975 and adc_1.value < 830 and adc_
47     ser.write("4".encode('utf-8'))
48     x=ser.read(9)
49     print(x)
50     ser.close()
51     time.sleep(2)
52 #ser = serial.Serial ("/dev/ttyS0")
53 #ser.baudrate = 9600
54 #ser.write("5".encode('utf-8'))
55 #x=ser.read(9)
56 #print(x)
57 #ser.close()
58

Shell x
channel 2 = 701
channel 3 = 729
channel 0 = 960
channel 1 = 750
channel 2 = 755
channel 3 = 543
```

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)
GPIO.setwarnings(False)
while True:
    GPIO.output(17,0) #left
```

```
GPIO.output(27,0) #right movemet
GPIO.output(23,0) #back
GPIO.output(24,0) #frwd movement
sleep(1)
```

6.5 HC12

The HC-12 is a half-duplex wireless serial port correspondence module which has 100 channels (433.4-473.0 MHz band). It is equipped for moving information at up to 1 km. This undertaking has been working with the HC-12 to make a remote association between two raspberry Pi. We are likewise utilizing HC12 module in this strategy. HC-12 is not pricey. It allows 433 MHz remote correspondence. The scope of this module with sequential port is up to 1800 meters in outside. 100dBm, contingent upon the speed. Acknowledges a 3.2-5.5 V force supply and cannot be utilized with both 3.3 V and 5V, 3.3 V gadget voltage gadgets, and dependent on the SI4463 RF chip, has an inherent microcontroller that can be arranged through the AT order, and you can likewise utilize an outside receiving wire that upholds establishment. A recurrence that is isolated into 100 channels, going from 433.4-473.0 MHz to 400 kHz (channel division. The most extreme yield power is 100 Mw (20 dbm), and the collector affectability is - 117 dbm. The HC-12 sequential port backings the following information move rates:

1200bps

2400bps

4800bps

9600bit (default)

19200

38400bps

57600bps

115200 bps

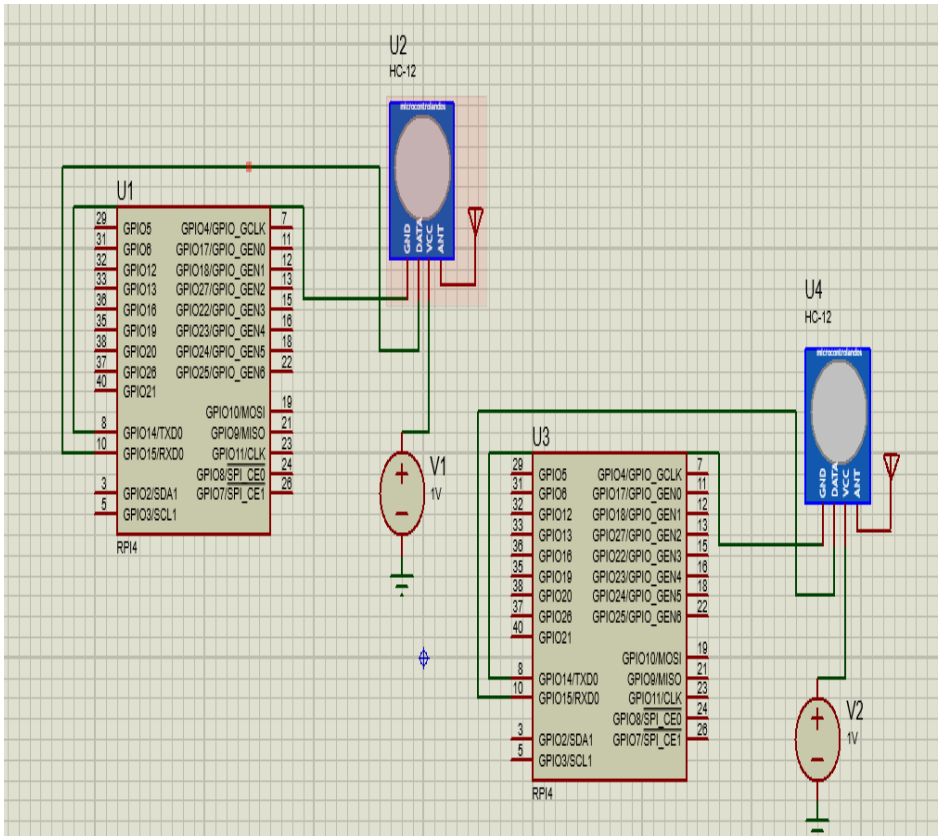


Figure 18 HC-12 configuration

6.6 UART Protocol

As a "general" arrangement, we can design UART to work with a wide range of sorts of sequential conventions. UART was adjusted into single-chip units in the mid-1970s, beginning with Western Digital's WD1402A.

In a UART correspondence plot:

1. One chip's Tx (send) pin interfaces straightforwardly to the next's Rx (get) pin and the other way around. Generally, the transmission will occur at 3.3 or 5V. UART is a solitary expert, single-slave convention, where one gadget is set up to speak with just one accomplice.

2. Information goes to and from a UART in corresponding to the controlling gadget (e.g., a CPU).

3. When sending on the Tx pin, the primary UART makes an interpretation of this equal data into sequential and communicates it to the getting partner.

The second UART gets this information on its Rx pin and changes it back into corresponding to speak with its controlling gadget.

UARTs send information sequentially, in one of three modes:

- Full duplex: Simultaneous correspondence to and from each expert and slave

- Half duplex: Data streams toward each path in turn

- Simplex: One-way correspondence in particular

Information transmission happens as information bundles, starting with a beginning piece, where the normally high line is pulled to ground. After the beginning piece, five to nine information bits communicate in what is known as the bundle's information outline, trailed by a discretionary equality touch to confirm legitimate information transmission. At long last, at least one stop pieces are sent where the line is set to high. This finishes a bundle.

6.7 RC car Arduino

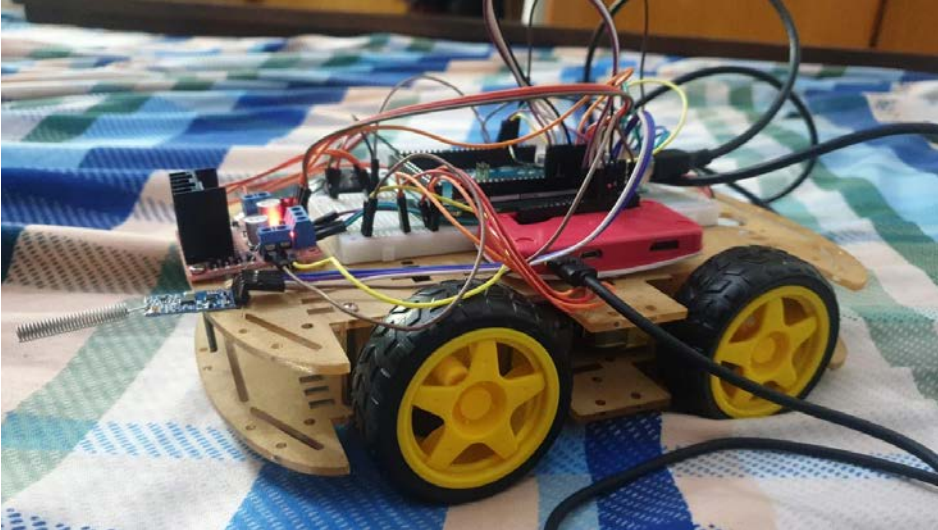


Figure 19 RC car

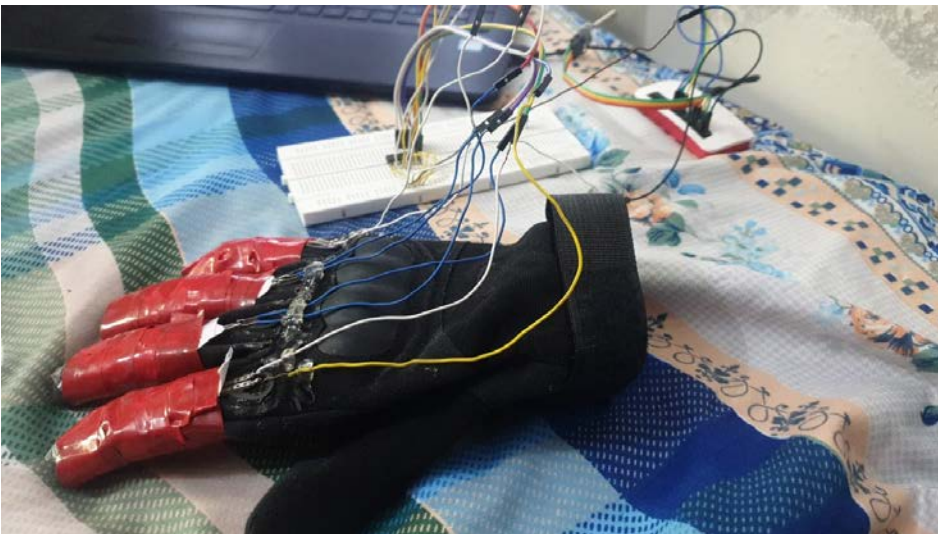
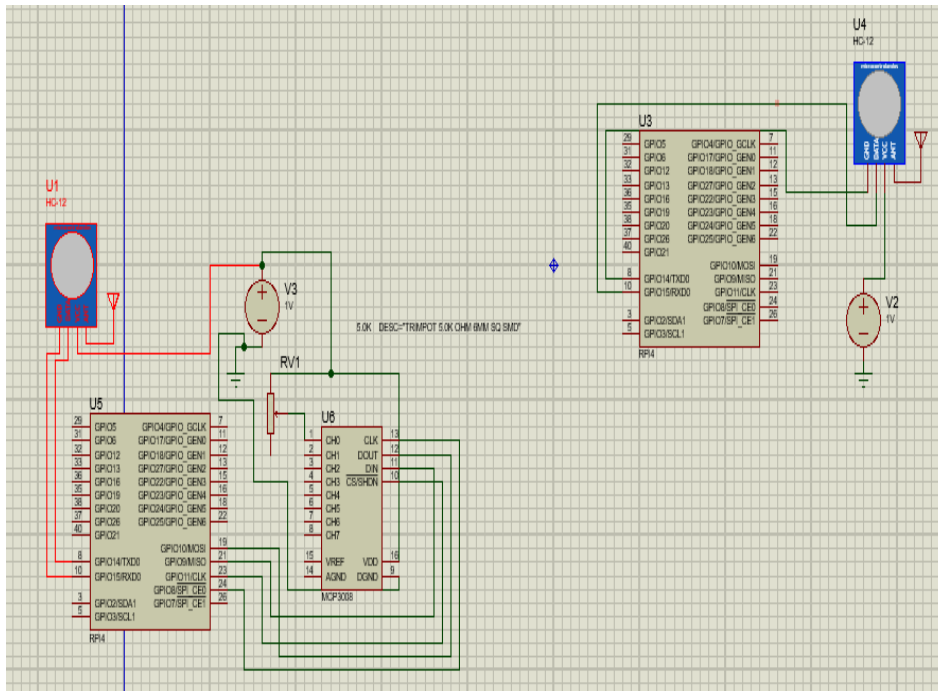


Figure 20 Hand Glove

6.8 Final Diagram:



Chapter 6

7. Results and Analyses

```
receiver.py ✕
22  if x=='1':
23      print(chr(data[0]))                #Send back the received data
24      ser.write("Gesture 1".encode('utf-8'))
25      ser.close()
26      GPIO.output(17,0) #left
27      GPIO.output(27,0) #right movemet
28      GPIO.output(23,0) #back
29      GPIO.output(24,1) #frwd movemet
30  elif x=='2':
31      print(chr(data[0]))                #Send back the received data
32      ser.write("Gesture 2".encode('utf-8'))
33      ser.close()
34      GPIO.output(17,0) #left
35      GPIO.output(27,0) #right movemet
36      GPIO.output(23,1) #back
```

Figure 21 receiver side code.

```
38  elif x=='3':
39      print(chr(data[0]))                #Send back the received data
40      ser.write("Gesture 3".encode('utf-8'))
41      ser.close()
42      GPIO.output(17,0) #left
43      GPIO.output(27,1) #right movemet
44      GPIO.output(23,0) #back
45      GPIO.output(24,0) #frwd movemet
46  elif x=='4':
47      print(chr(data[0]))                #Send back the received data
48      ser.write("Gesture 4".encode('utf-8'))
49      ser.close()
50      GPIO.output(17,1) #left
51      GPIO.output(27,0) #right movemet
```

Figure 22 receiver side code.

```

47     print(chr(data[0]))           #Send back the received data
48     ser.write("Gesture 4".encode('utf-8'))
49     ser.close()
50     GPIO.output(17,1) #left
51     GPIO.output(27,0) #right movemet
52     GPIO.output(23,0) #back
53     GPIO.output(24,0) #frwd movemet
54 elif x=='5':
55     print(chr(data[0]))           #Send back the received data
56     ser.write("Gesture 5".encode('utf-8'))
57     ser.close()
58     GPIO.output(17,0) #left
59     GPIO.output(27,0) #right movemet
60     GPIO.output(23,0) #back
61     GPIO.output(24,0) #frwd movemet

```

Figure 23 receiver side code.

Shell x

```

1
1
2
3
4
5

```

Figure 24 receiver side code.

At the receiver side, serial communication port is opened in a while loop at the start. The raspberry pi then starts the communications. As it is the receiver side, it first waits for data at the receiving side. Whenever a data is read, it is stored in a named 'data'. This data is then compared in a if statement and whenever an if statement is true, it sends an output signal to the specified GPIO and other GPIOs are at voltage low. In this way different set of data invoked different output at different GPIOs.

```
FYP.py x
28
29
30 if x>=975 and y > 830 and a > 860 and b < 715:
31     ser.write("1".encode('utf-8'))
32     x=ser.read(9)
33     print(x)
34     ser.close()
35 elif adc.value>=975 and adc_1.value < 830 and adc_2.value > 860 and adc_3.value < 715:
36     ser.write("2".encode('utf-8'))
37     x=ser.read(9)
38     print(x)
39     ser.close()
40
41 elif adc.value>=975 and adc_1.value < 830 and adc_2.value < 860 and adc_3.value < 715:
42
Shell x
success
b'Gesture 1'
success
b'Gesture 2'
success
b'Gesture 3'
success
b'Gesture 4'
success
b'Gesture 5'
```

Figure 25 thresholds set in if-else statements.

```
FYP.py x
29
30 if x>=950 and y < 680 and a < 680 and b < 715:
31     ser.write("1".encode('utf-8'))
32     c=ser.read(9)
33     print(c)
34     ser.close()
35 elif x>=950 and y >= 680 and a < 680 and b < 715:
36     ser.write("2".encode('utf-8'))
37     c=ser.read(9)
38     print(c)
39     ser.close()
40
41 elif x>=950 and y >= 680 and a >= 680 and b < 715:
42
43     ser.write("3".encode('utf-8'))
44     c=ser.read(9)
45     print(c)
46     ser.close()
```

Figure 26 thresholds set.

```
FY
41
42     ser.write("3".encode('utf-8'))
43     c=ser.read(9)
44     print(c)
45     ser.close()
46     elif x>=950 and y >= 680 and a >= 680 and b >= 715:
47         ser.write("4".encode('utf-8'))
48         c=ser.read(9)
49         print(c)
50         ser.close()
51     elif x<950 and y < 680 and a < 680 and b < 715:
52         ser.write("5".encode('utf-8'))
53         c=ser.read(9)
54         print(c)
55         ser.close()
56     time.sleep(2)
```

```
FYP.py x
41
42     ser.write("3".encode('utf-8'))
43     c=ser.read(9)
44     print(c)
```

```
Shell x
>>> %Run FYP.py
channel 0 = 972
channel 1 = 742
channel 2 = 728
channel 3 = 831
b'Gesture 4'
channel 0 = 961
channel 1 = 747
channel 2 = 664
channel 3 = 836
b'Gesture 4'
channel 0 = 975
channel 1 = 745
channel 2 = 688
channel 3 = 835
```

Figure 27 notice channel values for gesture 4 meeting the if conditions.


```
FYP.py x
41
42     ser.write("3".encode('utf-8'))
43     c=ser.read(9)
44     print(c)

Shell x
channel 0 = 976
channel 1 = 639
channel 2 = 662
channel 3 = 635
b'Gesture 2'
channel 0 = 969
channel 1 = 687
channel 2 = 646
channel 3 = 679
b'Gesture 1'
channel 0 = 979
channel 1 = 688
channel 2 = 655
channel 3 = 691
b'Gesture 2'
channel 0 = 967
channel 1 = 675
channel 2 = 670
channel 3 = 679
b'Gesture 2'
```

Figure 28 notice channel values for gesture 4 meeting the if conditions.

At the sender side, the values at the 5 channels of the raspberry pi are taken as input and stored in the variable's 'x', 'y', 'a' and 'b'. These values are compared in an if statement in such a way, it judges a gesture. When a specified gesture is invoked, the sensor values fall in the specified range at that moment. Hence, an if statement at that moment becomes true. In an if statement, as it is the sender side, it first writes the specified command. Then it waits for the receiver to send us an acknowledgment in the form of gesture number. In figure 23, we can see different sensor values at different channels. When an if statement is true, the received acknowledgment can also be seen in the output such as 'gesture 1' or 'gesture 2' etc.

Chapter 7

8. Future work Recommendations and Conclusion:

The hand recognition system can be used in many other fields like medicine robotics, video editing television and sign language etc. Number of possible improvements can be done to extend this work in future. As recognition of the hand gestures in mix with different advances, like discourse recognition, you can preclude the likelihood that the customary mouse and console to connect with the client and the PC. The hand gesture recognition framework gives a more effective and normal collaboration, and a component of creative applications. Medical applications will likewise profit incredibly from the utilization of the intelligent hand gesture, the administrations productively and adequately to a patient during an operation in a sterile climate. Another significant application is to depend on communication via gestures recognition for hard of hearing individuals.

Crane control using hand gestures:

Crane operators require a lot of training before controlling cranes for challenging tasks. The training cost is very high. Using hand gesture recognition, we can reduce the cost and labor. Close human supervision is required in workspace of cranes. To minimize the risk of human hazards, we can customize HGCM according to the requirements of crane and its payload. The crane driver can simply wear a glove and with the help of few sets of gesture, a heavy loader like crane can be controlled.

Sign language for mute people:

In future two gloves can be made to communicate with each other. So that mute people can communicate with a person who does not know sign language. The hand gestures of the mean people can be read and converted into either voice signal or text. The sensors recognize the finger position each finger is equivalent to some word please or later these words, signals or letter could be converted into an electrical signal. The electrical signal can be sent to the Raspberry Pi through a module which forwards it to its specific GPO pin.

Self-Driving cars:

Gesture recognition can be used to build autonomous car. It is a vehicle that drives on its own. It can sense its environment and operating without drive being involved. It is not necessary for passenger to take control at any time. The vehicle can even drive without a passenger. For instance, Tesla.

An intriguing expansion to the investigation of the utilization of the combination of a hand gesture recognition, look, eye stare, and discourse acknowledgment, a wise multi-modular method for communication among individuals and their administration robots that help the wiped out and the debilitated, just as for the climate, and surveillance applications.

Another fascinating augmentation of our work is following and perceive the pose with two hands; this will prompt a set of gestures. Two-hand following, it very well may be utilized with one hand to make a gesture to make the number of the events, just as the other to play out every one of the important developments of the body. Two hand-held lights that can be incorporated with the virtual environment and to interface in a more regular method of utilizing the PC in a virtual environment. One of the further developments could be the tracking of hand in 3D space. A significant space of future exploration in the space of hand gesture recognition is the capturing of hand gestures and modifying them for various applications, guaranteeing that they can meet the requirements of the user. For instance, your hand are hands are wet, you want to unlock the phone you can simply use hand gestures to do it. it will protect you from dirty, wet hands and make it convenient. hand gesture recognition can also be used in the future from engineering. Gestures can control cranes, cars and heavy loaders. It can save cost and labor. Some tasks are life threatening and for those gestures can be used.

The world as it seems has evolved drastically in the last decade from imagining wireless discussions to practically controlling robots wirelessly have become a reality now. The HGCM is a practical demonstration of it enabling a person to control a machine at your fingertips. The development if continued can lead us into much bigger discoveries. As of now, the technology is limited to our imagination. The wireless communication between two raspberry pi is a better approach for hand gesture recognition than the usual vision-based technologies. As HGCM is cheaper and provides the same results as vision based.

Appendix

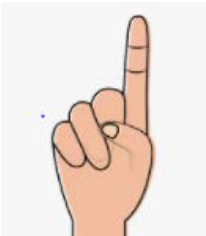
Instructions and Guidance:

Power up the glove and the machine.

Connect the 4 terminals of the machine with the end device you want to control. (Note: pin 13,14,16,18 is for gesture 1,2,3 and 4 respectively).

Now wear the glove in your hand.

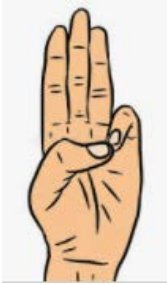
- To invoke the first gesture, bend your right hand's last three fingers and straighten up your index finger. This motion of fingers will be detected by the sensors and will send a specific signal to the machine.



- To invoke the second gesture, bend your right hand's last two fingers and straighten up your index finger and the middle finger. This motion of fingers will be detected by the sensors and will send a specific signal to the machine.



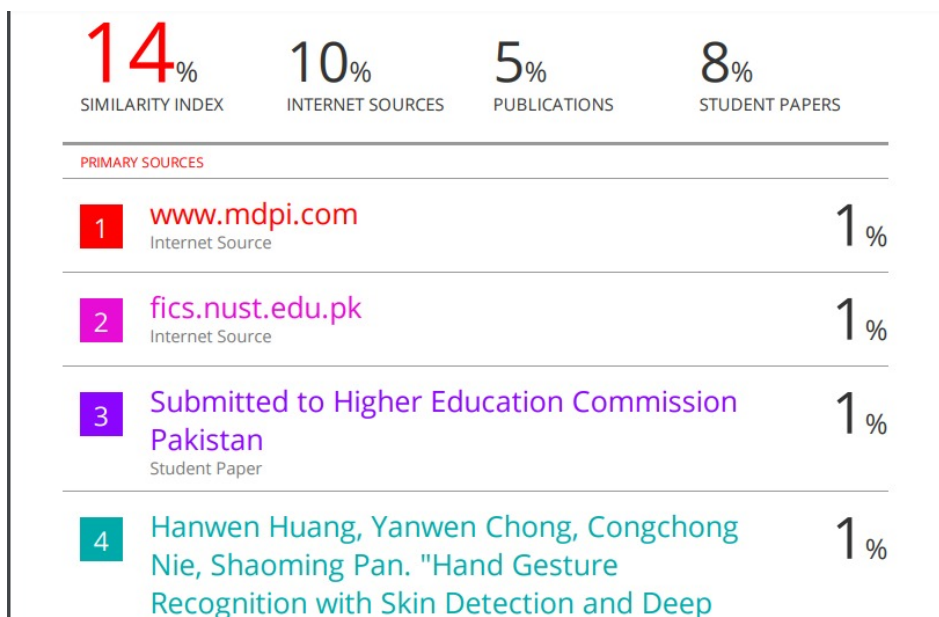
- To invoke the third gesture, bend your right hand's last finger and straighten up the other three fingers. This motion of fingers will be detected by the sensors and will send a specific signal to the machine.



- To invoke the fourth gesture, straighten up the four fingers. This motion of fingers will be detected by the sensors and will send a specific signal to the machine.



Plagiarism Report:



9. Bibliography

[1]https://www.researchgate.net/publication/221516122_Evaluation_of_alternative_presentation_control_techniques

[2] https://link.springer.com/chapter/10.1007%2F978-3-642-16493-4_36

Ng, C. B., Tay, Y. H., & Goi, B.-M. (2012). Vision-based Human Gender Recognition: A Survey. *arXiv: Computer Vision and Pattern Recognition*. Retrieved 6 20, 2021, from <https://arxiv.org/abs/1204.1611>

Pang, Y. Y., Ismail, N. A., & Gilbert, P. L. (2010). *A Real Time Vision-Based Hand Gesture Interaction*. Retrieved 6 20, 2021, from <http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.ieee-000005489218>

Su, M.-C., & Hung, C.-H. (2006). *Hand gesture recognition based on SOM and ART*. Retrieved 6 20, 2021, from <http://wseas.us/e-library/conferences/2006csc/papers/534-965.pdf>

Torralba, Murphy, Freeman, & Rubin. (2003). *Context-based vision system for place and object recognition*. Retrieved 6 20, 2021, from http://people.csail.mit.edu/billf/publications/context-based_vision_system.pdf

Yang, Z., Li, Y., Chen, W., & Zheng, Y. (2012). *Dynamic hand gesture recognition using hidden Markov models*. Retrieved 6 20, 2021, from <https://ieeexplore.ieee.org/document/6295092>

Gesture Recognition (Pang, Ismail, & Gilbert, 2010)

(Yang, Li, Chen, & Zheng, 2012) (Torralba, Murphy, Freeman, & Rubin, 2003)

(Ng, Tay, & Goi, 2012)

Comment [DT1]: [Explore Gesture Recognition](#)