VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnanasangama, Macche, Santibastwada Road
Belagavi-590018, Karnataka

A
UG PROJECT REPORT
on

Development of Health Monitoring System with Automated Bed Control using Hand Gesture & Voice Recognition

Submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering
in
Electronics & Communications Engineering - ECE
by

1DS19EC714	**Dhananjay Natani**
1DS19EC715	**Ekta Sinha**
1DS19EC721	**Krati Sharma**
1DS19EC725	**Narayan Kumar**

Under the guidance
of

Prof. Manasa R.K
Departmental Project Guide/Supervisor
Assistant Professor, ECE Dept., DSCE, Bengaluru

Department of Electronics & Communication Engineering
(An Autonomous College affiliated to VTU Belgaum, accredited by NBA & NAAC, Ranked by NIRF)
Shavige Malleshwara Hills, Kumaraswamy Layout,
Bengaluru-560078, Karnataka, India

2022-23

Certificate

Certified that the project work entitled "**Development of Health Monitoring System with Automated Bed Control using Hand Gesture & Voice Recognition**" carried out by **Dhananjay Natani** (1DS19EC714), **Ekta Sinha** (1DS19EC715), **Krati Sharma** (1DS19EC721), **Narayan Kunar** (1DS19EC725) are bonafide students of the ECE Dept. of Dayananda Sagar College of Engineering, Bangalore, Karnataka, India in partial fulfillment for the award of Bachelor of Engineering in Electronics & Communication Engineering of the Visvesvaraya Technological University, Belagavi, Karnataka during the academic year 2022-23. It is certified that all corrections / suggestions indicated for project work have been incorporated in the report deposited to the ECE department, the college central library & to the university. This final year project report **(Course Code : 19EC8ICPR2)** Phase-II has been approved as it satisfies the academic requirement in respect of project work prescribed for the said degree.

Dept. Project Coordinators (Section incharges) Prof. Manasa R	Project Guide Prof. Manasa R
Head of the Department Dr. T.C.Manjunath, Ph.D. (IIT Bombay)	Dr. B.G. Prasad Principal, DSCE

External Project Viva-Voce

Name of the project examiners (int & ext) with date :

1 : Signature : _____

2 : Signature : _____

Declaration

Certified that the project work entitled, "**Development of Health Monitoring System with Automated Bed Control using Hand Gesture & Voice Recognition**" with the project work course code **19EC8ICPR2** is a bonafide work that was carried out by ourselves in partial fulfillment for the award of degree of Bachelor of Engineering in Electronics & Communication Engg. of the Visvesvaraya Technological University, Belagavi, Karnataka during the academic year 2022-23. We, the students of the project group/batch no. S-07 do hereby declare that the entire project work has been done on our own & we have not copied or duplicated any other's work or may be the extension of the works done by the earlier students. The results embedded in this UG project report has not been submitted elsewhere for the award of any type of undergraduate degree.

Student Name-1 : Mr. Dhananjay Natani
USN : 1DS19EC714

Sign : _____

Student Name-2 : Ms. Ekta Sinha
USN : 1DS19EC715

Sign : _____

Student Name-3 : Ms. Krati Sharma
USN : 1DS19EC721

Sign : _____

Student Name-4 : Ms. Narayan Kumar
USN : 1DS19EC725

Sign : _____

Date : 10/ 06 /2023

Place : Bengaluru - 78

Acknowledgement

It has been a long and strenuous project, but we have finally reached the finish line with our final product. We owe an immense amount of gratitude to all those who contributed in one way or another; without you, this would not be possible. The completion of such an endeavor requires support from many people, so thank you for your contributions!

We are grateful to everyone including our Chairman – Dr. Hemchandra Sagar, Vice- Chairman – Dr. Premchandra Sagar, our Secretary Galiswamy Sir, and Joint Secretary- Tintisha Sagar.

We would also like to express our gratitude to our college Principal sir- Dr. B.G. Prasad wholeheartedly.

This project would not have been possible without the help of our teachers and supervisors for their kindness as well as patience during the process. So we want to thank our HOD sir- Dr. T.C. Manjunath, Department Project Coordinators- Prof. Manasa.

Also, a special thanks to our project guide- Prof. Manasa R, for instructing, guiding, and helping us at every step during the whole process.

Last but not the least, we sincerely appreciate all those whose contribution was either direct or indirect–our Dept. staff members (Teaching and non-teaching), and our friends and family. We could have never done it without them!

Finally, we want to extend our thanks toward God (The Great Almighty) because he always had his blessing on us

Table of Contents

Title Sheet		i
Certificate		ii
Declaration		iii
Acknowledgement		iv
Table of Contents		v
List of Figures		vi
List of Tables		vii
Nomenclature and Acronyms		viii
Abstract		1
Chapter 1	Introduction	2
	Overview of the project work	3
	Background information about the project work	3
	Motivation obtained to take up the project work	3
	Problem statement of the project work	4
	Objectives of the project work	4
	Scope of the project work	5
	Organization of the project report	5
Chapter 2	Literature Survey	6
Chapter 3	Project Details	10
Chapter 4	Simulation or Experimental Results & Discussions	24
Chapter 5	Conclusions, Future Work & Outcome of the project work	29
References		31
Appendix		33
Hard copy of the presented conference paper / published journal paper		75
Plagiarism reports		76
CO-PO Mapping Justification Sheets		81
Budget Estimation Sheets		82

List of Figures

Fig. 1: Block-diagram of the proposed methodology 10

Fig. 2: Circuit diagram of the proposed system 11

Fig. 3: Flow-chart of the proposed methodology for Hand Gesture 15

Fig. 4: Flow diagram of voice recognition model 15

Fig. 5: Flow diagram of temperature sensor model 16

Fig. 6: Flow diagram of fc-22 gas sensor model 16

Fig. 7: Flow diagram of heart rate sensor model 17

Fig. 8: Flow diagram of max30100 sensor model 18

Fig. 9: Screenshot of Thingspeak where sensors vital are monitored 24

Fig. 10: Voice command is given to system and bed movement is shown 24

Fig. 11: Sensors is activated my placing finger on heart rate sensor 25

Fig. 12: placing finger on heart rate sensor & is shown on lcd 25

Fig. 13: Hand Gestures 25

Fig. 14: Hardware Prototype of proposed system 26

Fig. 15: Bed Movement using Hand Gestures for up and down movement 26

Fig. 16: Bluetooth IC 35

Fig 17: H Bridge Motor Control Circuit Using L293d IC 36

Fig. 18: H Bridge Module 36

Fig. 19: DC Motor 37

Fig. 20: Raspberry Camera 5MP 38

Fig.21: The dual power supply circuit provides regulated DC output 39

Fig.22: Dual Supply board 40

Fig.23: USB Port Module 41

Fig24: DHT11 Sensor 43

Fig 25: Gas Sensor Working 44

Fig. 26: Sp02 Sensor 45

Fig.27: Max30102 Pinout 46

Fig 28: LCD Pinout 47

List of Tables

Table 1: Technical details of Bluetooth module 35
Table 2: Camera 15 pin connector pinout 39
Table 3: DHT11 Pinout 42
Table 4: Gas Sensor Pinout 45
Table 5: LCD Pinout 49

Nomenclature and Acronyms

Abbreviations (Alphabetical Order):

AI	Artificial Intelligence
API	Application Programming Interface
BLE	Bluetooth Low Energy
BP	Blood Pressure
DHT11	Digital Humidity and Temperature Sensor
DSCE	Dayananda Sagar College of Engineering
ECE	Electronics & Communication Engineering
GPIO	General Purpose Input/Output
IEEE	Institute of Electrical & Electronics Engineers
IoT	Internet of Things
LED	Light-Emitting Diode
LCD	Liquid Crystal Display
PCB	Printed Circuit Board
PWM	Pulse Width Modulation
RFID	Radio Frequency Identification
SDK	Software Development Kit
SMPS	Switch Mode Power Supply
SPo2	Oxygen Saturation
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
VR	Virtual Reality
WLAN	Wireless Local Area Network

Abstract

The Gesture-Controlled and Voice-Controlled Bed System is a groundbreaking project that aims to address the challenges faced by the elderly and individuals with disabilities or paralysis in accessing continuous care and improving their overall well-being. The project focuses on developing an innovative bed system that can be controlled through hand gestures and voice commands, offering greater independence and comfort for patients.

The project involves the design and integration of gesture recognition and voice recognition technologies into the bed system, enabling users to adjust the bed's position effortlessly. Additionally, the system incorporates various sensors such as heartbeat sensors, temperature sensors, SPo2 and blood pressure monitors, humidity sensors, and air quality sensors to continuously monitor the patient's vital parameters and environmental conditions.

The anticipated outcomes of this project include the development of a user-friendly and intuitive bed system that empowers patients to control their bed positions independently. The continuous monitoring of vital parameters and environmental conditions will provide valuable insights for healthcare professionals, facilitating timely interventions and personalized care. Furthermore, the integration of emergency alert features ensures prompt response and assistance during critical situations.

Overall, the Gesture-Controlled and Voice-Controlled Bed System project aims to revolutionize patient care by enhancing comfort, autonomy, and the overall healthcare experience. The anticipated results have the potential to significantly impact the lives of individuals with disabilities or paralysis and contribute to the advancement of healthcare technology.

Keywords: *Gesture-controlled bed system, Voice-controlled bed system, Continuous monitoring, Elderly care, Disabilities and paralysis, Independence, Healthcare technology, Vital parameter monitoring, Hand gesture recognition, Voice recognition*

Chapter-1

Introduction

In modern times, society is facing a multitude of social and economic challenges, leading to less time being available for the elderly and individuals with disabilities or paralysis. Many disabled individuals require continuous monitoring, which can prove to be difficult due to the unavailability of full-time caregivers or financial constraints. To address this problem, we have developed a gesture-controlled & voice-controlled bed system that utilizes hand gestures and voice recognition modules to control the bed's motion, specifically the up and down movements. This bed system is intended for use in hospitals and various other situations, such as for disabled individuals, paralyzed patients, accident victims, and the elderly. The goal is to minimize the need for caretakers while increasing patient comfort. By integrating voice recognition technology into the Smart bed, we have made the system user-friendly and easy to operate.

In the realm of healthcare, continuous monitoring of patients' vital parameters is crucial for timely intervention and ensuring their well-being. To address this need, we have developed a sophisticated Smart bed system equipped with various sensors, including a heartbeat sensor, temperature sensor, SPo2 and blood pressure monitor, humidity sensor, and air quality sensor. These sensors enable accurate measurement of both the patient's vital parameters and the environmental conditions surrounding them while lying on the bed. By integrating these sensors into the Smart bed, we can gather real-time data on the patient's heart rate, temperature, blood oxygen levels, blood pressure, humidity levels, and air quality. This comprehensive information provides healthcare professionals with valuable insights into the patient's overall health status and the environment they are in.

In addition to monitoring, the Smart bed system also incorporates an emergency button that can be used by the patient in case of any discomfort or distress. If any vital health parameter exceeds the normal threshold values, a built-in buzzer is activated to alert both the patient and the healthcare staff, ensuring immediate attention and necessary action can be taken.

1.1 Overview of the project work

The gesture-controlled and voice-controlled bed system is an innovative solution designed to address the challenges faced by the elderly and individuals with disabilities or paralysis. By incorporating hand gesture recognition and voice recognition technology, the system allows users to control bed movements independently, enhancing their autonomy and reducing the need for full-time caregivers. The system also includes a range of sensors for real-time monitoring of vital parameters such as heartbeat, temperature, blood pressure, and air quality. This comprehensive monitoring enables timely interventions and personalized care. With its user-friendly interface and emergency alert features, the Smart bed system revolutionizes patient care, providing comfort, safety, and improved healthcare outcomes.

1.2 Background information about the project work

The project work focuses on addressing the challenges faced by the elderly and individuals with disabilities or paralysis in accessing continuous care. The present healthcare system relies heavily on full-time caregivers, which may not always be feasible due to social and financial constraints. This creates a need for innovative solutions that enhance patient independence and comfort while ensuring their well-being. The integration of gesture control and voice recognition technology in the bed system offers a user-friendly and intuitive approach to adjust bed positions. By incorporating sensors for continuous monitoring of vital parameters, the system enables timely interventions and personalized care. This project aims to contribute to the advancement of healthcare technology by providing a comprehensive and efficient solution to improve the quality of life for vulnerable populations.

1.3 Motivation obtained to take up the project work

The motivation to undertake this project work stems from the pressing need to address the challenges faced by the elderly and individuals with disabilities or paralysis in accessing continuous care. The limitations of full-time caregivers, coupled with social and economic constraints, have created a demand for innovative solutions that enhance patient independence and comfort. By developing a gesture-controlled and voice-controlled bed system, we aim to provide a user-friendly and intuitive solution that allows individuals to adjust their bed positions according to their needs. Additionally,

integrating sensors for continuous monitoring of vital parameters ensures timely interventions and personalized care. The opportunity to improve the quality of life for vulnerable populations and contribute to the advancement of healthcare technology drives our motivation to undertake this project work.

1.4 Problem statement of the project work

- The electronic bed systems used in hospitals, healthcare centers used for disabled patients, paralyzed persons, in accident cases and for old age people are not completely automated and advanced.
- The available existing electronic bed systems have only two movements of the bed such as up and down and they require assistance of another person to operate and cannot be used independently by all kind of patients and disabled people.
- There is a fulltime need of caretaker requirement which may not be always possible due to social and financial issues.
- Vital health parameters are not efficiently monitored.
- Emergencies may not be promptly addressed, and healthcare providers may not be notified in a timely manner.
- Patients face increased risks of adverse outcomes and discomfort.
- Caregiving resources are not allocated efficiently.
- Access to healthcare services is limited, especially for individuals in remote areas or home settings.

1.5 Objectives of the project work

The main objective of the project work is to develop a hand gesture detection model and voice recognition system to act as intermediate for the people to communicate the need (change the position of the bed). So, they not depend on others for their needs and regular monitoring of certain health parameters and timely reminder of supplements can help in avoiding untimely health conditions and can even save a life.

- Objective 1: To Design an Automatic Bed System which can be controlled by Hand Gestures.
- Objective 2: To Design an Automatic Bed System which can be controlled by voice recognition.

- Objective 3: To monitor the patient conditions using HB, Temperature, SpO2, BP.
- Objective 4: In case of Emergency send intimation message to doctor or care taker through Telegram.
- Objective 5: To monitor real time data of health parameters and syncing it to the ThingSpeak.

1.6 Scope of the project work

The scope of this project extends beyond the immediate context of healthcare facilities and has potential applications in various settings. The gesture-controlled and voice-controlled bed system can be implemented in hospitals, rehabilitation centers, nursing homes, and even in home care for the elderly or individuals with disabilities. Furthermore, the integration of sensor technology for continuous health monitoring opens avenues for remote patient monitoring, telemedicine, and the development of smart healthcare environments. The project's outcomes have the potential to revolutionize patient care, enhance independence, and improve the overall quality of life for individuals in need of continuous care.

1.7 Organization of the project report

The project report is structured in a well-organized manner to provide a comprehensive overview of the undertaken work. In Chapter 1, the report begins with an introduction that outlines the project's objectives and significance. This section sets the context for the entire report, giving readers a clear understanding of the project's purpose.

Chapter 2, the Literature Review, delves into an extensive summary of existing research and studies conducted by other researchers and authors. It critically evaluates the limitations and drawbacks of previous works, establishing the need for the current project.

Moving on to Chapter 3, the System Overview and Hardware/Software Description section, readers are presented with the block diagram and working principle of the project. This chapter also includes a detailed discussion on the hardware and software tools utilized in the project, providing readers with insights into the technical aspects of the system.

Chapter 4, titled Experimental Setup, Results, and Discussion, takes a closer look at the practical implementation of the project. It showcases the experimental setup used for testing and presents the obtained results. The chapter includes an in-depth analysis of the experimental findings, highlighting the application advantages and limitations of the project.

In Chapter 5, the report concludes with the final section encompassing the Conclusion, Future Work, and Scope of the Project. Here, the key findings and achievements of the project are summarized, and potential avenues for future development are discussed. This chapter also explores the broader scope of the project's application in the outside world, shedding light on its potential impact and significance.

Additionally, the appendices section includes supplementary materials such as paper presentations, publications, certificates, photographs, plagiarism reports, CO PO mapping sheet, and budget estimations, providing supporting evidence and documentation for the project.

Chapter 2

Literature Survey

S. No.	Title	Author	Year
1	Care-giver alerting for bedridden patients using hand gesture recognition system	E Spandana, M Rajasekar and N Sandhya	2021
2	Smart Medical beds in patient-care environment of the 21st century: A state of Art	Ignacio Ghersi , Mario Mariño1 and Mónica Teresita Miralles.	2018
3	Gesture Controlled Bed Movement	Ms. T. Umarani1, Ms. I. Preethy Ms. N.M. Thennarasi, Ms. J.E. Jeyanthi	2018
4	Survey Automatic Bed Position Control Based on Hand Gesture Recognition for Disabled Patients	Ms. Kalpana Lamb, Prof.Mrs. Swati Madhe	2016

[1] E Spandana, M Rajasekar and N Sandhya. "Care-giver alerting for bedridden patients using hand gesture recognition system". Journal of Physics: Conference Series, 2021.

In this study, they discuss an alerting system that makes a better alarm model which notifies care-givers when patients require assistance by giving defined hand gestures. The study is related to hand gesture based alerting system which uses camera module to capture the hand images. The system uses background subtraction, image recognition and speech conversion to notify the healthcare providers. With this alerting system it increases the operational efficiency, responsiveness, patient's and other stakeholders' satisfaction. For recognized hand gesture text and voice is obtained as output which alerts the care-giver to help the bed-ridden patients. The implemented methodology is a real-time approach for hand gesture recognition system on alerting system for bed-ridden patients. 1200 hand gesture images i.e., 300 images for each hand gestures with defined meanings have been used. The gesture is recognized through Image segmentation and morphological operation on the captured image for further feature extraction. For recognized hand gesture text and voice is obtained as output which alerts the care-giver to help the bed-ridden patients rather than using wire-controlled system.

[2]. Ignacio Ghersi , Mario Mariño1 and Mónica Teresita Miralles. "Smart medical beds in patient-care environments of the twenty-first century: a state-of-art survey". Ghersi et al. BMC Medical Informatics and Decision Making (2018)

In this study, the new generation of electric medical beds is defined, with the final stage of the proposed timeline for these devices being covered. Functional, aesthetic andinteractive features are presented, and the current global market for medical beds and related standards are also assessed. Finally, discussions concerning rising challenges and opportunities for these systems are explored, with the potential for adding further monitoring and assistive implementations into medical devices and environments being highlighted.

[3]. Ms. T. Umarani1, Ms. I. Preethy Ms. N.M. Thennarasi, Ms. J.E. Jeyanthi. "Gesture controlled bed movement". International Research Journal of Engineering and Technology (IRJET) Volume: 06 Issue: 12 | Dec 2019

This project states that a problem of adjusting the position of the bed with the help of hand gesture. It needs inter mutual application for hand gesture recognition for implementing this work in a real-time system. For this real-time system, webcams are used to enable the hand gesture for human-computer interaction. It is proposed to demonstrate a real-time system to change the position of bed by hand gesture signs using webcams and simple hardware components.

[4]. Ms. Kalpana Lamb, Prof. Mrs. Swati Madhe "Automatic Bed Position Control Based on Hand Gesture Recognition for Disabled Patients". International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), 2016

This paper presents the background subtraction algorithm, giving the promising results for hand gesture recognition. This can be effectively used in real time processing in real time applications. Here we use different wavelet transform out of which Symlet wavelet transform give results accurate as compared with other transforms. The gesture is then given to the micro-controller via RS-232 communication cable for further processing, the controller processes the transferred data code gesture and passes this to DC motor driver to drive the motor clockwise or

anti-clockwise depend on the gesture given by the patient the position of the bed changes as per the given gesture input as up/down and right/left. In majority of the work done by the various authors presented in the previous paragraphs, there were certain drawbacks or short-comings that they lack on certain aspects such as movements of those beds. The smart beds also lack monitoring of certain vital parameters of human body and thus the proposed result in this project work comprise of overcoming these. The project comprises of certain other new features included for which algorithms are going to be developed in order to inculcate them in the prototype.

Chapter 3

Project Details

3.1 Block diagram of the proposed system

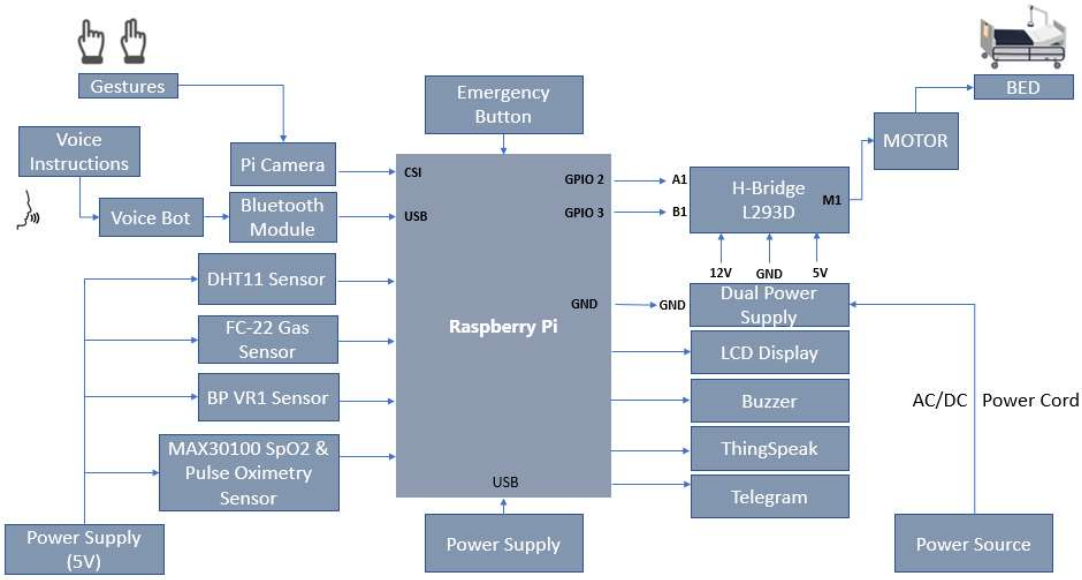


Fig. 1 : Block-diagram of the proposed methodology

The project utilizes various blocks to implement its functionality. Here is a brief description of the blocks used:

1. **Gesture Recognition Block**: This block incorporates gesture recognition technology, which enables users to control the bed's movements through hand gestures. It uses sensors and algorithms to interpret the gestures and trigger the corresponding actions.
2. **Voice Recognition Block**: The voice recognition block allows users to control the bed using voice commands. It utilizes speech recognition algorithms and audio processing techniques to interpret and execute the spoken instructions.
3. **DC Motor Control Block**: This block includes a DC motor and a DC motor driver responsible for controlling the bed's movement. The motor driver receives signals from the gesture recognition or voice recognition block and adjusts the motor's speed and direction accordingly.
4. **Sensor Network Block**: The sensor network block consists of various sensors, such as heartbeat sensors, temperature sensors, SPo2 and blood pressure monitors,

humidity sensors, and air quality sensors. These sensors continuously monitor the patient's vital parameters and the environmental conditions around the bed.

5. **Raspberry Board Block**: The Raspberry board block is responsible for processing the analog biomedical signals obtained from the sensors. It performs signal conditioning, data conversion, and analysis to derive meaningful insights about the patient's health status.

   These blocks work together to enable gesture-controlled and voice-controlled movements of the bed, while also facilitating continuous monitoring of vital parameters and environmental conditions.

3.2 Circuit diagram of the proposed system

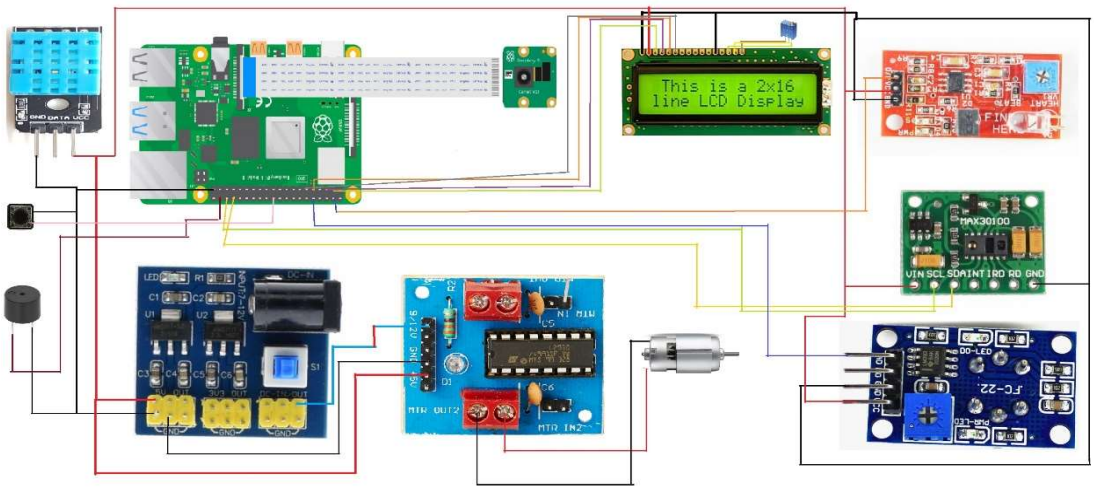


Fig. 2 : Circuit diagram of the proposed system

3.3 Algorithm

3.3.1 Algorithm For Hand Gesture

1. Initialize the Raspberry Pi and connect a camera module to it.
2. Install an image processing library on the Raspberry Pi.
3. Set up the motor driver and microcontroller to control the motor of the bed.
4. Write a program to capture video input from the camera module.
5. Process the video input using the image processing library to detect hand gestures.
6. If the detected gesture is a "one" gesture, send a signal to the motor to move the bed up.

7. If the detected gesture is a "two" gesture, send a signal to the motor to move the bed down.
8. If no hand gesture is detected, ignore it and go back to step 4 to capture more video input.
9. Test the system and ensure that it is functioning correctly.
10. Ensure that the motorized bed is safe to use for the intended purposes and that all necessary safety precautions have been taken.
11. If any errors or exceptions occur, handle them appropriately.
12. When finished, turn off the Raspberry Pi and disconnect the camera module and motor.

3.3.2 Algorithm For Voice Recognition

1. Initialize the Raspberry Pi and connect a Bluetooth module using the USB port to it.
2. Set up the motor driver and microcontroller to control the motor of the bed.
3. Capture audio input from the microphone on a mobile phone.
4. Process the audio input using the voice-to-text bot application installed on the mobile phone and send the text output to Raspberry Pi via the Bluetooth module.
5. If the recognized text is “move", send a signal to the motor to move the bed up.
6. If the recognized text is "down", send a signal to the motor to move the bed down.
7. If the recognized text is neither “move" nor "down", ignore it and go back to step 3 to capture more audio input.
8. Test the system and ensure that it is functioning correctly.
9. When finished, turn off the Raspberry Pi and disconnect the mobile phone and motor.

3.3.3 Algorithm For DHT11 Temperature Sensor

1. Initialize the DHT11 sensor and establish communication with the raspberry pi.
2. Initialize the Telegram API for sending alerts.
3. Initialize the ThingSpeak API for sending data.
4. Set up a loop to continuously monitor temperature data.
5. Read the temperature value from the DHT11 sensor.
6. If the temperature is greater than 32 degrees Celsius:

7. Send an alert message on Telegram.
8. Send the temperature data to ThingSpeak for storage and visualization.
9. Wait for a specified interval (e.g., 1 minute) before repeating the process.
10. Handle any errors or exceptions that may occur during the communication or data retrieval process.
11. continue running as needed.

3.3.4 Algorithm For FC-22 Gas Sensor

1. Initialize the FC22 gas sensor and establish communication with the microcontroller.
2. Initialize the Telegram API for sending notifications.
3. Initialize the ThingSpeak API for sending data.
4. Set up a loop to continuously monitor air quality.
5. Read the gas sensor values to measure air quality.
6. Perform data processing or calibration to obtain meaningful air quality measurements.
7. If the air quality is below a certain threshold (indicating poor air quality):
8. Send a notification message on Telegram.
9. Send the air quality data to ThingSpeak for storage and analysis.
10. Wait for a specified interval (e.g., 1 minute) before repeating the process.
11. Handle any errors or exceptions that may occur during the communication or data retrieval process.
12. continue running as needed.

3.3.5 Algorithm For Heart Rate Sensor

1. Initialize the heart rate sensor and establish communication with the microcontroller.
2. Initialize the Telegram API for sending notifications.
3. Initialize the ThingSpeak API for sending data.
4. Set up a loop to continuously monitor the heart rate.
5. Read the heart rate sensor values to measure the heart rate.
6. If the heart rate is below 40 BPM:
7. Send a notification message on Telegram indicating low heart rate.
8. Send the heart rate data to ThingSpeak for storage and analysis.
9. Wait for a specified interval (e.g., 1 minute) before repeating the process.

10. Handle any errors or exceptions that may occur during the communication or data retrieval process.
11. continue running as needed.

3.3.6 Algorithm For MAX30100 Sp02 & Pulse Oximetry Sensor

1. Initialize the MAX30100 sensor and establish communication with the microcontroller.
2. Initialize the Telegram API for sending notifications.
3. Initialize the ThingSpeak API for sending data.
4. Set up a loop to continuously monitor the SpO2 and blood pressure.
5. Read the SpO2 and blood pressure values from the MAX30100 sensor.
6. If the SpO2 is below 80% and blood pressure is greater than 170mmHg:
7. Send a notification message on Telegram indicating low SpO2 and high blood pressure.
8. Send the SpO2 and blood pressure data to ThingSpeak for storage and analysis.
9. Wait for a specified interval (e.g., 1 minute) before repeating the process.
10. Handle any errors or exceptions that may occur during the communication or data retrieval process.
11. continue running as needed

3.4 Flow-chart / Data Flow Diagram

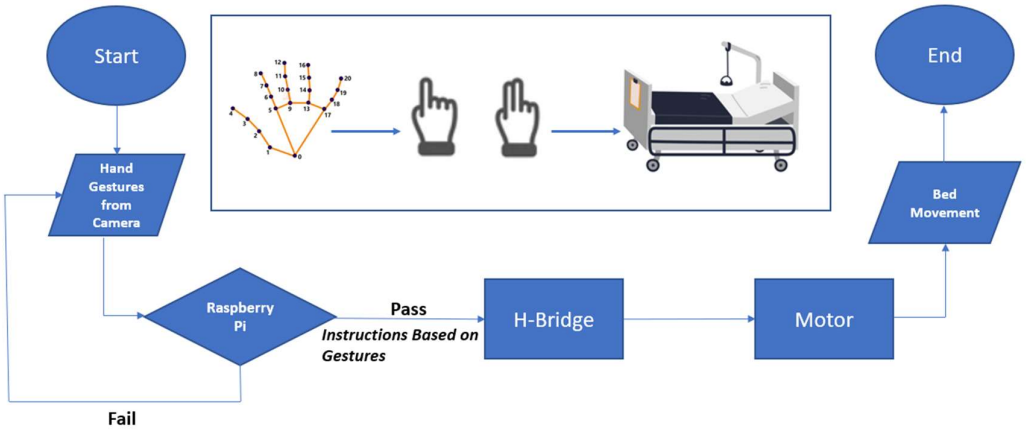


Fig. 3 : Flowdiagram of the proposed methodology for Hand Gesture

The plans and techniques used in the design and development of the Hand gesture recognition for change the position of automatic adjustable beds in hospitals are described in this section. In order to perform hand gesture recognition to control bed position, many tools, libraries and languages are used like Open-Source Computer Vision Library. Open CV established in Python Language. Python 3.7.4 version is an open-source environment for compile the source code. It has many inbuilt functions for recognizing hand gesture. The main component used here is the Raspberry Pi. It is used to get the information from the Patient, based on the command it can take the necessary action and then change the bed movement.

3.4.1 Flow Diagram of Voice Recognition

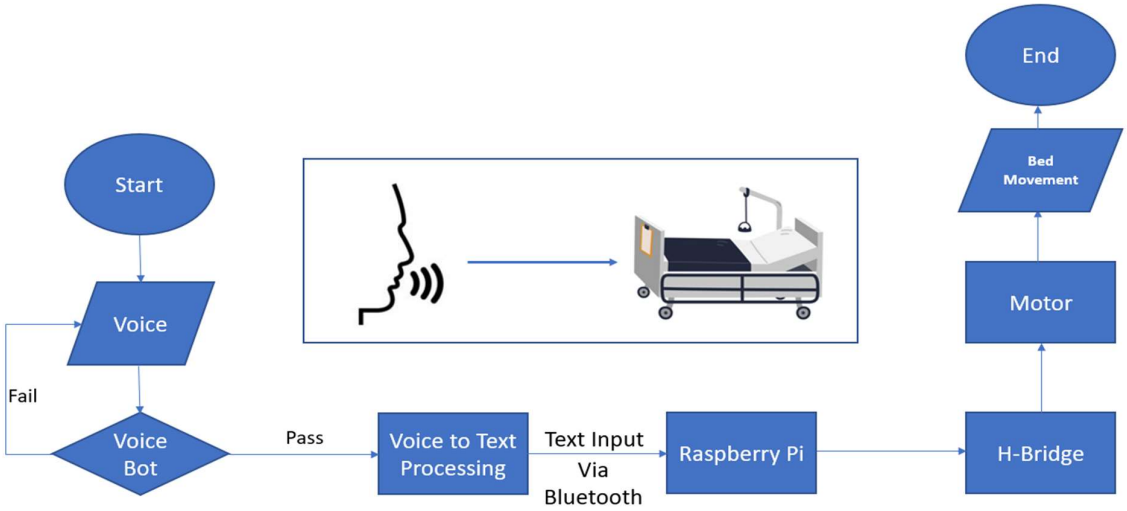


Fig. 4 Flow diagram of voice recognition model

3.4.2 Flow Diagram of DHT11 Sensor

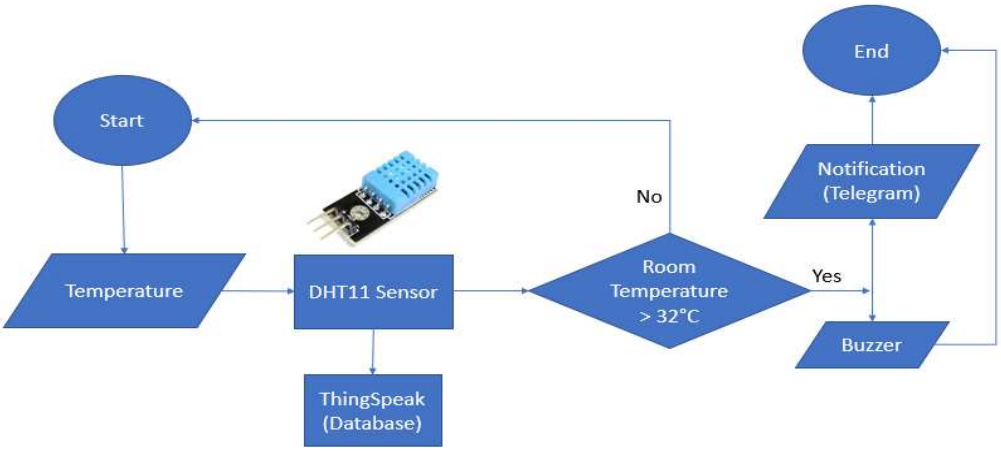


Fig. 5 Flow diagram of temperature sensor model

The flow diagram for the temperature sensor begins with the start of the temperature monitoring process. The temperature value is then read from the sensor, followed by a check to determine if it exceeds 32 degrees Celsius. If the temperature surpasses this threshold, a notification is sent on Telegram to alert the user. Concurrently, the temperature data is continuously transmitted to ThingSpeak, providing ongoing monitoring and data storage. The process repeats at regular intervals to ensure real-time tracking of temperature changes. Overall, this flow diagram showcases the systematic sequence of actions involved in monitoring the temperature, detecting threshold breaches, and facilitating timely notifications and data collection for further analysis.

3.4.3 Flow Diagram of FC-22 Gas Sensor

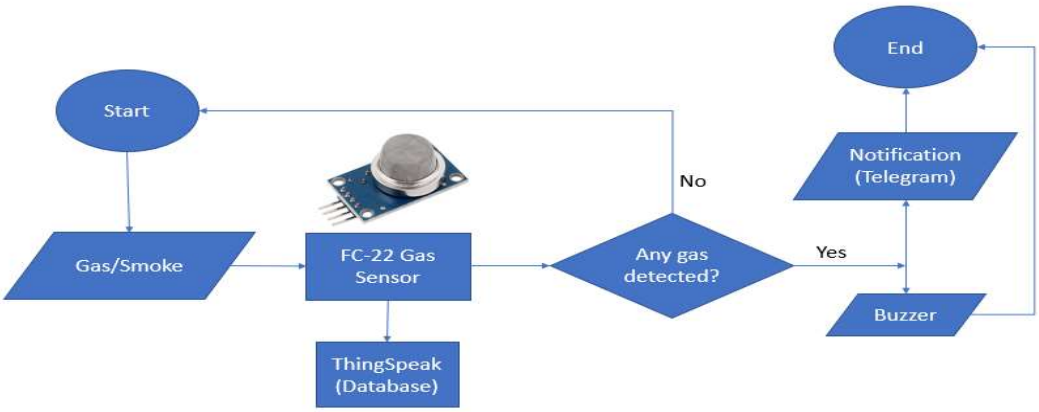


Fig. 6 Flow diagram of gas sensor model

The flow diagram for the gas sensor starts with the initiation of the gas quality monitoring process. The gas sensor measures the air quality and checks for the presence

of harmful gases. If the air quality is determined to be poor or if harmful gases are detected, a notification is sent on Telegram to alert the user. Simultaneously, the gas sensor continuously sends data to ThingSpeak, enabling continuous monitoring and data logging. The process repeats at regular intervals to ensure real-time tracking of air quality changes and prompt notifications in case of any concerns. This flow diagram illustrates the step-by-step process of monitoring gas levels, detecting abnormalities, and providing timely notifications and data collection for further analysis.

3.4.4. Flow Diagram of BP VR1 Sensor

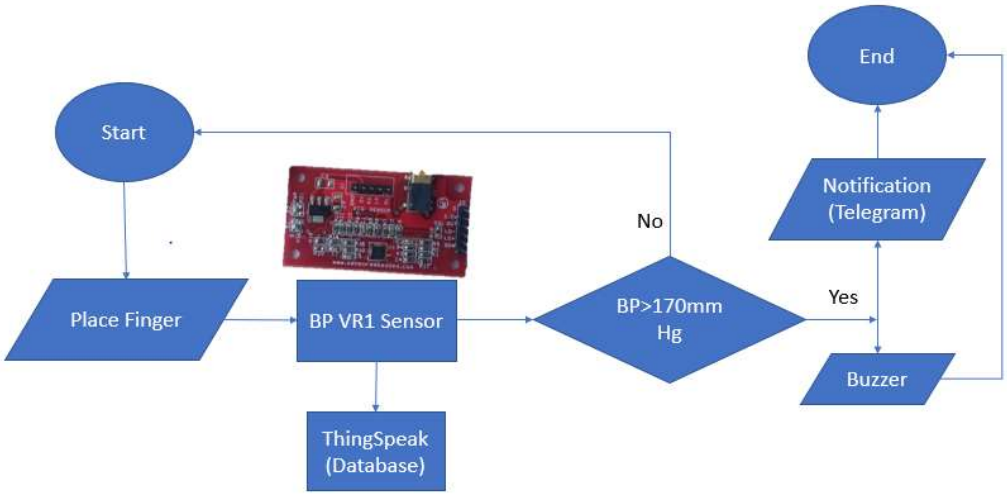


Fig. 7 Flow diagram of heart rate sensor model

The flow diagram for the heart rate sensor begins with the initiation of the heart rate monitoring process. The sensor measures the heart rate and continuously monitors it. If the heart rate drops below 40 beats per minute, a notification is sent on Telegram to notify the user about the low heart rate. Additionally, the heart rate sensor sends continuous data to ThingSpeak for real-time tracking and analysis. This allows for the ongoing monitoring of heart rate trends and the detection of any abnormalities. The flow diagram represents the sequential steps involved in monitoring the heart rate, detecting low heart rates, and providing notifications, as well as capturing and logging data for further evaluation and assessment.

3.4.5 Flow Diagram of MAX30100 SpO2 & Pulse Oximetry Sensor

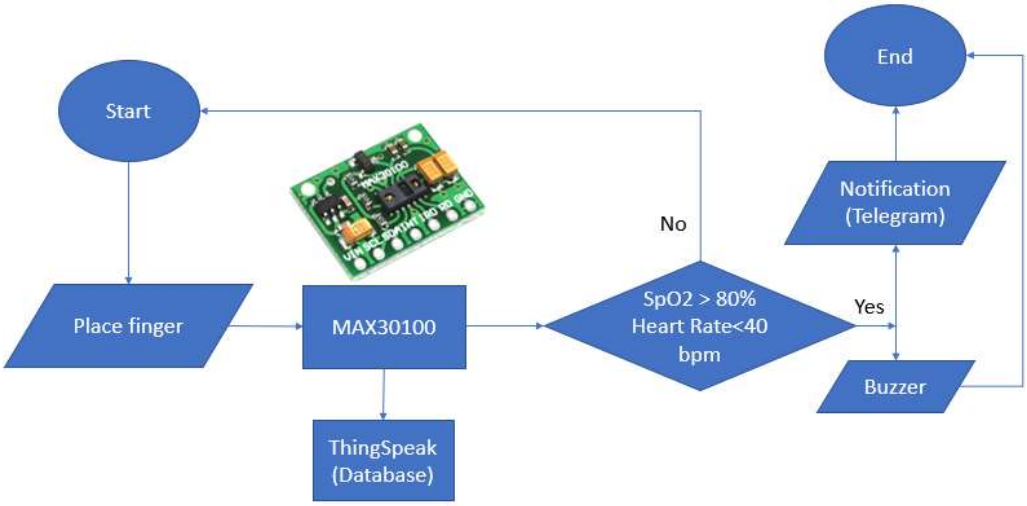


Fig. 8 Flow diagram of max30100 sensor model

The flow diagram for the MAX30100 SpO2 and Pulse Oximetry sensor starts with the initialization of the sensor for SpO2 and pulse rate measurement. The sensor continuously reads the SpO2 and pulse rate values from the user's finger. If the SpO2 level drops below 80% or if the blood pressure exceeds 170mmHg, a notification is sent on Telegram to alert the user about the abnormal readings. Additionally, the MAX30100 sensor sends continuous data to ThingSpeak for real-time monitoring and analysis. This allows for ongoing tracking of SpO2 levels and pulse rates, enabling early detection of any potential health issues. The flow diagram demonstrates the sequential steps involved in measuring SpO2 and pulse rate, detecting abnormal readings, notifying the user, and capturing data for further analysis and evaluation.

3.5 Flow-chart / Data Flow Diagram

Start: The project starts with the initialization of the system.

User Input: The user provides input in the form of hand gestures or voice commands.

Gesture Recognition: The system recognizes and processes the hand gestures to determine the desired bed movements.

Voice Recognition: If voice commands are given, the system performs voice recognition to convert the spoken commands into actionable instructions.

Control Decision: The system evaluates the recognized gestures or voice commands and makes decisions on how to control the bed.

DC Motor Control: Based on the control decision, the system sends commands to the DC motor to adjust the bed's position accordingly.

Sensor Data Collection: The system continuously collects data from various sensors, such as heartbeat sensors, temperature sensors, etc.

Data Processing: The collected sensor data is processed and analyzed to derive meaningful insights about the patient's health status.

Alert Generation: If any abnormal health conditions are detected, the system generates alerts or notifications to inform the relevant parties.

Output Display: The system displays the adjusted bed position, health status information, and any other relevant outputs on a user interface or external systems.

End: The project concludes after completing the desired tasks.

3.6 Hardware used

In the project, several hardware components have been utilized to address the defined problem and achieve the desired objectives. Here is a brief description of how some of these tools have been used:

Raspberry Pi 3 B: The Raspberry Pi serves as the main controller unit of the system, responsible for processing data, controlling the bed movements, and integrating various components.

Bluetooth BC417 Module: This module enables wireless communication between the Raspberry Pi and external devices, facilitating the integration of voice recognition and gesture control capabilities.

H-Bridge L293D Motor Driver: The motor driver is used to control the DC motor responsible for adjusting the bed's position. It provides the necessary power and direction control for smooth bed movement.

Raspberry Pi Camera 5MP: The camera module is utilized for capturing images or video for monitoring purposes, such as detecting hand gestures or recording the environment.

USB UART Serial Converter port: This component enables serial communication between the Raspberry Pi and other devices, facilitating data transfer and control.

Dual power supply: It is used to provide power to the H-Bridge motor driver in addition to powering the DC motor. The 5V power supply handles the control signals and logic components, while the 12V power supply drives the motor through the H-Bridge. This setup ensures efficient and reliable operation of the motor and allows for precise control of bed movements in the gesture-controlled system.

DC Motor: A DC motor is utilized to control the movement of the bed in the project.

DHT11 Sensor: The DHT11 sensor is used in the project to measure temperature and humidity levels in the environment where the patient is lying on the bed. It provides real-time data on these parameters, allowing healthcare professionals to monitor the comfort and well-being of the patient. The sensor's compact size, low power consumption, and reliable performance make it suitable for integration into the bed system for accurate environmental monitoring.

FC-22 Gas Sensor: The FC-22 gas sensor is utilized in the project to monitor air quality and detect harmful gases in the vicinity of the patient. It provides reliable measurements of various gases, allowing for early detection of potential health hazards. By integrating the FC-22 sensor into the bed system, healthcare professionals can ensure a safe and healthy environment for the patient, while enabling timely interventions in case of any gas-related risks or issues.

BP VR1 Sensor: The BP level sensor is utilized in the project to measure and monitor the blood pressure of the patient. It provides valuable information on the patient's blood pressure levels, enabling healthcare professionals to assess cardiovascular health and detect any abnormalities or fluctuations. By incorporating the BP level sensor into the bed system, continuous monitoring of blood pressure is achieved, allowing for timely intervention and management of hypertension or other blood pressure-related conditions. This helps in providing personalized care and ensuring the well-being of the patient.

LCD: The LCD 16*2 display is utilized in the project to provide a user-friendly interface for displaying the values of various sensors used in the system. It allows for clear and concise visualization of important information such as heart rate, blood pressure, temperature, humidity, and air quality readings. The LCD display enhances the usability of the system, enabling healthcare professionals and caregivers to easily monitor and

interpret the sensor data, promoting effective decision-making and ensuring optimal patient care.

MAX30100 SpO2 & Pulse Oximetry Sensor: The MAX30100 sensor is employed in the project to measure and monitor various vital parameters such as heart rate, blood oxygen saturation (SpO2), and pulse oximetry. It utilizes advanced optical sensing technology to accurately capture and analyze the patient's physiological data. By integrating the MAX30100 sensor into the bed system, continuous and non-invasive monitoring of vital signs is achieved, enabling healthcare professionals to assess the patient's overall health status and detect any abnormalities. This information facilitates timely interventions and personalized care, improving patient outcomes and ensuring their well-being.

All these sensors like heart rate sensor, SpO2 and BP sensor, temperature sensor, humidity sensor, and air quality sensor are used to collect real-time data on the patient's vital signs and environmental conditions. These sensors provide essential inputs for continuous monitoring and analysis.

3.7 Software used

The software tools used in the project work include **Raspberry Pi OS (Raspbian)**, **Python 3**, **TensorFlow, Mediapipe, Numpy,** and **ThingSpeak**.

Raspberry Pi OS is the operating system used on the Raspberry Pi board, providing a platform for running the project software.

Python 3 is the programming language used to develop the code for the gesture-controlled bed system and integrate various functionalities.

TensorFlow is a machine learning framework used for developing and training the hand gesture recognition model. It enables the system to accurately interpret and recognize different hand gestures for controlling the bed movements.

Mediapipe is a library that provides ready-to-use solutions for tasks such as hand tracking, pose estimation, and facial recognition. It is used to track and analyze hand movements in real-time.

Numpy is a numerical computing library in Python, used for efficient data handling and manipulation in the project. It helps in processing and analyzing the sensor data obtained from various sensors integrated into the bed system.

ThingSpeak is an IoT platform used for collecting and visualizing the sensor data. It enables remote monitoring of the patient's vital parameters and provides insights into their health condition.

These software tools were utilized to develop an integrated system that combines gesture recognition, sensor data processing, and IoT capabilities, enabling precise control and monitoring of the bed movements and patient's health parameters.

3.7 Interfacing

The interfacing process involved configuring the connections, ensuring proper voltage levels and signal compatibility, and writing code to establish communication and control between the hardware components and software tools. It required careful consideration of hardware specifications, pin mappings, and software libraries to ensure seamless integration and functionality.

3.8 Working principle of the proposed system

The overall system works by integrating hardware components, such as sensors, motors, and a microcontroller (Raspberry Pi), with software tools and algorithms to enable gesture-controlled and voice-controlled bed movements, as well as continuous monitoring of vital parameters.

Gesture and Voice Control: The system utilizes hand gesture recognition and voice recognition technology to allow users to control the bed movements. Hand gestures captured by a camera module are processed using TensorFlow and Mediapipe libraries to recognize specific gestures, which are then translated into commands to control the DC motor. Voice commands captured through a microphone are processed using speech recognition algorithms to interpret the user's instructions and adjust the bed accordingly.

Bed Movement: The DC motor, controlled by the H-Bridge L293D motor driver, adjusts the position of the bed based on the received commands. By analyzing the recognized gestures or voice commands, the system determines whether the bed should move up,

down, or remain in its current position. The motor driver regulates the motor's speed and direction to achieve the desired movement.

Continuous Vital Parameter Monitoring: The system incorporates various sensors, such as heart rate, SpO2 and blood pressure, temperature, humidity, and air quality sensors, to continuously monitor the patient's vital parameters and environmental conditions. The sensor data is acquired by the microcontroller and processed in ThingSpeak accordingly. The obtained information is used for real-time monitoring and analysis, allowing healthcare professionals to assess the patient's health status and detect any abnormalities.

User Interface and Control: The system provides a user-friendly interface, which can be a graphical user interface (GUI) on a connected device or a voice-based interaction. Users can input their commands through hand gestures or voice commands, and the system responds accordingly by adjusting the bed position or providing real-time data on vital parameters.

Overall, the system enables individuals to control the bed movements using gestures or voice commands, while also providing continuous monitoring of vital parameters for improved healthcare management and patient comfort.

3.9 Overall Conclusions of the Chapter 3

In conclusion, Chapter 3 provides a detailed description of the hardware components, software tools, interfacing techniques, and power supply considerations used in the project. This information lays the foundation for the successful implementation of the gesture-controlled and voice-controlled bed system, facilitating intuitive control and continuous monitoring of the bed position and vital parameters.

Chapter-4

Results and Discussions

4.1 Simulation Results

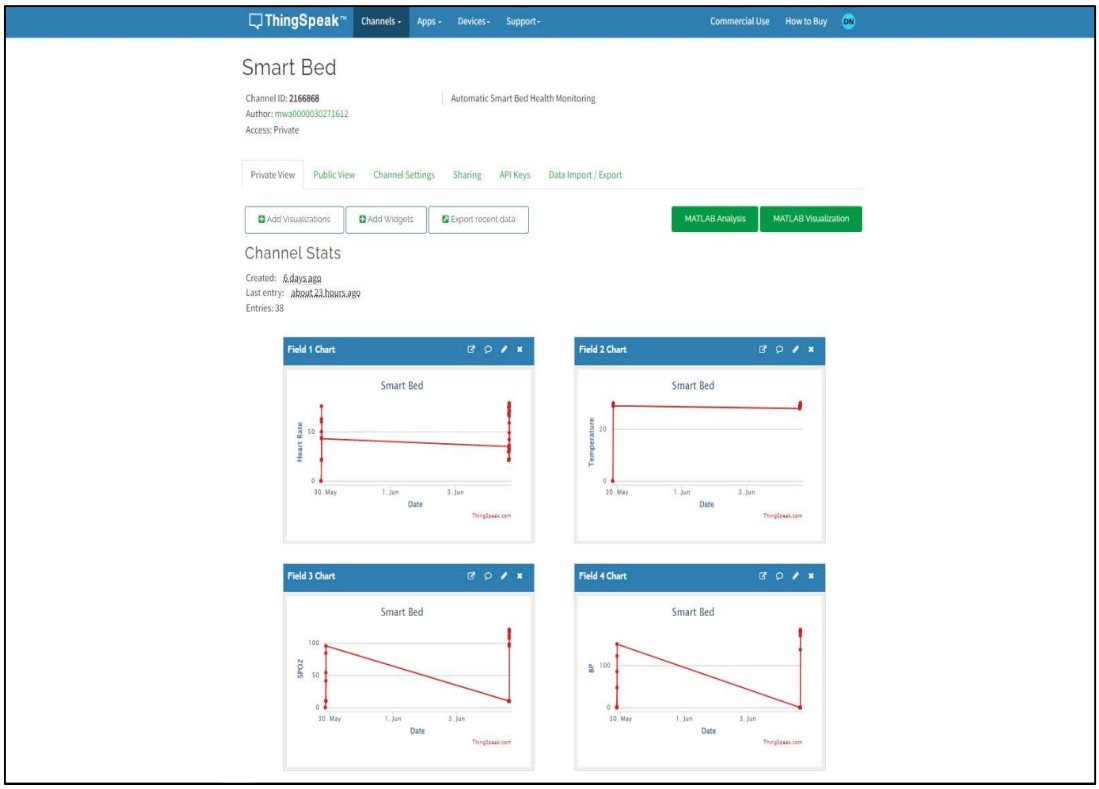


Fig. 9 Screenshot of thingspeak where all sensors vital are monitored & visualized

4.2 Experimental Results

Fig. 10 Voice command is given to system and bed movement is shown

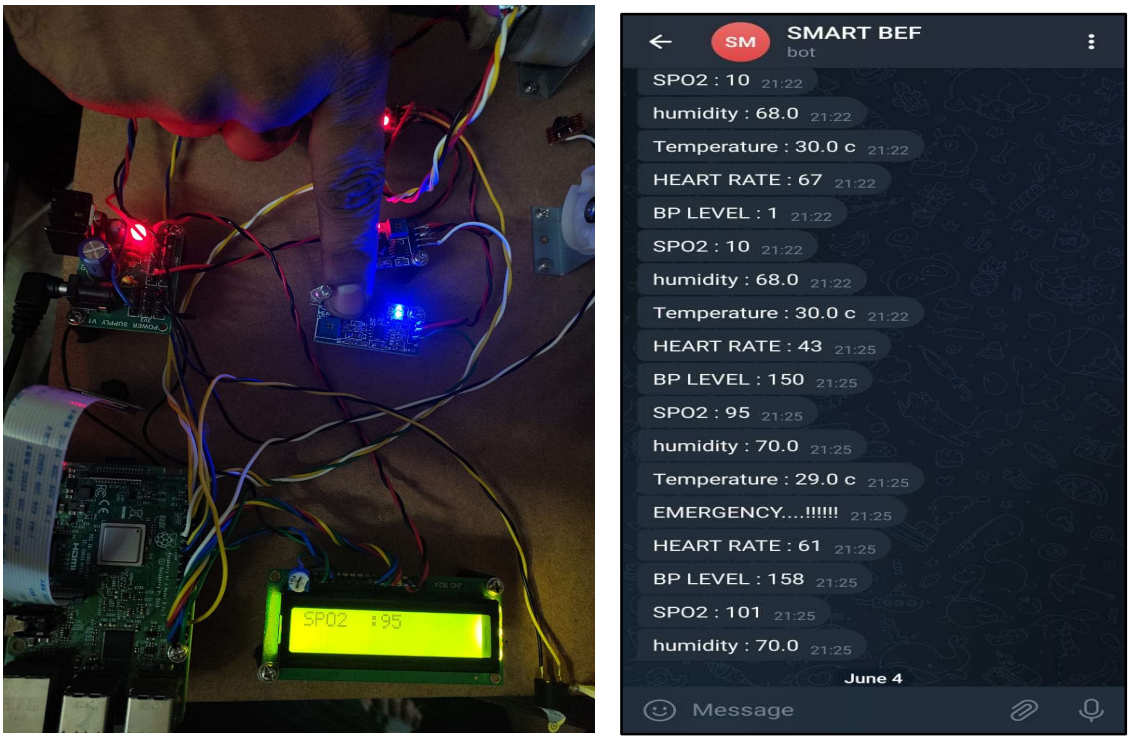


Fig. 11 Sensors is activated my placing finger on heart rate sensor & all parameters is shown on lcd and on telegram

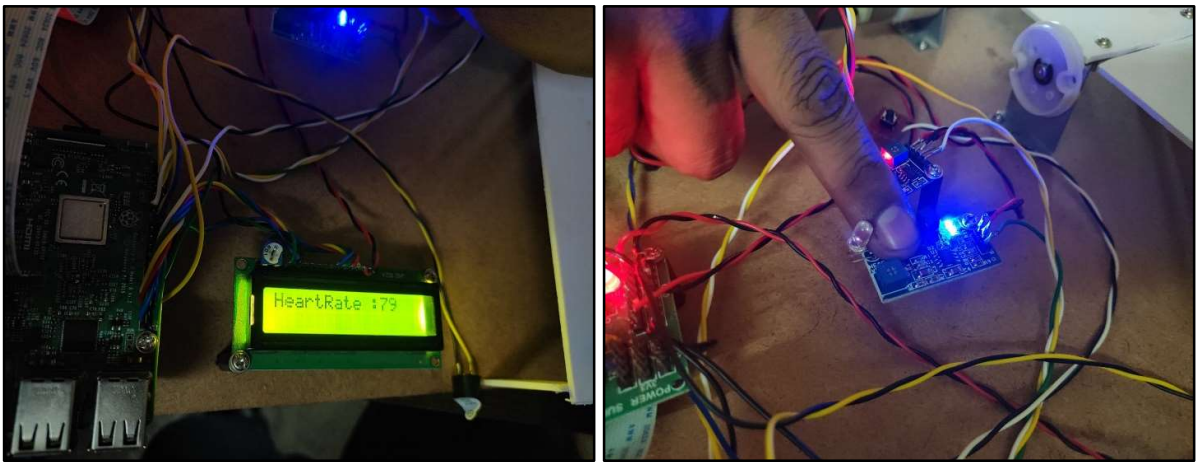


Fig. 12 placing finger on heart rate sensor & is shown on lcd

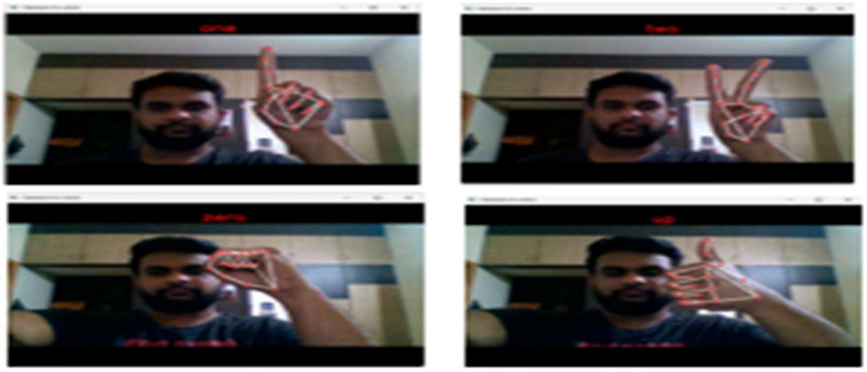


Fig. 13
Hand Gestures

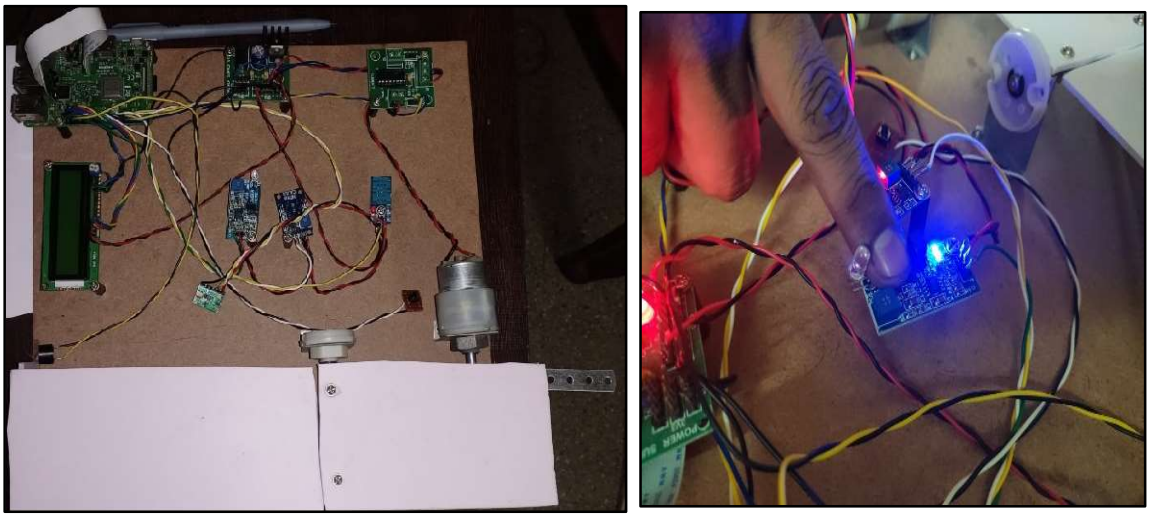


Fig. 14 Hardware Prototype of proposed system

Fig. 15 Bed Movement using Hand Gestures for up and down movement

4.3 Applications, Advantages, Limitations

4.3.1 Applications

- Accurate recording of health parameters for any type of patient.
- No need for assistance from others for bed positioning.
- Suitable for use in hospitals and at home for the elderly.
- Can be used in health checkup mini-vans for remote areas.
- Useful in small clinics and dispensaries with limited resources.

4.3.2. Advantages

- Enhanced independence: The gesture and voice-controlled bed system allows individuals with disabilities or paralysis to control the bed's movements without the need for assistance, promoting independence and a sense of empowerment.
- Improved comfort: Patients can easily adjust the bed's position using hand gestures or voice commands, enabling them to find the most comfortable position for resting or engaging in activities like reading or watching TV.
- Cost-effective solution: The Smart bed system reduces the need for full-time caregivers, which can significantly lower the overall cost of care for individuals with disabilities or paralysis. This makes it a more accessible option for those with financial constraints.
- Efficient healthcare monitoring: The bed system incorporates various sensors that continuously monitor vital parameters such as heart rate, temperature, blood oxygen levels, blood pressure, humidity levels, and air quality. This real-time data provides healthcare professionals with comprehensive insights into the patient's health status, facilitating timely interventions and personalized care.
- Proactive healthcare approach: By gathering and analyzing vital health data, the Smart bed system enables early detection of abnormalities or changes in the patient's condition. This allows healthcare providers to intervene promptly and prevent potential complications.
- Emergency alert system: The inclusion of an emergency button and a built-in buzzer ensures that patients can quickly alert healthcare staff in case of any discomfort or distress. This immediate notification system helps in providing timely assistance and appropriate medical attention.

4.3.3 Limitations

- The system may face issues with irrelevant objects overlapping with the hand during recognition, leading to incorrect detection and response.
- If the objects in the background are larger than the hand, the system may extract the wrong object instead of the hand, causing incorrect detection and response.
- Recognition performance may decrease if the distance between the user and the camera is greater than 1.5 meters, leading to incorrect detection and response.
- The system has several limitations that restrict its application, such as gesture recognition being restricted to the right hand only, the arm being required to be vertical, the palm facing the camera, and the background being plain and uniform. These limitations may impact the system's effectiveness in real-world scenarios.

4.3 Conclusions of the chapter-4

In conclusion, Chapter 4 highlights the successful implementation of the gesture-controlled and voice-controlled bed system, presenting the experimental results and discussing its application advantages and limitations. The outcomes demonstrate the system's potential to enhance user independence, improve healthcare management, and contribute to a more comfortable and proactive patient care experience.

Chapter-5

Conclusions, Future Work & Scope of Project

5.1 Conclusions

In conclusion, the integration of gesture control and voice recognition technology in the design of a patient bed system has the potential to revolutionize healthcare and improve the quality of life for individuals with disabilities, paralysis, and the elderly. By eliminating the need for full-time caregivers and allowing patients to control the bed's positioning using hand gestures and voice commands, this innovative solution promotes independence and enhances comfort. The incorporation of sensors for monitoring vital health parameters ensures continuous and accurate assessment of the patient's well-being, enabling early detection of any abnormalities and timely intervention. Moreover, the system's ability to promptly notify healthcare providers and family members in case of emergencies ensures that appropriate actions can be taken immediately, reducing the risks of adverse outcomes. The project addresses the existing limitations in healthcare by offering a user-friendly, efficient, and cost-effective solution that can be utilized in hospitals, home settings, and even remote areas. With its potential to enhance patient care, promote independence, and optimize resource allocation, this gesture-controlled and voice-activated bed system holds great promise for the future of healthcare.

5.2 Scope for future works in this domain

The gesture-controlled and voice-controlled bed system presents a promising solution for enhancing patient care and independence. In the future, several areas can be explored to further advance this domain:

Refinement of Gesture Recognition: Improving the accuracy and robustness of hand gesture recognition algorithms will enhance the system's usability and responsiveness, allowing for more precise and intuitive control of bed movements.

Integration of Machine Learning: Employing machine learning techniques can enable the system to learn and adapt to individual user gestures, further enhancing gesture recognition performance and customization.

Expansion of Sensor Capabilities: Integrating additional sensors, such as respiratory rate sensors or movement sensors, can provide a more comprehensive picture of the patient's health and well-being, enabling proactive healthcare interventions.

Smart Home Integration: Integrating the bed system with smart home technology can create a connected ecosystem where the bed can interact with other devices, such as lighting, temperature control, or entertainment systems, to create a more personalized and comfortable environment for patients.

Clinical Trials and Validation: Conducting extensive clinical trials and validation studies will help assess the system's effectiveness, usability, and impact on patient outcomes, paving the way for its widespread adoption in healthcare settings.

By focusing on these areas, future research and development efforts can further improve the gesture-controlled and voice-controlled bed system, making it a valuable tool in enhancing patient care, comfort, and independence.

5.3 Outcome of the project works

The outcome of the project is a gesture-controlled and voice-controlled bed system that allows patients to independently adjust their bed positions using hand gestures and voice commands. The system incorporates a range of sensors for continuous monitoring of vital parameters and environmental conditions. It enables real-time data acquisition and analysis, providing healthcare professionals with valuable insights into patients' health status. The system's user-friendly interface, accurate gesture recognition, and comprehensive monitoring capabilities enhance patient comfort, autonomy, and safety. Overall, the project successfully developed an innovative solution for improving patient care in healthcare settings.

References

[1] Sajida Fayyaz, Sidra Tahir. "Adjustment of Bed for a Patient through Gesture Recognition: An Image Processing Approach". IEEE 2018.

[2] Hui-Shyong Yeo & Byung-Gook Lee & Hyotaek Lim. "Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware" Springer Science+Business Media New York 2013.

[3] Diego A. Socolinsky, Andrea Selinger, and Joshua D. Neuheisel. "Face recognition with visible and thermal infrared imagery". Computer Vision and Image Understanding 91 (24 march 2003) 72–114.

[4] Elli Angelopoulou, Rana Molana, and Kostas Daniilidis. "Multispectral Skin Color Modeling". Computer Information Science Department, University of Pennsylvania Philadelphia, *PA.*

[5] Mokhtar M.Hasan, Pramod K. Mishra. "HSV Brightness Factor Matching for Gesture Recognition System". International Journal of Image Processing (IJIP), Volume (4): Issue (5) .Department of Computer Science Faculty of Science BHU-Varanasi
Uttar Pradesh-221005, India.

[6] Gaglio. S, G.L. Re, and M. Morana. "Human activity recognition process using 3-D posture data". IEEE Transactions on Human-Machine Systems. vol. 45, no.5, pp. 586-597, 2015.

[7] L. Jin, Y. Zhou, Z. Cheng, et al. "Magic ring: A finger-worn device for multiple appliances control using static finger gestures". Sensors. vol. 12, no. 5, pp. 5775C5790, 2012.

[8] Zarit, Benjamin D., Boaz J. Super, and Francis KH Quek, "Comparison of five-color models in skin pixel classification". Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems Proceedings. International Workshop, IEEE, 1999, pp. 58-63.

[9] Do, Martin, Tamim Asfour, and Rüdiger Dillmann, "Particle filter-based fingertip tracking with circular hough transform features". Threshold. 2, 2011, pp.2

[10] Ali Hamza, Rahul Anand, Pragati Shivhare, Anjali Gaurav "Hand Gesture Recognition Applications", International Journal of Interdisciplinary Research (IJIR) Vol-3, Issue-4, 2017 ISSN: 2454-1362

[11] Hernandez-Rebollar, J. L. Kyriakopoulos, N.Lindeman R.W, "A new instrumented approach for translating American Sign Language into sound and text" ,Proceedings of the Sixth IEEE International Conference, 2004

[12] Mehdi S.A., Khan Y. N., "Sign language recognition using sensor gloves", Proceedings of the 9th International Conference, Volume:5, IEEE Conference Publications, 2002.

[13] Kunal Kadam, Rucha Ganu, Ankita Bhosekar, Prof. S. D. Joshi, "American Sign Language Interpreter", Proceedings of the IEEE Fourth International Conference on Technology for Education, 2012

[14] Satjakarn Vutinuntakasame, "An Assistive Body Sensor Network Glove for Speech- and Hearing- Impaired Disabilities", Proceedings of the IEEE Computer Society International Conference on Body Sensor Networks, 2011.

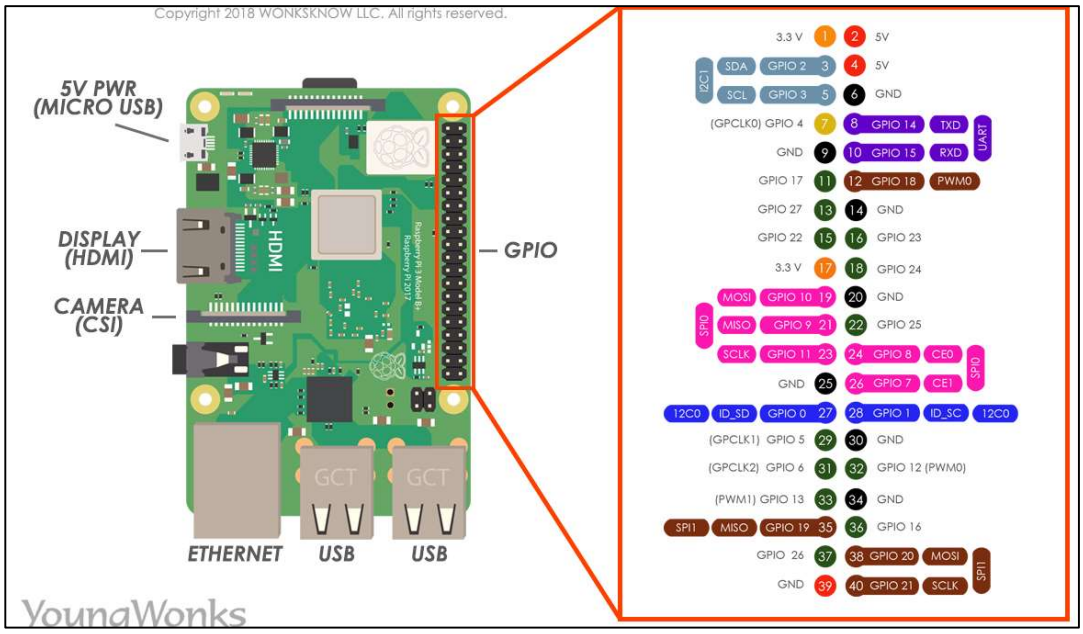
[5] Anbarasi Rajamohan, Hemavathy R., Dhanalakshmi M,"Deaf-Mute Communication Interpreter", International. Journal of Scientific Engineering and Technology (ISSN : 2277-1581) Volume 2 Issue 5, pp : 336-341, 1 May 2013

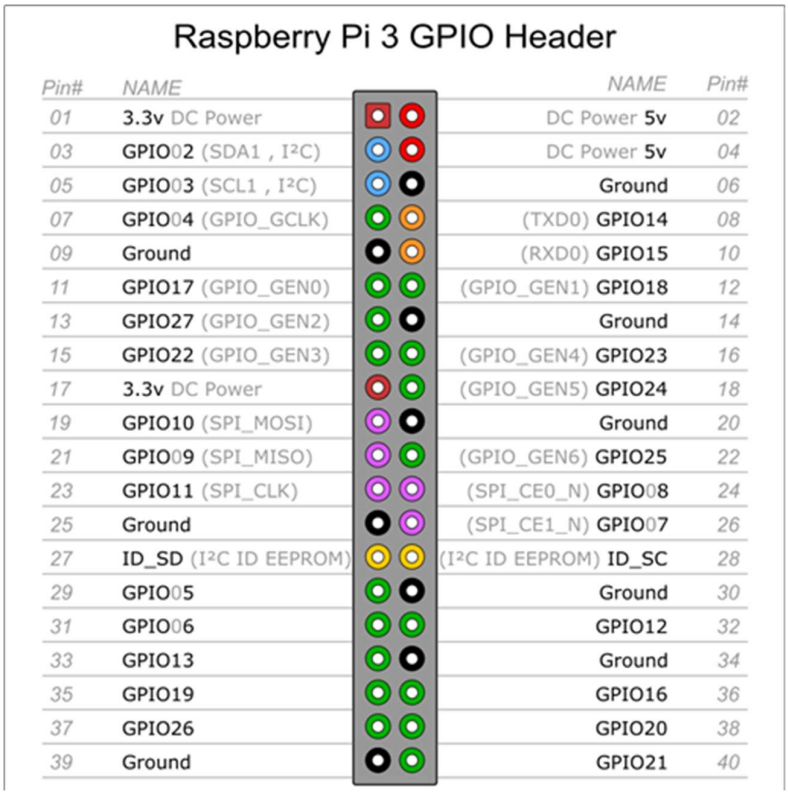
[16] K.C. Shriharipriya and K. Arthy, "Flex sensor based hand gesture recognition system", Proceedings of International Journal Of Innovative Research And Studies (IJIRS), Vellore, India, May 2013.

Appendix

Hardware Used: -

a) Raspberry Pi 3 B:





Raspberry Pi 3 GPIO Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I²C)	DC Power 5v	04
05	GPIO03 (SCL1 , I²C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)	(I²C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Raspberry Pi 3 Model B is the earliest model of the third-generation Raspberry Pi. It replaced Raspberry Pi 2 Model B in February 2016. See also Raspberry Pi 3 Model B+, the latest product in the Raspberry Pi 3 range.

• Quad Core 1.2GHz Broadcom BCM2837 64bit CPU

• 1GB RAM

• BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board

• 100 Base Ethernet

• 40-pin extended GPIO

• 4 USB 2 ports

• 4 Pole stereo output and composite video port

• Full size HDMI

• CSI camera port for connecting a Raspberry Pi camera

• DSI display port for connecting a Raspberry Pi touchscreen display

• Micro SD port for loading your operating system and storing data

• Upgraded switched Micro USB power source up to 2.5A

Raspberry Pi 3 Model B: - Single-board computer with wireless LAN and Bluetooth connectivity.

b) Bluetooth BC417 Module: -

Grove - Serial Bluetooth v3.0 is based on CSR BC417, designed for transparent wireless serial connection setup.

Features

- Complete Bluetooth Solution: Complete 2.4GHz radio transceiver and baseband with CSR Bluecore 04-Audio ROM, single chip bluetooth system with CMOS.
- Excellent power consumption: Low power standby modes to enable high efficient power management.
- High performance radio transceiver: Up to +4dBm RF transmit power with power level control.

- Grove unified socket: The sensor ensures "Plug and Play"

Description

Fig. 16 Bluetooth IC

Grove - Serial Bluetooth is an easy to use module compatible with the existing Grove Base Shield, based on CSR-BC417 and designed for transparent wireless serial connection setup. The serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR(Enhanced Data Rate) 2Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature).It has the smallest footprint of 12.7mm x 27mm. Hope it will simplify your overall design/development cycle.

Technical details

Dimensions	140mm x85mm x11.6mm
Weight	G.W 10g
Battery	Exclude
Operating Voltage	5.0VDC
Data Rate	2Mbps
RF Transmit Power (Max)	+4dBm
Sensitivity	-80dBm

Table 1 Technical details of Bluetooth module

c) H-Bridge L293D Motor Driver: -

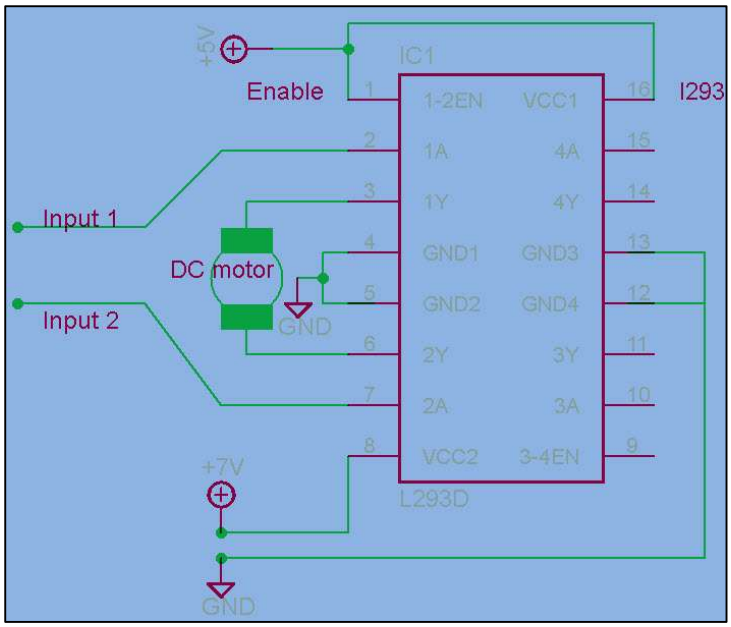
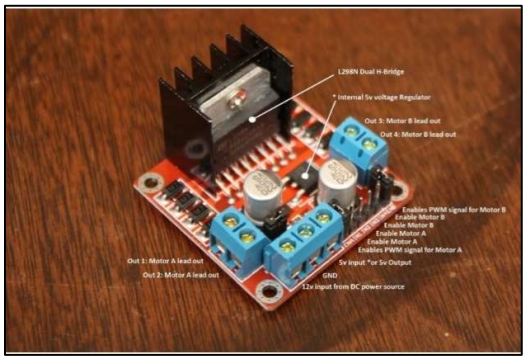


Fig 17 H Bridge Motor Control Circuit Using L293d IC

The IC LM293D consists of 4-i/p pins where, pin2 and 7 on the left side of the IC and Pin 10 and 15 on the right side of the IC. Left input pins on the IC will control the rotation of a motor. Here, the motor is connected across side and right i/p for the motor on the right hand side. This motor rotates based on the i/ps we provided across the input pins as Logic 0 and Logic 1.

Fig. 18 H Bridge Module



When a motor is connected to the o/p pins 3 and 6 on the left side of the IC. For rotating of the motor in clockwise direction, then the i/p pins have to be provided with Logic 0 and Logic 1.

When Pin-2= logic 1 & pin-7 = logic 0, then it rotates in clockwise direction.

Pin-2=logic 0 & Pin7=logic 1, then it rotates in anti clock direction

Pin-2= logic 0 & Pin7=logic 0, then it is idle (high impedance state)

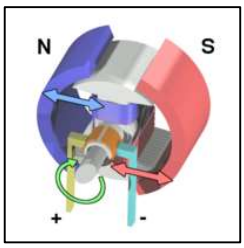
Pin-2= logic 1 & Pin7=logic 1, then it is idle

In a similar way the motor can also operate across input pin-15 and pin-10 for the motor on the right hand side.

The L4293D motor driver IC deals with huge currents, due to this reason, this circuit uses a heat sink to decrease the heat. Therefore, there are 4-ground pins on the L293D IC. When we solder these pins on the PCB (printed circuit board), then we can get a huge metallic area between the ground pins where the heat can be produced.

d) DC Motor: -

Fig. 19 DC Motor



A **DC motor** is any of a class of rotary electrical motors that converts direct current (DC) electrical energy into mechanical energy. The most common types rely on the forces produced by induced magnetic fields due to flowing current in the coil. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in part of the motor.

DC motors were the first form of motors widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor, a lightweight brushed motor used for portable power tools and appliances can operate on direct current and alternating current. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent

of power electronics has made replacement of DC motors with AC motors possible in many applications.

e) Raspberry Pi Camera: -

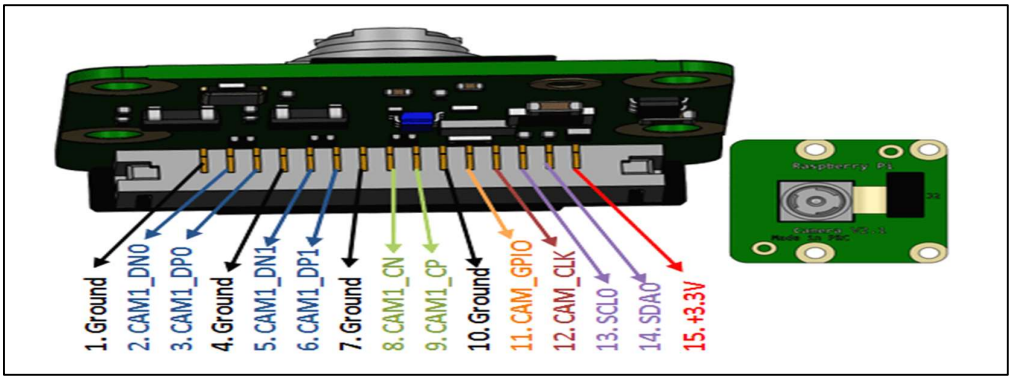


Fig. 20 Raspberry Camera 5MP

15-Pin Connector

The 15-pin connector is the most used one. Unless otherwise indicated, It's the "default" connector in terms of Raspberry Pi camera. You will see it in mainstream Raspberry Pi products such as the Model A&B series as well as V1&V2 camera modules.

Raspberry pi camera pinout (15-pin)

Pin #	Name	Description
1	GND	Ground
2	CAM_D0_N	MIPI Data Lane 0 Negative
3	CAM_D0_P	MIPI Data Lane 0 Positive
4	GND	Ground
5	CAM_D1_N	MIPI Data Lane 1 Negative
6	CAM_D1_P	MIPI Data Lane 1 Positive

Pin #	Name	Description
7	GND	Ground
8	CAM_CK_N	MIPI Clock Lane Negative
9	CAM_CK_P	MIPI Clock Lane Positive
10	GND	Ground
11	CAM_IO0	Power Enable
12	CAM_IO1	LED Indicator
13	CAM_SCL	I2C SCL
14	CAM_SDA	I2C SDA
15	CAM_3V3	3.3V Power Input

Table 2 Camera 15 pin connector pinout

f) 12V & 5V Dual Power Supply: -

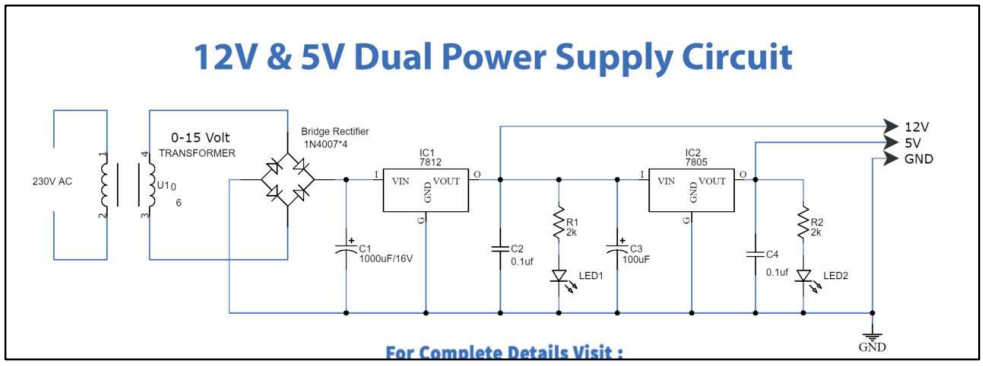


Fig.21 The dual power supply circuit provides regulated DC output after processing the input

Transformer: The transformer is a basic part of a power supply circuit. It steps down the input signal of 110, 230V AC to 15V AC signal. A device that works on the principle of electromagnetic induction is called a

transformer. It transfers electrical energy from one circuit to another. It generally increases or decreases the AC voltage.

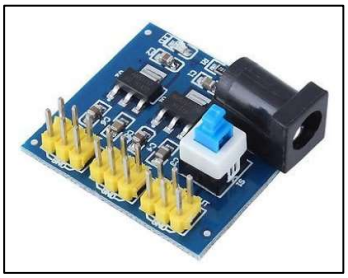


Fig.22 Dual Supply board

Rectifier: The rectifier circuit mainly consists of diodes, it converts the bidirectional AC signal into a unidirectional DC signal by eliminating one side of the AC signal and allowing only unidirectional current to pass through.

Filter: The DC signal after rectification consists of AC ripples and voltage fluctuations is fed into capacitor C1 which removes most of the AC ripples from the signal.

IC 7812: The filtered signal is fed into voltage regulation IC which receives the unregulated DC voltage at its input and provides regulated 12V DC at its output.

In the end, the capacitor C2 is used to filter out the high-frequency ripples remaining in the output signal to provide a ripple-free 12V DC output.

IC 7805: Further the 12V DC signal is again filtered and fed into the IC 7805 which takes in the input signal and provides regulated 5V DC at its output and the signal is again filtered to remove the high-frequency ripples by using capacitor C4. LED1 and LED2 indicate the presence of 12V DC voltage and 5V DC voltage at the output respectively.

g) <u>USB TO UART CONVERTER</u>: -

The CP2102 USB to UART/TTL Converter Module provides a serial port for your computer, i.e. a virtual COM port to connect to a serial device. The IC used CP2102 is stable and reliable. It supports baud rates from 300bps to

1.5Mbps and driver support for a wide range of Operating Systems. This is a great little tool for embedded systems that require a serial connection to a computer. The board can simply attach to a USB bus and will appear as a standard COM port.

ThisCP2102 doesn't require any external oscillator, it onboard-board voltage regulator, and it even uses a reprogrammable internal EEPROM for the device description. The full hardware UART has flow control for baud rates from 300bps to 921600bps.

This breakout also allows you to connect the TX/RX pins of your favorite microcontroller or serial application to the RX/TX pins of the breakout, creating a simple serial cable replacement.

PIN DESCRIPTION

- Pin 1: 3V3 3.3 volts
- Pin 2: TXD Transmit Data
- Pin 3: RXD Receive Data
- Pin 4: GND Ground
- Pin 5: +5V 5 volts

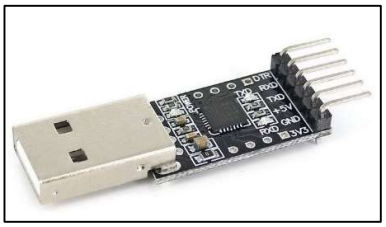


Fig.23 USB Port Module

SPECIFICATION:

Model	CP2102 USB to TTL UART Converter
USB Version	High speed USB version 2.0
Baud Rate	300 bps to 1.5 Mbps
Output Pins Voltage	300 bps to 1.5 Mbps.
Dimensions	5 x 2 x 1 cms
Weight	5 grams

h) DHT11 Sensor: -

The **DHT11** is a commonly used **Temperature and humidity sensor that** comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data.

DHT11 Pinout Configuration

No:	Pin Name	Description
For DHT11 Sensor		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	NC	No Connection and hence not used
4	Ground	Connected to the ground of the circuit
For DHT11 Sensor module		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	Ground	Connected to the ground of the circuit

Table 3 DHT11 Pinout

DHT11 Specifications

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%

- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

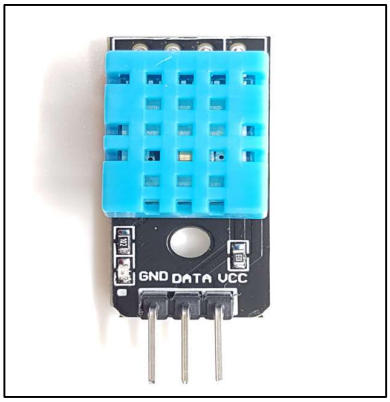


Fig24
DHT11 Sensor

DHT11 digital temperature and humidity sensor is a calibrated digital signal output of the temperature and humidity combined sensor. It uses dedicated digital modules capture technology and the temperature and humidity sensor technology to ensure that products with high reliability and excellent long-term stability. Sensor includes a resistive element and a sense of wet NTC temperature measurement devices, and with a high-performance 8-bit microcontroller connected.

i) FC-22-1 Harmful Gas Detector Sensor: -

- Model: FC-22-1

- Quantity: 1

- Color: Blue

- Material: PCB board

- Range: 10~1000ppm

- High quality double panel design, with power and TTL signal output indicator

- With DO switch signal (TTL) output and AO analog signal output

- TTL output effective signal for low level (when output lower than usually the signal light, can be directly

meet SCM or relay module)

- Analog output voltage: 0~5V, the higher concentration, the higher voltage

- High sensitivity for the harmful gases detection

- There are four screw holes for positioning

- Product size: 32mm (L) x 20mm (W) x 22mm (H)

- Long service life and reliable stability, fast response and recovery characteristics

- Working voltage: 5V

- Suitable for home / industry carbon monoxide, coal gas, natural gas, smoke and other harmful

monitoring gases detection

- Before use, please heat the board for about 20 seconds

- Packing list:

- 1 x Sensor module

Dimensions: 1.38 in x 0.79 in x 0.91 in (3.5 cm x 2.0 cm x 2.3 cm)

Weight: 0.25 oz (7 g

The voltage that the sensor outputs changes accordingly to the smoke/gas level that exists in the atmosphere. The sensor outputs a voltage that is proportional to the concentration of smoke/gas.In other words, the relationship between voltage and gas concentration is the following:

- **The greater** the gas concentration, **the greater** the output voltage
- **The lower** the gas concentration, **the lower** the output voltage

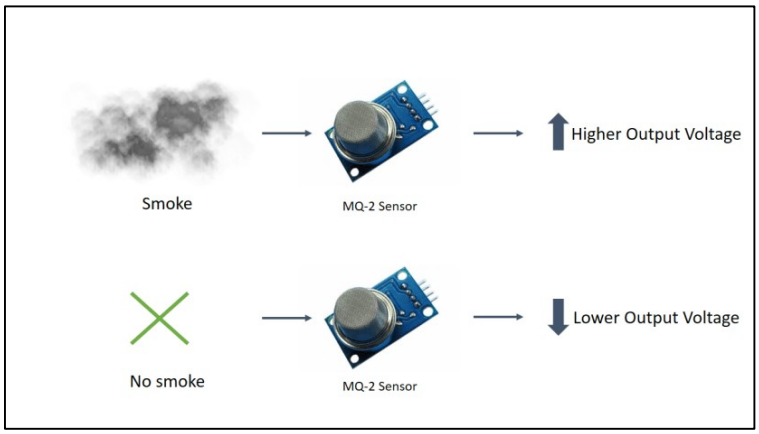


Fig 25 Gas Sensor Working

The output can be an analog signal (A0) that can be read with an analog input of the Arduino or a digital output (D0) that can be read with a digital input of the Arduino.

Pin Wiring

The MQ-2 sensor has 4 pins.

Pin	**Wiring to Arduino Uno**
A0	Analog pins
D0	Digital pins
GND	GND
VCC	5V

Table 4 Gas Sensor Pinout

j) <u>MAX30102 Sp02 & Pulse Oximetry Sensor:</u> - The MAX30102 is a very versatile sensor and it can also measure body temperature other than heart rate and blood oxygen level. This is a sensor designed by Analog Devices and features two LEDs (one Infrared and one Red), a photodetector, optics, and low-noise signal processing unit to detect pulse oximetry (SpO2) and heart rate (HR) signals.

Fig. 26 Sp02 Sensor

The main idea is that you shine a single LED at a time and check the amount of light that is getting reflected back to the sensor. Based on the reflection you can determine the blood oxygen level and heart rate. This is a very short explanation of how this sensor works and we will elaborate more later in the article. But for now, let's check other parameters of the sensor

MAX30102 Pulse Oximeter and Heart Rate Sensor Pinout

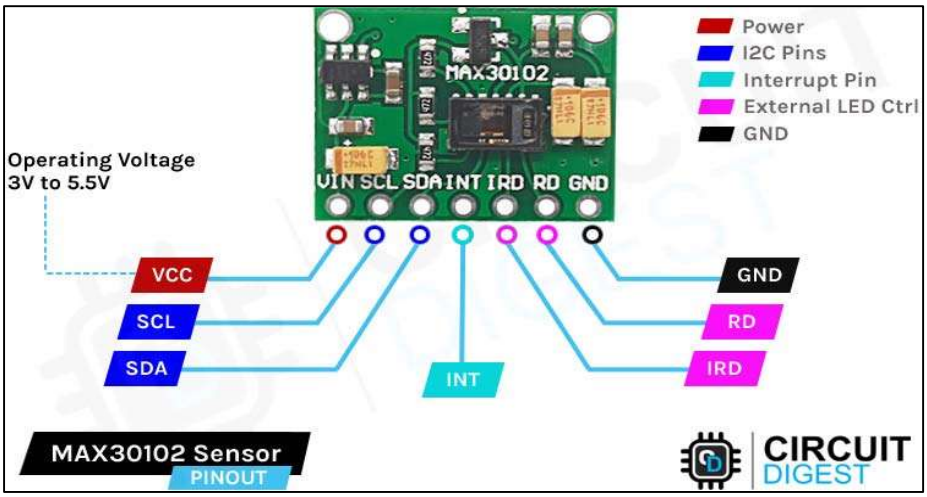


Fig.27 Max30102 Pinout

The MAX30102 Digital Pulse Oximeter and Heart Rate Sensor, and this module have 7 pins VCC, SCL, SDA, INT, IRD, RD, and GND. All the pins of this sensor module are digital, except VCC and Ground. The **Pinout of the Pulse Oximeter and Heart Rate Sensor** is shown below:

VCC is the power supply pin of the Pulse Oximeter and Heart Rate Sensor that can be connected to 3.3V or 5V of the supply.

SCL SCL stands for Serial Clock the master device pulses this pin at a regular interval to generate clock signal for communication. Connect to the A5 pin of the Arduino.

SDA Stands for Serial Data, through this pin data exchange happens between two devices. Connect this pin to the A4 pin of the Arduino.

INT is the Interrupt pin of the IC. This IC can be programmed to generate an Interrupt a on each pulse. This pin is open drain in nature, so it's pulled high through an onboard resistor. When there is an interrupt this pin goes low and stays low until the interrupt is cleared.

IRD This is the pin that is connected to the IR LED of the Module. This module has a LED driver built in. use this pin if you want to manually drive the LED with code, otherwise leave the pin open.

RD this pin is similar to the IRD pin the only difference is that a RED LED is connected to this pin. Leave it untouched if you don't want to drive the LED yourself.

GND is the ground pin of the pulse Oximeter and Heart Rate Sensor module and it should be connected to the ground pin of the Arduino.

Operating Power of MAX30102 Pulse Oximeter and Heart Rate Sensor

The operating voltage of the module is 3.3V to 5V with 600uA of Maximum current draw. But If we are talking about the MAX30102 IC on the module board the IC requires two different supply voltages for stable operation, which is why in the module board you can find two linear low dropout regulators. One is for 3.3V and the other one is for 1.8V. You can check out the parts marking section for better understanding. One of the most interesting features of this module is its power consumption, MAX30102 consumes less than 600µA during measurement. And it consumes only 0.7µA when it's in standby mode. Because of this low power consumption, this device can be used in battery powered applications. Other than that this IC offers a I2C Interface for communication, a FIFO buffer for holding and sampling 16 different readings, and an ON-Chip Temperature Sensor to measure the body temperature.

LCD Display: -

16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. But the most used one is the 16*2 LCD, hence we are using it here.All the above mentioned LCD display will have 16 Pins and the programming approach is also the same and hence the choice is left to you. Below is the **Pinout and Pin Description of 16x2 LCD Module**:

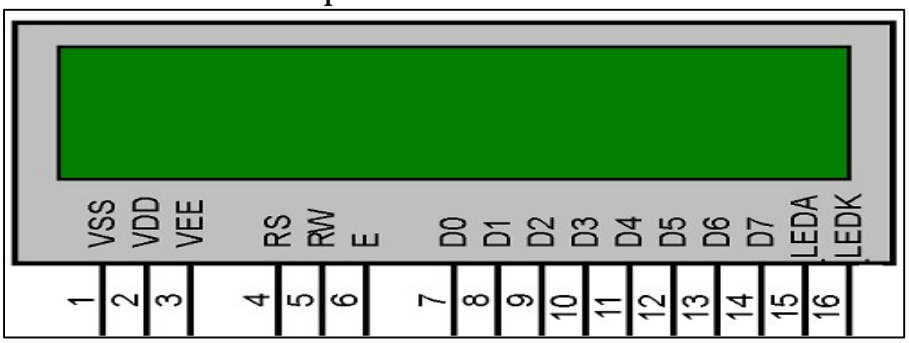


Fig 28 LCD Pinout

Sr. No	Pin No.	Pin Name	Pin Type	Pin Description	Pin Connection
1	Pin 1	Ground	Source Pin	This is a ground pin of LCD	Connected to the ground of the MCU/ Power source
2	Pin 2	VCC	Source Pin	This is the supply voltage pin of LCD	Connected to the supply pin of Power source
3	Pin 3	V0/VEE	Control Pin	Adjusts the contrast of the LCD.	Connected to a variable POT that can source 0-5V
4	Pin 4	Register Select	Control Pin	Toggles between Command/Data Register	Connected to a MCU pin and gets either 0 or 1. 0 -> Command Mode 1-> Data Mode
5	Pin 5	Read/Write	Control Pin	Toggles the LCD between Read/Write Operation	Connected to a MCU pin and gets either 0 or 1. 0 -> Write Operation 1-> Read Operation
6	Pin 6	Enable	Control Pin	Must be held high to perform Read/Write Operation	Connected to MCU and always held high.
7	Pin 7-14	Data Bits (0-7)	Data/Command Pin	Pins used to send Command or data to the LCD.	<u>In 4-Wire Mode</u> Only 4 pins (0-3) is connected to

					MCU In 8-Wire Mode All 8 pins(0-7) are connected to MCU
8	Pin 15	LED Positive	LED Pin	Normal LED like operation to illuminate the LCD	Connected to +5V
9	Pin 16	LED Negative	LED Pin	Normal LED like operation to illuminate the LCD connected with GND.	Connected to ground

Table 5 LCD Pinout

Code Developed

For Hand Gesture Recognition: -

 # import necessary packages

```
import cv2
import numpy as np
import mediapipe as mp
from tensorflow.keras.models import load_model
import time
#!/usr/bin/env python
import time
import serial
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)

IN1=2
IN2=3
GPIO.setup(IN1,GPIO.OUT)
GPIO.setup(IN2,GPIO.OUT)

GPIO.output(IN1,False)
GPIO.output(IN2,False)



def level1():
   print('move down')
   GPIO.output(IN1,True)
   GPIO.output(IN2,False)
   time.sleep(0.2)
   GPIO.output(IN1,False)
   GPIO.output(IN2,False)

def flat():
```

```
    print('move up')
    GPIO.output(IN1,False)
    GPIO.output(IN2,True)
    time.sleep(0.2)
    GPIO.output(IN1,False)
    GPIO.output(IN2,False)
count1=0
count2=0

mpHands = mp.solutions.hands

hands = mpHands.Hands(max_num_hands=1, min_detection_confidence=0.7)

mpDraw = mp.solutions.drawing_utils

# Load the gesture recognizer model
model = load_model('mp_hand_gesture')

# Load class names
f = open('gesture.names', 'r')
classNames = f.read().split('\n')
f.close()

print(classNames)

# Initialize the webcam
cap = cv2.VideoCapture(0)

while True:
    # Read each frame from the webcam
    _, frame = cap.read()

    x, y, c = frame.shape
```

```
    # Flip the frame vertically
    frame = cv2.flip(frame, 1)
    framergb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    # Get hand landmark prediction
    result = hands.process(framergb)

    # print(result)

    className = ''
    count = 0
    # post process the result
    if result.multi_hand_landmarks:
        landmarks = []
        #print(result.multi_hand_landmarks)
        for handslms in result.multi_hand_landmarks:
            for lm in handslms.landmark:
                # print(id, lm)
                count += 1

                lmx = int(lm.x * x)
                lmy = int(lm.y * y)

                landmarks.append([lmx, lmy])

            # Drawing landmarks on frames
            mpDraw.draw_landmarks(frame,handslms,
pHands.HAND_CONNECTIONS)

            # Predict gesture
            prediction = model.predict([landmarks])
            # print(prediction)
```

```
            classID = np.argmax(prediction)
            className = classNames[classID]



            if count1<2:

                if className == 'one':
                    flat()
                    count1=count1+1

            if count1>0:

                if className == 'two':
                    level1()
                    count1=count1-1

            # show the prediction on the frame
            cv2.putText(frame,          className,          (300,          50),
cv2.FONT_HERSHEY_SIMPLEX,
                    1, (0,0,255), 2 )

        # Show the final output
        cv2.imshow("Gesture to voice", frame)

        if cv2.waitKey(1) == ord('q'):
            break

# release the webcam and destroy all active windows
cap.release()
cv2.destroyAllWindows()
```

Code For Voice Recognition

```
#!/usr/bin/env python
import time
import serial
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
ser = serial.Serial(
  port='/dev/ttyUSB0',  #ttyAMA0 ttyS0
  baudrate = 9600,
  parity=serial.PARITY_NONE,
  stopbits=serial.STOPBITS_ONE,
  bytesize=serial.EIGHTBITS,
  timeout=1
)
IN1=2
IN2=3
GPIO.setup(IN1,GPIO.OUT)
GPIO.setup(IN2,GPIO.OUT)


GPIO.output(IN1,False)
GPIO.output(IN2,False)
import time
from gtts import gTTS
```

```
from mutagen.mp3 import MP3
import time
import pygame
def level1():
    print('move down')
    GPIO.output(IN1,True)
    GPIO.output(IN2,False)
    time.sleep(0.2)
    GPIO.output(IN1,False)
    GPIO.output(IN2,False)


def flat():
    print('move up')
    GPIO.output(IN1,False)
    GPIO.output(IN2,True)
    time.sleep(0.2)
    GPIO.output(IN1,False)
    GPIO.output(IN2,False)
count1=0
while True:
   # flat()
   x=ser.readline()
   x = x.decode('UTF-8','ignore')
   x = x.lower()
```

```
if len(x.strip())>0:
    print('You said {}'.format(x))
    y = 'say again'
    if count1<3:


        if x == 'move':
            flat()
            count1=count1+1
    if count1>0:


        if x == 'down':
            level1()
            count1=count1-1


    print(y)
```

Code For Health Monitoring Sensors

```
#!/usr/bin/env python
import time
import serial
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
import time
import max30100
import Adafruit_DHT
mx30 = max30100.MAX30100()
```

```
mx30.enable_spo2()
import telepot
import urllib.request
import requests
import threading
import json

import random
bot=telepot.Bot("6281868215:AAFxaP6Z4psULn59XUad-KN2z2EfHQoStUY")
sensor = Adafruit_DHT.DHT11
global heartRate,bp,spo2,temperature,humidity



pin = 4
LCD_RS = 21
LCD_E  = 20
LCD_D4 = 26
LCD_D5 = 19
LCD_D6 = 13
LCD_D7 = 6

# Define some device constants
LCD_WIDTH = 20    # Maximum characters per line
LCD_CHR = True
LCD_CMD = False

LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
LCD_LINE_3 = 0x90 # LCD RAM address for the 2nd line
LCD_LINE_4 = 0xD0 # LCD RAM address for the 2nd line

# Timing constants
E_PULSE = 0.0005
```

```
E_DELAY = 0.0005

####lcd end here###########

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

#GPIO.setup(22, GPIO.IN, pull_up_down=GPIO.PUD_UP)#Button to GPIO12
    # Use BCM GPIO numbers
GPIO.setup(LCD_E, GPIO.OUT)  # E
GPIO.setup(LCD_RS, GPIO.OUT) # RS
GPIO.setup(LCD_D4, GPIO.OUT) # DB4
GPIO.setup(LCD_D5, GPIO.OUT) # DB5
GPIO.setup(LCD_D6, GPIO.OUT) # DB6
GPIO.setup(LCD_D7, GPIO.OUT) # DB7

####################LCD#######################
def lcd_init():
  # Initialise display
  lcd_byte(0x33,LCD_CMD) # 110011 Initialise
  lcd_byte(0x32,LCD_CMD) # 110010 Initialise
  lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
  lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
  lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
  lcd_byte(0x01,LCD_CMD) # 000001 Clear display
  time.sleep(E_DELAY)

def lcd_byte(bits, mode):
  # Send byte to data pins
  # bits = data
  # mode = True  for character
  #        False for command
```

```
GPIO.output(LCD_RS, mode) # RS

# High bits
GPIO.output(LCD_D4, False)
GPIO.output(LCD_D5, False)
GPIO.output(LCD_D6, False)
GPIO.output(LCD_D7, False)
if bits&0x10==0x10:
  GPIO.output(LCD_D4, True)
if bits&0x20==0x20:
  GPIO.output(LCD_D5, True)
if bits&0x40==0x40:
  GPIO.output(LCD_D6, True)
if bits&0x80==0x80:
  GPIO.output(LCD_D7, True)

# Toggle 'Enable' pin
lcd_toggle_enable()

# Low bits
GPIO.output(LCD_D4, False)
GPIO.output(LCD_D5, False)
GPIO.output(LCD_D6, False)
GPIO.output(LCD_D7, False)
if bits&0x01==0x01:
  GPIO.output(LCD_D4, True)
if bits&0x02==0x02:
  GPIO.output(LCD_D5, True)
if bits&0x04==0x04:
  GPIO.output(LCD_D6, True)
if bits&0x08==0x08:
  GPIO.output(LCD_D7, True)
```

```
  # Toggle 'Enable' pin
  lcd_toggle_enable()

def lcd_toggle_enable():
  # Toggle enable
  time.sleep(E_DELAY)
  GPIO.output(LCD_E, True)
  time.sleep(E_PULSE)
  GPIO.output(LCD_E, False)
  time.sleep(E_DELAY)

def lcd_string(message,line):
  # Send string to display
  message = message.ljust(LCD_WIDTH," ")

  lcd_byte(line, LCD_CMD)

  for i in range(LCD_WIDTH):
    lcd_byte(ord(message[i]),LCD_CHR)
####################LCD HERE######################
BZ = 16
import Adafruit_DHT
GPIO.setup(BZ, GPIO.OUT)
GPIO.output(BZ, False)
gas= 17
HR_SENSOR = 12
GPIO.setup(HR_SENSOR,GPIO.IN)
GPIO.setup(gas,GPIO.IN)
lcd_init()
lcd_byte(0x01, LCD_CMD)
lcd_string(" SMART BED  ",LCD_LINE_1)
lcd_string("  SYSTEM    ",LCD_LINE_2)
time.sleep(3)
```

```
lcd_byte(0x01, LCD_CMD)
lcd_string(" HEALTH MONITORING ",LCD_LINE_1)
lcd_string("  SYSTEM    ",LCD_LINE_2)
time.sleep(2)
global tempFlag
global bpFlag
global hrFlag
tempFlag = 0
bpFlag   = 0
hrFlag   = 1
while True:
  if hrFlag == 1 :
    print('Hold The finger On sensor')
    lcd_byte(0x01, LCD_CMD)
    lcd_string('Hold finger On sensor',LCD_LINE_1)
    lcd_string('On sensor',LCD_LINE_2)
    time.sleep(1)
    #time.sleep(1)
    sensorCounter = 0
    startTime     = 0
    endTime       = 0
    rateTime      = 0
  while sensorCounter < 1 and  hrFlag == 1:
    if (GPIO.input(HR_SENSOR)):
      if sensorCounter == 0:
        startTime = int(round(time.time()*1000))
        #print startTime
      sensorCounter = sensorCounter + 1
      #print sensorCounter
      while(GPIO.input(HR_SENSOR)):
        if hrFlag == 0:
          break
        pass
```

```
    time.sleep(1)
    endTime  = int(round(time.time()*1000))
    #print endTime
    rateTime = endTime - startTime
    #print rateTime
    rateTime = rateTime / sensorCounter
    heartRate = (60000 / rateTime) #/ 3
    heartRate = abs(heartRate)
    heartRate=int(heartRate+20)
    print (heartRate)
    lcd_byte(0x01, LCD_CMD)
    lcd_string("HeartRate :{}".format(heartRate),LCD_LINE_1)
    bot.sendMessage("1210994566",str("HEART RATE : {}".format(heartRate)))
    time.sleep(1)
    if heartRate<40:
        GPIO.output(BZ, True)
        lcd_byte(0x01, LCD_CMD)
        lcd_string("HEARTRATE IS ABNORMAL",LCD_LINE_1)
        time.sleep(1)
        GPIO.output(BZ, False)
#def SPO2():
    mx30.read_sensor()
    mx30.ir, mx30.red
    bp = int(mx30.ir / 100)
    spo2 = int(mx30.red / 100)+10
    if mx30.ir != mx30.buffer_ir :
        print("BP LEVEL:",bp);
        bot.sendMessage("1210994566",str("BP LEVEL : {}".format(bp)))
        lcd_byte(0x01, LCD_CMD)
        lcd_string("BP :{}".format(bp),LCD_LINE_1)
        time.sleep(1)
        if bp> 170:
```

```
            GPIO.output(BZ, True)
            lcd_byte(0x01, LCD_CMD)
            lcd_string("BP IS MORE",LCD_LINE_1)
            time.sleep(1)
            GPIO.output(BZ, False)
        if mx30.red != mx30.buffer_red:
            print("SPO2:",spo2);
            lcd_byte(0x01, LCD_CMD)
            lcd_string("SPO2  :{}".format(spo2),LCD_LINE_1)
            bot.sendMessage("1210994566",str("SPO2 : {}".format(spo2)))
            time.sleep(1)
            if spo2<80:
                GPIO.output(BZ, True)
                lcd_byte(0x01, LCD_CMD)
                lcd_string("SPO2 IS LESS",LCD_LINE_1)
                time.sleep(1)
                GPIO.output(BZ, False)
        time.sleep(1)
    #def HT_Reading():
        humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
        print('Humidity: {} and Temperature: {}'.format(humidity, temperature))

        lcd_byte(0x01, LCD_CMD)
        lcd_string("humidity  :{}".format(humidity),LCD_LINE_1)
        lcd_string("Temperature  :{}".format(temperature),LCD_LINE_1)
        bot.sendMessage("1210994566",str("humidity : {}".format(humidity)))
        bot.sendMessage("1210994566",str("Temperature : {} c".format(temperature)))
        time.sleep(1)
        if temperature>32:
            GPIO.output(BZ, True)
            lcd_byte(0x01, LCD_CMD)
            lcd_string("MORE TEMPERATURE",LCD_LINE_1)
            time.sleep(1)
```

```
            GPIO.output(BZ, False)

#def air():
  if(GPIO.input(gas)==True):
    print("air quality is bad")
    lcd_byte(0x01, LCD_CMD)
    lcd_string("BAD AIR QUALITY",LCD_LINE_1)
    bot.sendMessage("1210994566",str("Air quality is bad "))
    time.sleep(2)


URl='https://api.thingspeak.com/update?api_key=4B35650D21X8YEEF&field
1=10&field2=20&field3=30&field4=40&field5=50'
  KEY='4B35650D21X8YEEF'

HEADER='&field1={}&field2={}&field3={}&field4={}&field5={}'.format(heartRa
te,temperature,spo2,bp,humidity)
  NEW_URL = URl+KEY+HEADER
  print(NEW_URL)
  data=urllib.request.urlopen(NEW_URL)
  print(data)
```

Code For MAX30100 Sensor

```
import smbus

INT_STATUS   = 0x00  # Which interrupts are tripped
INT_ENABLE   = 0x01  # Which interrupts are active
FIFO_WR_PTR  = 0x02  # Where data is being written
OVRFLOW_CTR  = 0x03  # Number of lost samples
FIFO_RD_PTR  = 0x04  # Where to read from
FIFO_DATA    = 0x05  # Ouput data buffer
MODE_CONFIG  = 0x06  # Control register
SPO2_CONFIG  = 0x07  # Oximetry settings
```

```
LED_CONFIG   = 0x09  # Pulse width and power of LEDs
TEMP_INTG    = 0x16  # Temperature value, whole number
TEMP_FRAC    = 0x17  # Temperature value, fraction
REV_ID       = 0xFE  # Part revision
PART_ID      = 0xFF  # Part ID, normally 0x11

I2C_ADDRESS  = 0x57  # I2C address of the MAX30100 device


PULSE_WIDTH = {
    200: 0,
    400: 1,
    800: 2,
   1600: 3,
}

SAMPLE_RATE = {
    50: 0,
   100: 1,
   167: 2,
   200: 3,
   400: 4,
   600: 5,
   800: 6,
  1000: 7,
}

LED_CURRENT = {
       0: 0,
     4.4: 1,
     7.6: 2,
    11.0: 3,
    14.2: 4,
```

```
    17.4: 5,
    20.8: 6,
    24.0: 7,
    27.1: 8,
    30.6: 9,
    33.8: 10,
    37.0: 11,
    40.2: 12,
    43.6: 13,
    46.8: 14,
    50.0: 15
}

def _get_valid(d, value):
    try:
        return d[value]
    except KeyError:
        raise KeyError("Value %s not valid, use one of: %s" % (value, ', '.join([str(s) for s
in d.keys()])))

def _twos_complement(val, bits):
    """compute the 2's complement of int value val"""
    if (val & (1 << (bits - 1))) != 0: # if sign bit is set e.g., 8bit: 128-255
        val = val - (1 << bits)
    return val

INTERRUPT_SPO2 = 0
INTERRUPT_HR = 1
INTERRUPT_TEMP = 2
INTERRUPT_FIFO = 3

MODE_HR = 0x02
MODE_SPO2 = 0x03
```

```
class MAX30100(object):

    def __init__(self,
            i2c=None,
            mode=MODE_HR,
            sample_rate=100,
            led_current_red=11.0,
            led_current_ir=11.0,
            pulse_width=1600,
            max_buffer_len=10000
            ):

        # Default to the standard I2C bus on Pi.
        self.i2c = i2c if i2c else smbus.SMBus(1)

        self.set_mode(MODE_HR)  # Trigger an initial temperature read.
        self.set_led_current(led_current_red, led_current_ir)
        self.set_spo_config(sample_rate, pulse_width)

        # Reflectance data (latest update)
        self.buffer_red = []
        self.buffer_ir = []

        self.max_buffer_len = max_buffer_len
        self._interrupt = None

    @property
    def red(self):
        return self.buffer_red[-1] if self.buffer_red else None

    @property
    def ir(self):
        return self.buffer_ir[-1] if self.buffer_ir else None
```

```
    def set_led_current(self, led_current_red=11.0, led_current_ir=11.0):
        # Validate the settings, convert to bit values.
        led_current_red = _get_valid(LED_CURRENT, led_current_red)
        led_current_ir = _get_valid(LED_CURRENT, led_current_ir)
        self.i2c.write_byte_data(I2C_ADDRESS, LED_CONFIG, (led_current_red << 4) |
led_current_ir)

    def set_mode(self, mode):
        reg = self.i2c.read_byte_data(I2C_ADDRESS, MODE_CONFIG)
        self.i2c.write_byte_data(I2C_ADDRESS, MODE_CONFIG, reg & 0x74) # mask
the SHDN bit
        self.i2c.write_byte_data(I2C_ADDRESS, MODE_CONFIG, reg | mode)

    def set_spo_config(self, sample_rate=100, pulse_width=1600):
        reg = self.i2c.read_byte_data(I2C_ADDRESS, SPO2_CONFIG)
        reg = reg & 0xFC  # Set LED pulsewidth to 00
        self.i2c.write_byte_data(I2C_ADDRESS, SPO2_CONFIG, reg | pulse_width)

    def enable_spo2(self):
        self.set_mode(MODE_SPO2)

    def disable_spo2(self):
        self.set_mode(MODE_HR)

    def enable_interrupt(self, interrupt_type):
        self.i2c.write_byte_data(I2C_ADDRESS, INT_ENABLE, (interrupt_type + 1)<<4)
        self.i2c.read_byte_data(I2C_ADDRESS, INT_STATUS)

    def get_number_of_samples(self):
        write_ptr = self.i2c.read_byte_data(I2C_ADDRESS, FIFO_WR_PTR)
        read_ptr = self.i2c.read_byte_data(I2C_ADDRESS, FIFO_RD_PTR)
        return abs(16+write_ptr - read_ptr) % 16
```

```
def read_sensor(self):
    bytes = self.i2c.read_i2c_block_data(I2C_ADDRESS, FIFO_DATA, 4)
    # Add latest values.
    self.buffer_ir.append(bytes[0]<<8 | bytes[1])
    self.buffer_red.append(bytes[2]<<8 | bytes[3])
    # Crop our local FIFO buffer to length.
    self.buffer_red = self.buffer_red[-self.max_buffer_len:]
    self.buffer_ir = self.buffer_ir[-self.max_buffer_len:]

def shutdown(self):
    reg = self.i2c.read_byte_data(I2C_ADDRESS, MODE_CONFIG)
    self.i2c.write_byte_data(I2C_ADDRESS, MODE_CONFIG, reg | 0x80)

def reset(self):
    reg = self.i2c.read_byte_data(I2C_ADDRESS, MODE_CONFIG)
    self.i2c.write_byte_data(I2C_ADDRESS, MODE_CONFIG, reg | 0x40)

def refresh_temperature(self):
    reg = self.i2c.read_byte_data(I2C_ADDRESS, MODE_CONFIG)
    self.i2c.write_byte_data(I2C_ADDRESS, MODE_CONFIG, reg | (1 << 3))

def get_temperature(self):
    intg = _twos_complement(self.i2c.read_byte_data(I2C_ADDRESS, TEMP_INTG))
    frac = self.i2c.read_byte_data(I2C_ADDRESS, TEMP_FRAC)
    return intg + (frac * 0.0625)

def get_rev_id(self):
    return self.i2c.read_byte_data(I2C_ADDRESS, REV_ID)

def get_part_id(self):
    return self.i2c.read_byte_data(I2C_ADDRESS, PART_ID)
```

```
    def get_registers(self):
        return {
            "INT_STATUS": self.i2c.read_byte_data(I2C_ADDRESS, INT_STATUS),
            "INT_ENABLE": self.i2c.read_byte_data(I2C_ADDRESS, INT_ENABLE),
            "FIFO_WR_PTR": self.i2c.read_byte_data(I2C_ADDRESS, FIFO_WR_PTR),
            "OVRFLOW_CTR": self.i2c.read_byte_data(I2C_ADDRESS, OVRFLOW_CTR),
            "FIFO_RD_PTR": self.i2c.read_byte_data(I2C_ADDRESS, FIFO_RD_PTR),
            "FIFO_DATA": self.i2c.read_byte_data(I2C_ADDRESS, FIFO_DATA),
            "MODE_CONFIG": self.i2c.read_byte_data(I2C_ADDRESS, MODE_CONFIG),
            "SPO2_CONFIG": self.i2c.read_byte_data(I2C_ADDRESS, SPO2_CONFIG),
            "LED_CONFIG": self.i2c.read_byte_data(I2C_ADDRESS, LED_CONFIG),
            "TEMP_INTG": self.i2c.read_byte_data(I2C_ADDRESS, TEMP_INTG),
            "TEMP_FRAC": self.i2c.read_byte_data(I2C_ADDRESS, TEMP_FRAC),
            "REV_ID": self.i2c.read_byte_data(I2C_ADDRESS, REV_ID),
            "PART_ID": self.i2c.read_byte_data(I2C_ADDRESS, PART_ID),
        }
```

Code For LCD

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
from sys import argv

import RPi.GPIO as GPIO
import time
import os

###lcd ####################

LCD_RS = 21
LCD_E  = 20
LCD_D4 = 26
LCD_D5 = 19
LCD_D6 = 13
LCD_D7 = 6
```

```
# Define some device constants
LCD_WIDTH = 20    # Maximum characters per line
LCD_CHR = True
LCD_CMD = False

LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
LCD_LINE_3 = 0x90 # LCD RAM address for the 2nd line
LCD_LINE_4 = 0xD0 # LCD RAM address for the 2nd line

# Timing constants
E_PULSE = 0.0005
E_DELAY = 0.0005

####lcd end here###########
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

#GPIO.setup(22, GPIO.IN, pull_up_down=GPIO.PUD_UP)#Button to GPIO12
    # Use BCM GPIO numbers
GPIO.setup(LCD_E, GPIO.OUT)  # E
GPIO.setup(LCD_RS, GPIO.OUT) # RS
GPIO.setup(LCD_D4, GPIO.OUT) # DB4
GPIO.setup(LCD_D5, GPIO.OUT) # DB5
GPIO.setup(LCD_D6, GPIO.OUT) # DB6
GPIO.setup(LCD_D7, GPIO.OUT) # DB7

####################LCD#######################
def lcd_init():
  # Initialise display
  lcd_byte(0x33,LCD_CMD) # 110011 Initialise
  lcd_byte(0x32,LCD_CMD) # 110010 Initialise
```

```
  lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
  lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
  lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
  lcd_byte(0x01,LCD_CMD) # 000001 Clear display
  time.sleep(E_DELAY)

def lcd_byte(bits, mode):
  # Send byte to data pins
  # bits = data
  # mode = True  for character
  #        False for command

  GPIO.output(LCD_RS, mode) # RS

  # High bits
  GPIO.output(LCD_D4, False)
  GPIO.output(LCD_D5, False)
  GPIO.output(LCD_D6, False)
  GPIO.output(LCD_D7, False)
  if bits&0x10==0x10:
    GPIO.output(LCD_D4, True)
  if bits&0x20==0x20:
    GPIO.output(LCD_D5, True)
  if bits&0x40==0x40:
    GPIO.output(LCD_D6, True)
  if bits&0x80==0x80:
    GPIO.output(LCD_D7, True)

  # Toggle 'Enable' pin
  lcd_toggle_enable()

  # Low bits
  GPIO.output(LCD_D4, False)
```

```
  GPIO.output(LCD_D5, False)
  GPIO.output(LCD_D6, False)
  GPIO.output(LCD_D7, False)
  if bits&0x01==0x01:
    GPIO.output(LCD_D4, True)
  if bits&0x02==0x02:
    GPIO.output(LCD_D5, True)
  if bits&0x04==0x04:
    GPIO.output(LCD_D6, True)
  if bits&0x08==0x08:
    GPIO.output(LCD_D7, True)

  # Toggle 'Enable' pin
  lcd_toggle_enable()

def lcd_toggle_enable():
  # Toggle enable
  time.sleep(E_DELAY)
  GPIO.output(LCD_E, True)
  time.sleep(E_PULSE)
  GPIO.output(LCD_E, False)
  time.sleep(E_DELAY)

def lcd_string(message,line):
  # Send string to display
  message = message.ljust(LCD_WIDTH," ")

  lcd_byte(line, LCD_CMD)

  for i in range(LCD_WIDTH):
    lcd_byte(ord(message[i]),LCD_CHR)

#####################LCD HERE######################
```

```
##extract results
lcd_init()
while True:
   try :
      lcd_byte(0x01, LCD_CMD)
      lcd_string(" SMART BED  ",LCD_LINE_1)
      lcd_string("  SYSTEM    ",LCD_LINE_2)
      lcd_string("        ",LCD_LINE_3)
      time.sleep(4)
   except KeyboardInterrupt:
       gpio.cleanup()
       raise
```

Hard copy of the presented conference paper / published journal paper

JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Development Of Health Monitoring System With Automated Bed Control Using Hand Gesture & Voice Recognition

[1]Manasa R, [2]Dhananjay Natani, [3]Ekta Sinha, [4]Krati Sharma, [5]Narayan Kumar
[1]Professor, Dept of ECE, [2]Student, Dept of ECE, [3]Student, Dept of ECE, [4]Student, Dept of ECE, [5]Student, Dept of ECE
[1]Department of Electronics and Communication Engineering,
[1]Dayananda Sagar College of Engineering, Bengaluru, India

Abstract : The project focuses on the development of a gesture-controlled and voice-controlled bed system for improved patient care and comfort. The system integrates various sensors such as temperature, humidity, air quality, heart rate, SpO2, and blood pressure, allowing for continuous monitoring of vital signs. The bed movements are controlled through hand gestures and voice commands, providing patients with greater independence and convenience. Additionally, the system alerts healthcare professionals via Telegram in case of abnormal temperature, poor air quality, low heart rate, or high blood pressure. The data collected from the sensors is also transmitted to ThingSpeak for further analysis and tracking. The project aims to enhance the healthcare experience for patients and support caregivers in delivering personalized care.

Keywords- Gesture controlled, voice controlled, patient care, continuous monitoring, healthcare technology

I. INTRODUCTION

The project revolves around addressing the challenges faced by individuals with disabilities, the elderly, and patients requiring continuous care. In contemporary society, the availability of full-time caregivers is limited, and there is a growing need for innovative solutions that can enhance the quality of life for these individuals. To meet this need, we have developed a groundbreaking gesture-controlled and voice-controlled bed system.

The primary objective of our project is to provide a more independent and comfortable experience for users through the integration of hand gesture and voice recognition technology in controlling the bed's position. By eliminating the need for constant assistance, individuals can have greater control over their bed movements, allowing them to find their desired position and achieve optimal comfort. This technology empowers users to adjust the bed's position effortlessly by simply using hand gestures or issuing voice commands, enabling them to customize their comfort levels according to their preferences and physical needs.

In addition to the innovative control mechanisms, the bed system incorporates various sensors to continuously monitor the user's health parameters. These sensors include heartbeat sensors, temperature sensors, SpO2 and blood pressure monitors, humidity sensors, and air quality sensors. The real-time data collected by these sensors is processed and analyzed to provide insights into the user's health status. By continuously monitoring vital signs and environmental conditions, the bed system ensures proactive care and facilitates early intervention in case of any health-related concerns. The smart bed system is designed with safety features to enhance user well-being. It includes an emergency button that users can easily access to request immediate assistance. The system also employs an alarm mechanism that triggers if any vital health parameter exceeds predefined thresholds, alerting both the user and the healthcare staff. These safety features provide peace of mind to users and their caregivers, knowing that help can be summoned promptly in case of any emergency.

In summary, our project focuses on improving the quality of life for individuals with disabilities, the elderly, and those in need of continuous care. The integration of gesture and voice control, along with comprehensive health monitoring and safety features, sets our bed system apart, offering enhanced independence, comfort, and peace of mind for users and their caregivers. Through our innovative solution, we aim to transform the way care is provided, enabling individuals to have greater autonomy and a more fulfilling life experience.

II. Related Work

In the field of alerting systems for bedridden patients and gesture-controlled bed movements, several related works have been conducted, aiming to enhance the caregiving experience and improve patient outcomes.One study by E Spandana, M Rajasekar, and N Sandhya (2021) focuses on an alerting system based on hand gesture recognition. The system utilizes a camera module to capture hand images, and through background subtraction, image recognition, and speech conversion, it notifies healthcare providers when patients require assistance. This real-time approach enhances operational efficiency, responsiveness, and patient satisfaction.Another study by Ignacio Ghersi, Mario Mariño, and Mónica Teresita Miralles (2018) explores the concept of smart medical beds in patient-care environments. They discuss the functional, aesthetic, and interactive features of these beds and assess the global market and related standards. The study also addresses emerging challenges and opportunities for adding monitoring and assistive implementations to medical devices.Ms. T. Umarani, Ms. I. Preethy, Ms. N.M. Thennarasi, and Ms. J.E. Jeyanthi (2019) propose a gesture-controlled bed movement system using webcams and simple hardware components. Their real-time system allows for changing the bed's position through hand gestures, facilitating human-computer interaction.In a study by Ms. Kalpana Lamb and Prof. Mrs. Swati Madhe (2016), an automatic bed position control system based on hand gesture recognition is presented. The system uses the background subtraction algorithm and Symlet wavelet transform for accurate hand gesture recognition. The detected gestures are processed by a micro-controller, which controls the DC motor to change the bed's position accordingly.These related works have contributed to the advancement of alerting systems and gesture-controlled bed movements, addressing various aspects such as hand gesture recognition, real-time processing, and accurate positioning. However, the proposed project aims to overcome certain limitations by incorporating additional features and algorithms, thereby enhancing the functionality and monitoring capabilities of the smart bed system.

III. Proposed System

A. BLOCK DIAGRAM OF PROPOSED SYSTEM

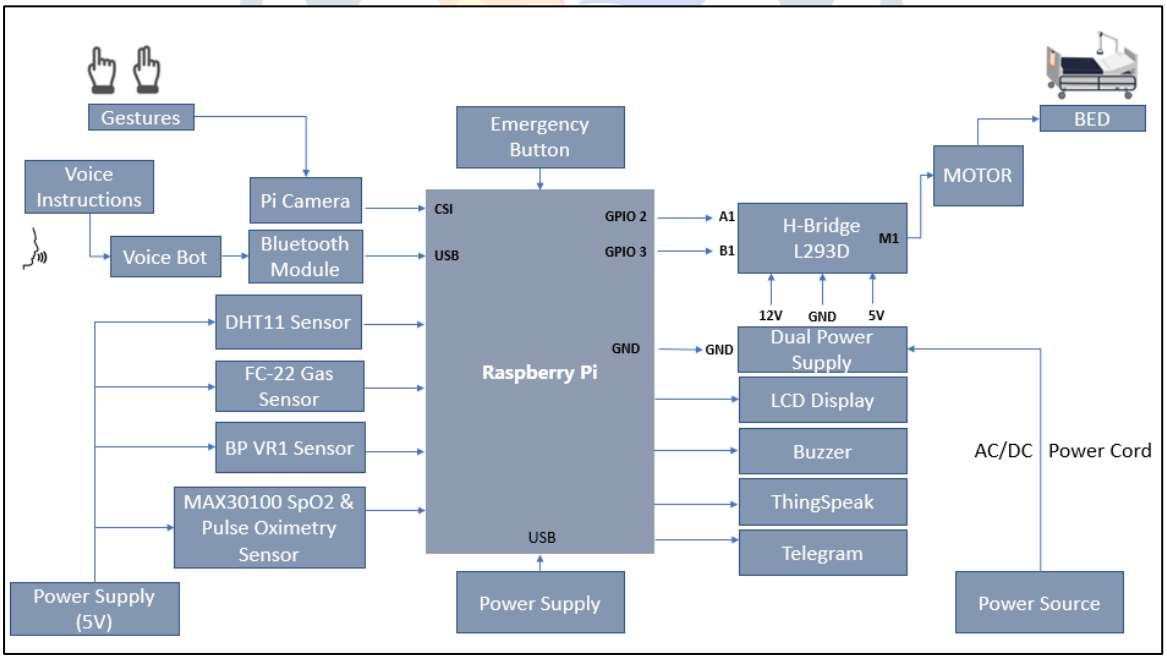


Fig. 1: Block-diagram of the proposed methodology

The proposed project incorporates multiple blocks to enable its functionality, as depicted in Figure 1. The first block is the Gesture Recognition Block, which employs advanced gesture recognition technology to interpret hand gestures made by the user. This block utilizes sensors and sophisticated algorithms to accurately detect and interpret the gestures, allowing for intuitive control of the bed's movements. By simply performing specific hand gestures, users can initiate actions such as adjusting the bed's position or activating predefined functions.The Voice Recognition Block is another essential component of the system. It employs powerful speech recognition algorithms and audio processing techniques to recognize and understand voice commands given by the user. This block enables hands-free control of

the bed, providing convenience and ease of use.The DC Motor Control Block plays a vital role in translating the user's commands into physical movements of the bed. It consists of a DC motor and a dedicated motor driver that receives signals from the Gesture Recognition or Voice Recognition Block and precisely controls the motor's speed and direction. This allows for smooth and accurate adjustments of the bed's position based on the user's inputs.The Sensor Network Block integrates various sensors, including heartbeat sensors, temperature sensors, SPo2 and blood pressure monitors, humidity sensors, and air quality sensors. These sensors continuously monitor the patient's vital signs and the surrounding environmental conditions. The collected data provides valuable insights into the patient's well-being and helps ensure a safe and comfortable environment.The Raspberry Board Block serves as the central processing unit of the system. It receives and processes the analog signals from the sensors, performing signal conditioning, data conversion, and analysis. This block enables real-time monitoring and analysis of the patient's health parameters, facilitating timely interventions if necessary.Overall, the proposed system utilizes a combination of gesture recognition, voice recognition, motor control, and sensor integration to provide intuitive and responsive control of the bed while ensuring continuous monitoring of vital parameters and environmental conditions.

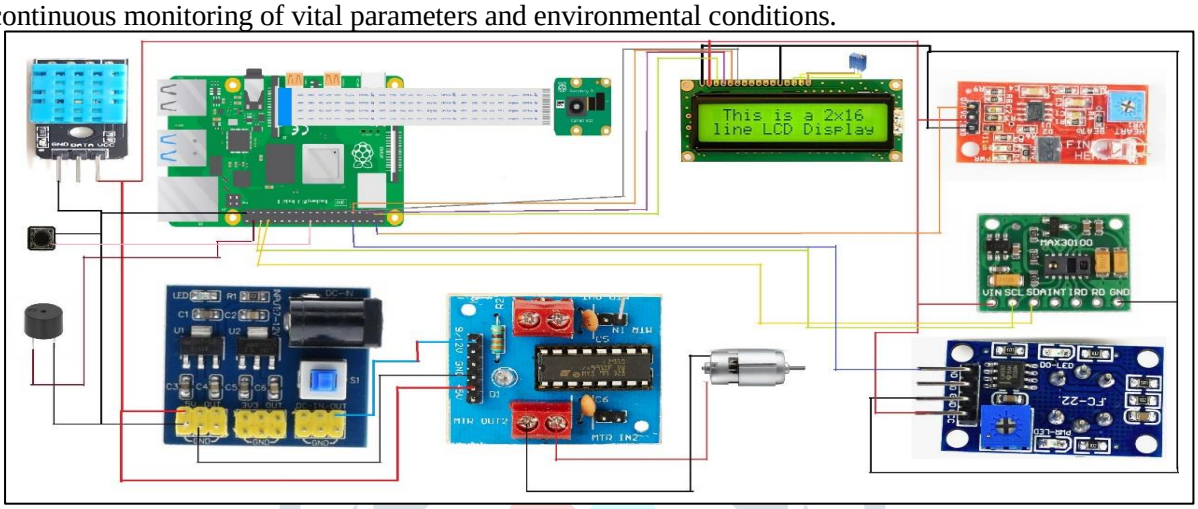


Fig. 2 : Circuit diagram of the proposed system

Figure 2 represents the circuit diagram of the proposed system, showcasing the interconnections and components involved in the project. It provides a visual representation of the electrical connections and the arrangement of various elements. The diagram includes the Raspberry Pi board, sensors such as temperature, humidity, and gas sensors, the motor driver, power supply, and other essential components. Each component's placement and connectivity are illustrated, helping to understand the overall circuitry of the system. The circuit diagram serves as a guide for assembling and wiring the components correctly, ensuring proper functionality and coordination among different elements of the project.

B. HARDWARE COMPONENTS

1. **Raspberry Pi 3 B:** The Raspberry Pi serves as the main controller unit of the system, responsible for processing data, controlling the bed movements, and integrating various components.
2. **Bluetooth BC417 Module:** This module enables wireless communication between the Raspberry Pi and external devices, facilitating the integration of voice recognition and gesture control capabilities.
3. **H-Bridge L293D Motor Driver**: The motor driver is used to control the DC motor responsible for adjusting the bed's position. It provides the necessary power and direction control for smooth bed movement.
4. **Raspberry Pi Camera 5MP:** The camera module is utilized for capturing images or video for monitoring purposes, such as detecting hand gestures or recording the environment.
5. **USB UART Serial Converter port:** This component enables serial communication between the Raspberry Pi and other devices, facilitating data transfer and control.
6. **Dual power supply**: It is used to provide power to the H-Bridge motor driver in addition to powering the DC motor. The 5V power supply handles the control signals and logic components, while the 12V power supply drives the motor through the H-Bridge. This setup ensures efficient and reliable operation of the motor and allows for precise control of bed movements in the gesture-controlled system.
7. **DC Motor:** A DC motor is utilized to control the movement of the bed in the project.
8. **DHT11 Sensor:** The DHT11 sensor is used in the project to measure temperature and humidity levels in the environment where the patient is lying on the bed. It provides real-time data on these parameters, allowing healthcare professionals to monitor the comfort and well-being of the patient. The sensor's compact size, low power consumption, and reliable performance make it suitable for integration into the bed system for accurate environmental monitoring.

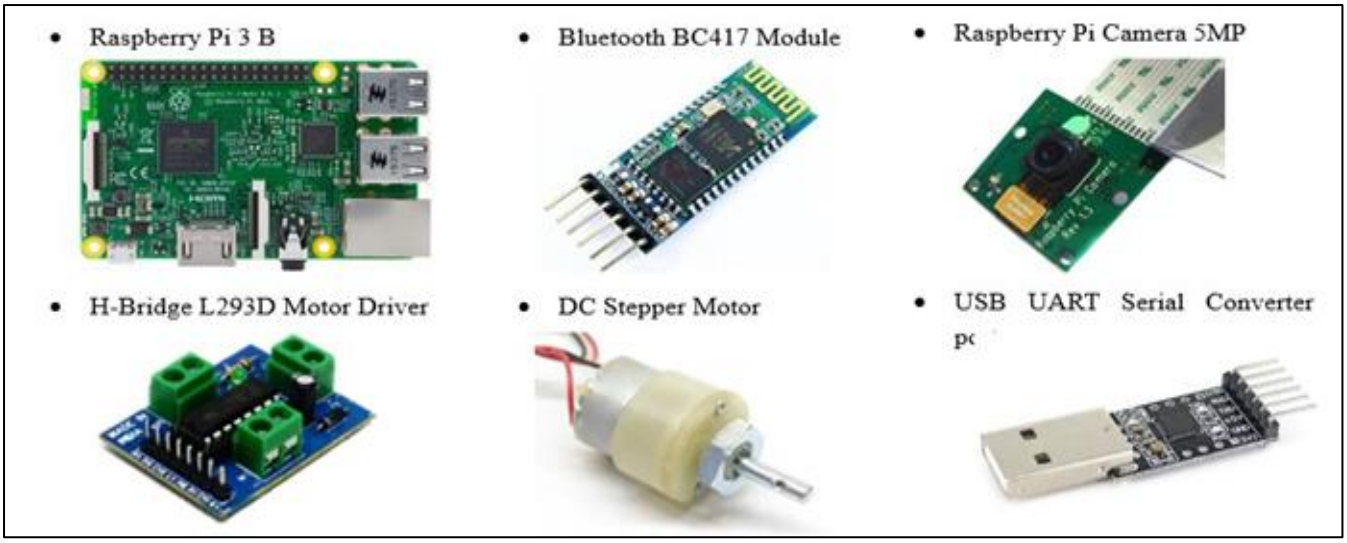


Fig.3 Hardware Components

9. **FC-22 Gas Sensor:** The FC-22 gas sensor is utilized in the project to monitor air quality and detect harmful gases in the vicinity of the patient. It provides reliable measurements of various gases, allowing for early detection of potential health hazards. By integrating the FC-22 sensor into the bed system, healthcare professionals can ensure a safe and healthy environment for the patient, while enabling timely interventions in case of any gas-related risks or issues.
10. **BP VR1 Sensor:** The BP level sensor is utilized in the project to measure and monitor the blood pressure of the patient. It provides valuable information on the patient's blood pressure levels, enabling healthcare professionals to assess cardiovascular health and detect any abnormalities or fluctuations. By incorporating the BP level sensor into the bed system, continuous monitoring of blood pressure is achieved, allowing for timely intervention and management of hypertension or other blood pressure-related conditions. This helps in providing personalized care and ensuring the well-being of the patient.
11. **LCD:** The LCD 16*2 display is utilized in the project to provide a user-friendly interface for displaying the values of various sensors used in the system. It allows for clear and concise visualization of important information such as heart rate, blood pressure, temperature, humidity, and air quality readings. The LCD display enhances the usability of the system, enabling healthcare professionals and caregivers to easily monitor and interpret the sensor data, promoting effective decision-making and ensuring optimal patient care.

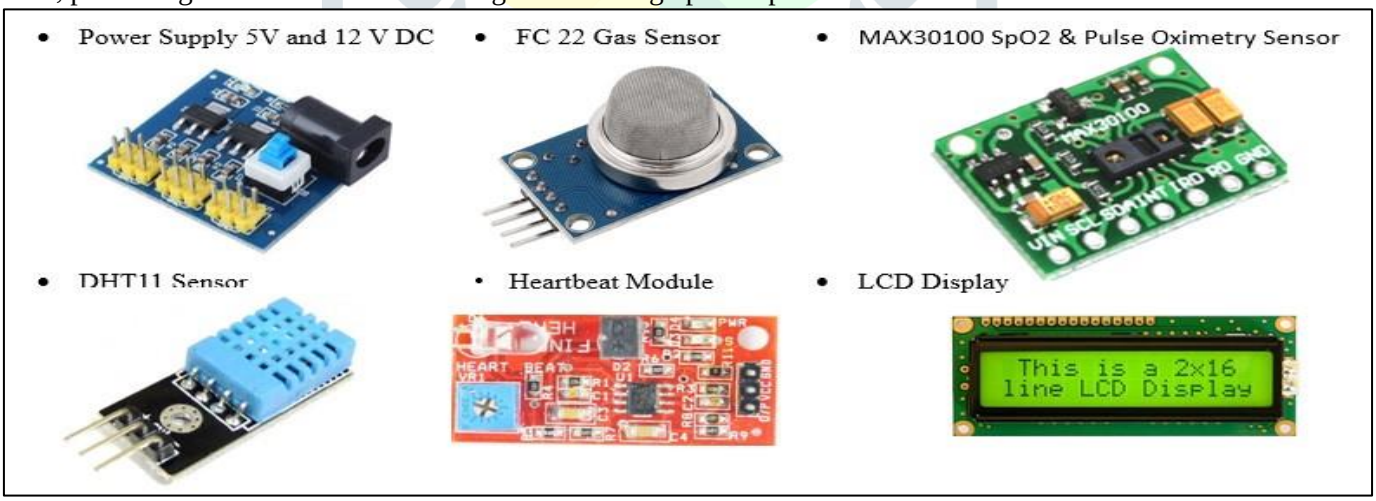


Fig. 4 Sensors used in proposed system

12. **MAX30100 SpO2 & Pulse Oximetry Sensor:** The MAX30100 sensor is employed in the project to measure and monitor various vital parameters such as heart rate, blood oxygen saturation (SpO2), and pulse oximetry. It utilizes advanced optical sensing technology to accurately capture and analyze the patient's physiological data. By integrating the MAX30100 sensor into the bed system, continuous and non-invasive monitoring of vital signs is achieved, enabling healthcare professionals to assess the patient's overall health status and detect any abnormalities. This information facilitates timely interventions and personalized care, improving patient outcomes and ensuring their well-being.

    All these sensors like heart rate sensor, SpO2 and BP sensor, temperature sensor, humidity sensor, and air quality sensor are used to collect real-time data on the patient's vital signs and environmental conditions. These sensors provide essential inputs for continuous monitoring and analysis.

C. SOFTWARE USED

The software tools used in the project work include **Raspberry Pi OS (Raspbian)**, **Python 3**, **TensorFlow, Mediapipe, Numpy,** and **ThingSpeak**.

1. **Raspberry Pi OS** is the operating system used on the Raspberry Pi board, providing a platform for running the project software.
2. **Python 3** is the programming language used to develop the code for the gesture-controlled bed system and integrate various functionalities.
3. **TensorFlow** is a machine learning framework used for developing and training the hand gesture recognition model. It enables the system to accurately interpret and recognize different hand gestures for controlling the bed movements.
4. **Mediapipe** is a library that provides ready-to-use solutions for tasks such as hand tracking, pose estimation, and facial recognition. It is used to track and analyze hand movements in real-time.
5. **Numpy** is a numerical computing library in Python, used for efficient data handling and manipulation in the project. It helps in processing and analyzing the sensor data obtained from various sensors integrated into the bed system.
6. **ThingSpeak** is an IoT platform used for collecting and visualizing the sensor data. It enables remote monitoring of the patient's vital parameters and provides insights into their health condition.

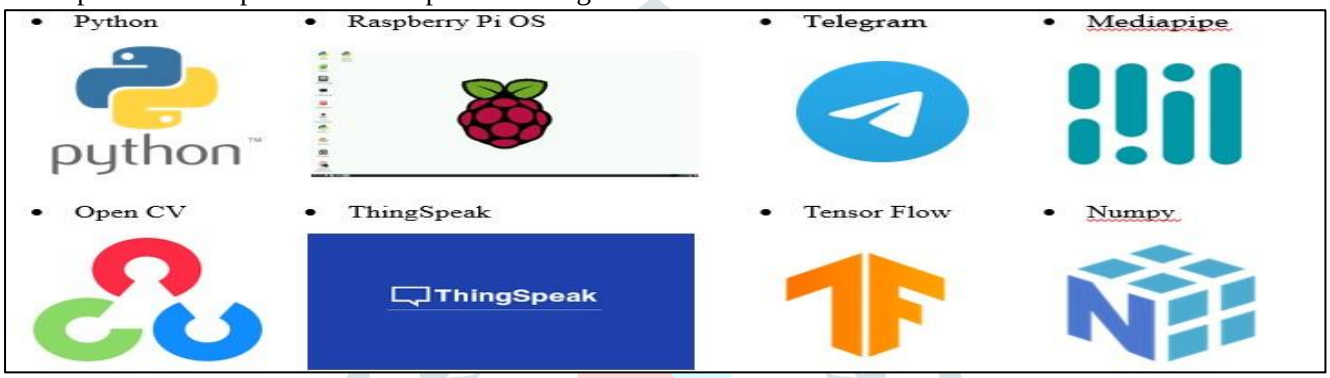


Fig. 5 Software used

These software tools in Figure 5 were utilized to develop an integrated system that combines gesture recognition, sensor data processing, and IoT capabilities, enabling precise control and monitoring of the bed movements and patient's health parameters.

D. FLOWCHART

1. Flow Diagram of Hand Gesture controlled bed:

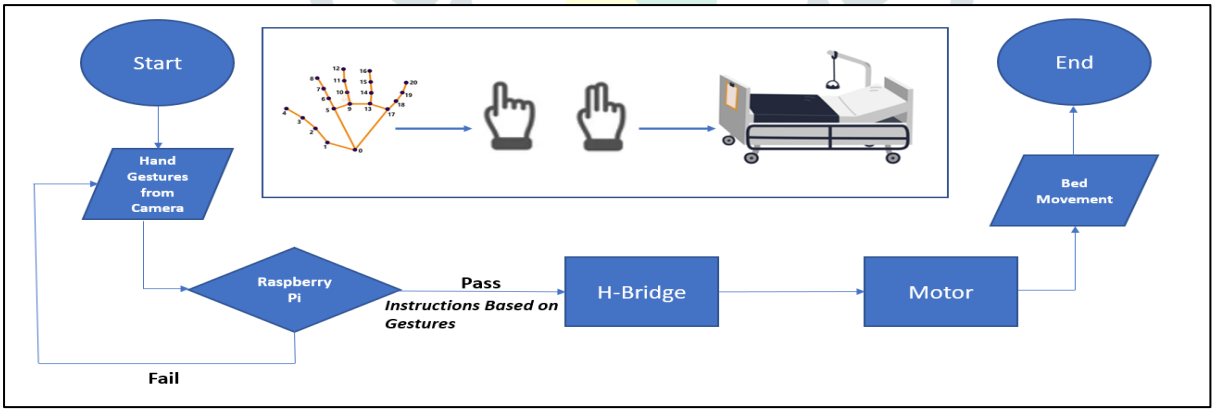


Fig. 6: Flow diagram of the proposed methodology for Hand Gesture

The plans and techniques used in the design and development of the Hand gesture recognition in Figure 6 is for change the position of automatic adjustable beds in hospitals are described in this section. In order to perform hand gesture recognition to control bed position, many tools, libraries and languages are used like Open-Source Computer Vision Library. Open CV established in Python Language. Python 3.7.4 version is an open-source environment for compile the source code. It has many inbuilt functions for recognizing hand gesture. The main component used here is the Raspberry Pi. It is used to get the information from the Patient, based on the command it can take the necessary action and then change the bed movement.

2. Flow Diagram of Voice Controlled bed:

The flowchart in Figure 7 for voice recognition illustrates the steps involved in the process of interpreting and executing voice commands. Firstly, the system starts by listening for audio input through a microphone. The audio signal is then preprocessed, including noise reduction and signal conditioning, to enhance the quality. Next, the

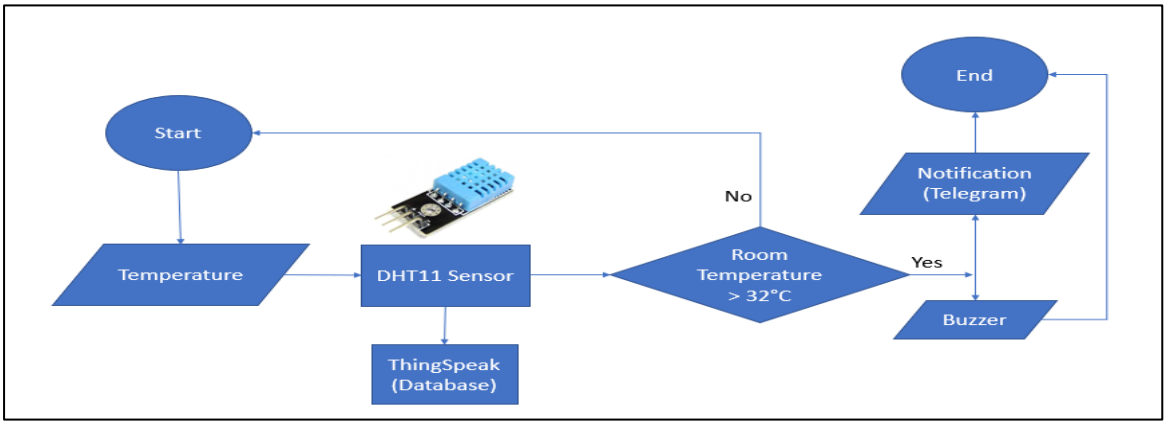


Fig. 7 Flow diagram of voice recognition model

preprocessed audio is analyzed using voice recognition algorithms, which convert the speech into text. The system then matches the recognized text with predefined commands or patterns stored in its database. If a match is found, the corresponding action associated with the command is executed. Finally, the system provides feedback or output based on the executed action, such as displaying information or performing a specific task. The flowchart guides the implementation of voice recognition, ensuring a systematic approach to capture, process, and interpret voice commands accurately for controlling the bed's movements.

3. Flow Diagram of Temperature Sensor:

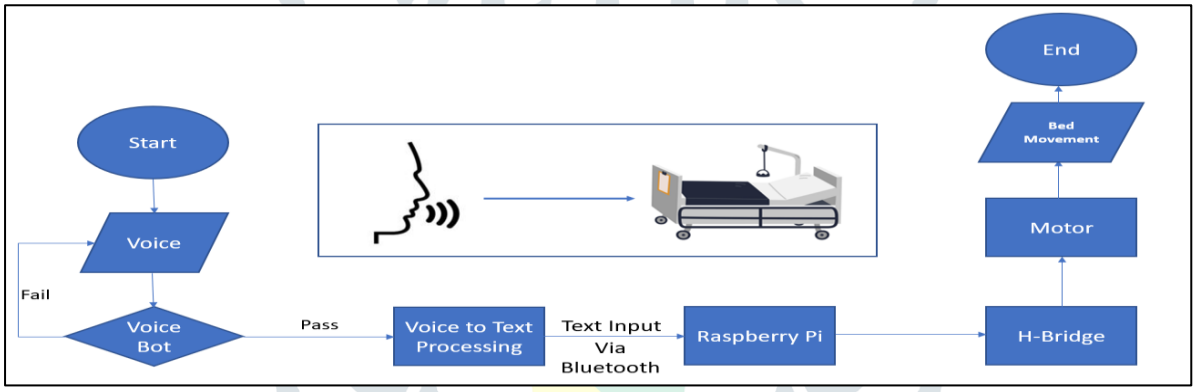


Fig. 8 Flow diagram of temperature sensor model

The flow diagram in Figure 8 for the temperature sensor in begins with the start of the temperature monitoring process. The temperature value is then read from the sensor, followed by a check to determine if it exceeds 32 degrees Celsius. If the temperature surpasses this threshold, a notification is sent on Telegram to alert the user. Concurrently, the temperature data is continuously transmitted to ThingSpeak, providing ongoing monitoring and data storage. The process repeats at regular intervals to ensure real-time tracking of temperature changes. Overall, this flow diagram showcases the systematic sequence of actions involved in monitoring the temperature, detecting threshold breaches, and facilitating timely notifications and data collection for further analysis.

4. Flow Diagram of FC-22 Gas Sensor:

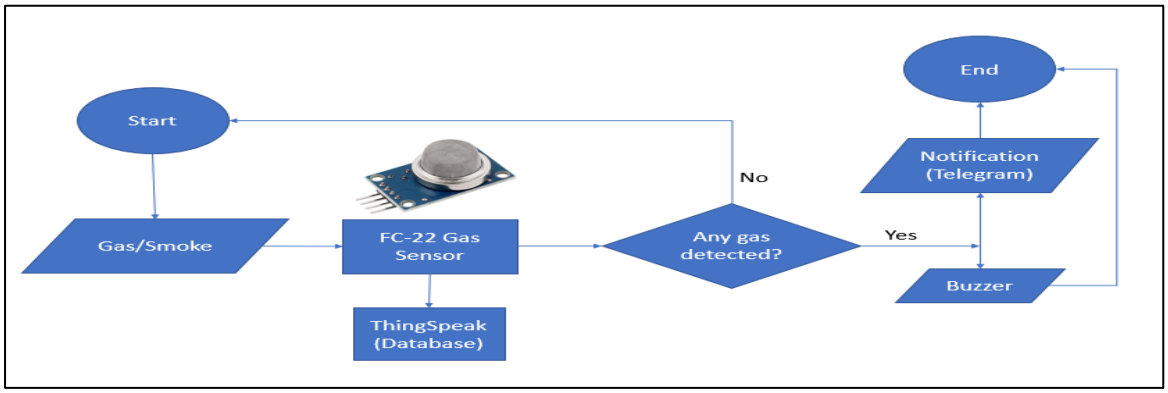


Fig. 9 Flow diagram of gas sensor model

The flow diagram for the gas sensor in Figure 9 starts with the initiation of the gas quality monitoring process. The gas sensor measures the air quality and checks for the presence of harmful gases. If the air quality is determined to be

poor or if harmful gases are detected, a notification is sent on Telegram to alert the user. Simultaneously, the gas sensor continuously sends data to ThingSpeak, enabling continuous monitoring and data logging. The process repeats at regular intervals to ensure real-time tracking of air quality changes and prompt notifications in case of any concerns. This flow diagram illustrates the step-by-step process of monitoring gas levels, detecting abnormalities, and providing timely notifications and data collection for further analysis.

5. Flow Diagram of Max30102 SpO2 Sensor:

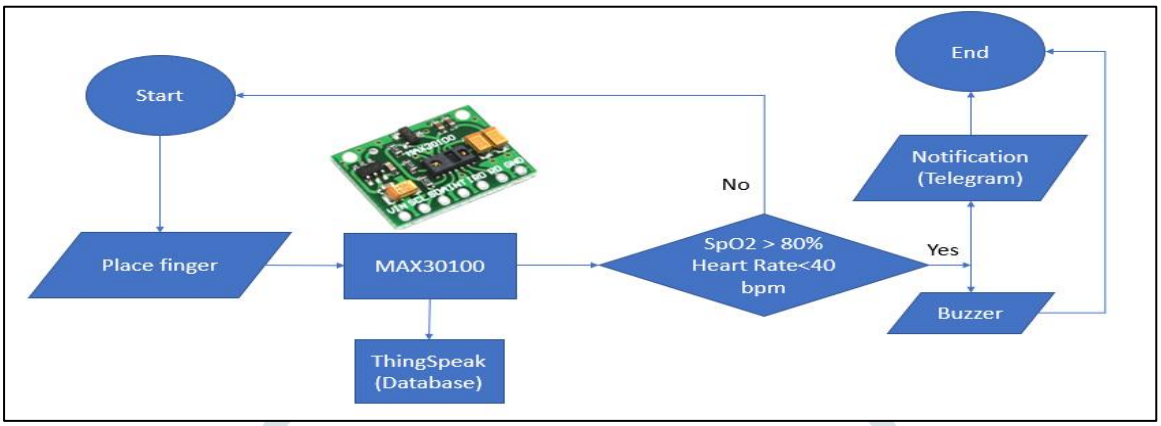


Fig. 8 Flow diagram of max30100 sensor model

The flow diagram in Figure 8 is for the MAX30100 SpO2 and Pulse Oximetry sensor starts with the initialization of the sensor for SpO2 and pulse rate measurement. The sensor continuously reads the SpO2 and pulse rate values from the user's finger. If the SpO2 level drops below 80% or if the blood pressure exceeds 170mmHg, a notification is sent on Telegram to alert the user about the abnormal readings. Additionally, the MAX30100 sensor sends continuous data to ThingSpeak for real-time monitoring and analysis. This allows for ongoing tracking of SpO2 levels and pulse rates, enabling early detection of any potential health issues. The flow diagram demonstrates the sequential steps involved in measuring SpO2 and pulse rate, detecting abnormal readings, notifying the user, and capturing data for further analysis and evaluation.

6. Flow Diagram of Heart Rate Sensor:

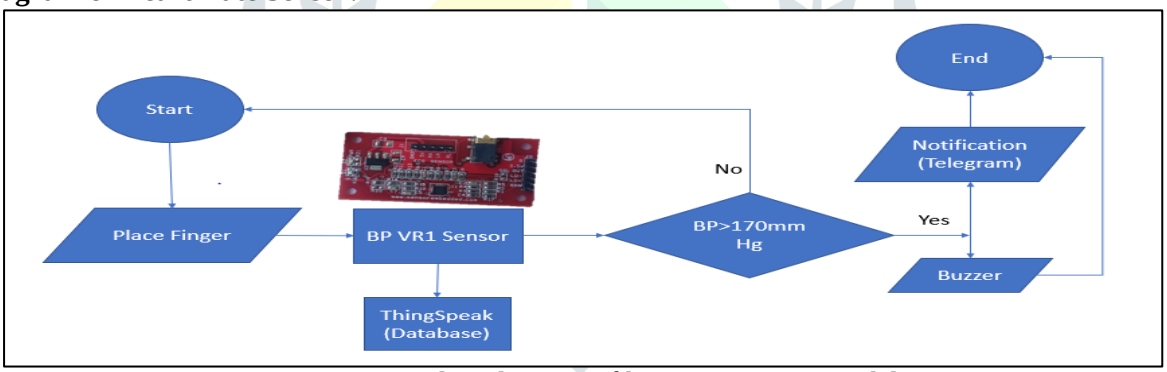


Fig. 9 Flow diagram of heart rate sensor model

The flow diagram in Figure 9 is for the heart rate sensor begins with the initiation of the heart rate monitoring process. The sensor measures the heart rate and continuously monitors it. If the heart rate drops below 40 beats per minute, a notification is sent on Telegram to notify the user about the low heart rate. Additionally, the heart rate sensor sends continuous data to ThingSpeak for real-time tracking and analysis. This allows for the ongoing monitoring of heart rate trends and the detection of any abnormalities. The flow diagram represents the sequential steps involved in monitoring the heart rate, detecting low heart rates, and providing notifications, as well as capturing and logging data for further evaluation and assessment.

IV. Experimental results

Fig. 10 Voice command is given to system and bed movement is shown

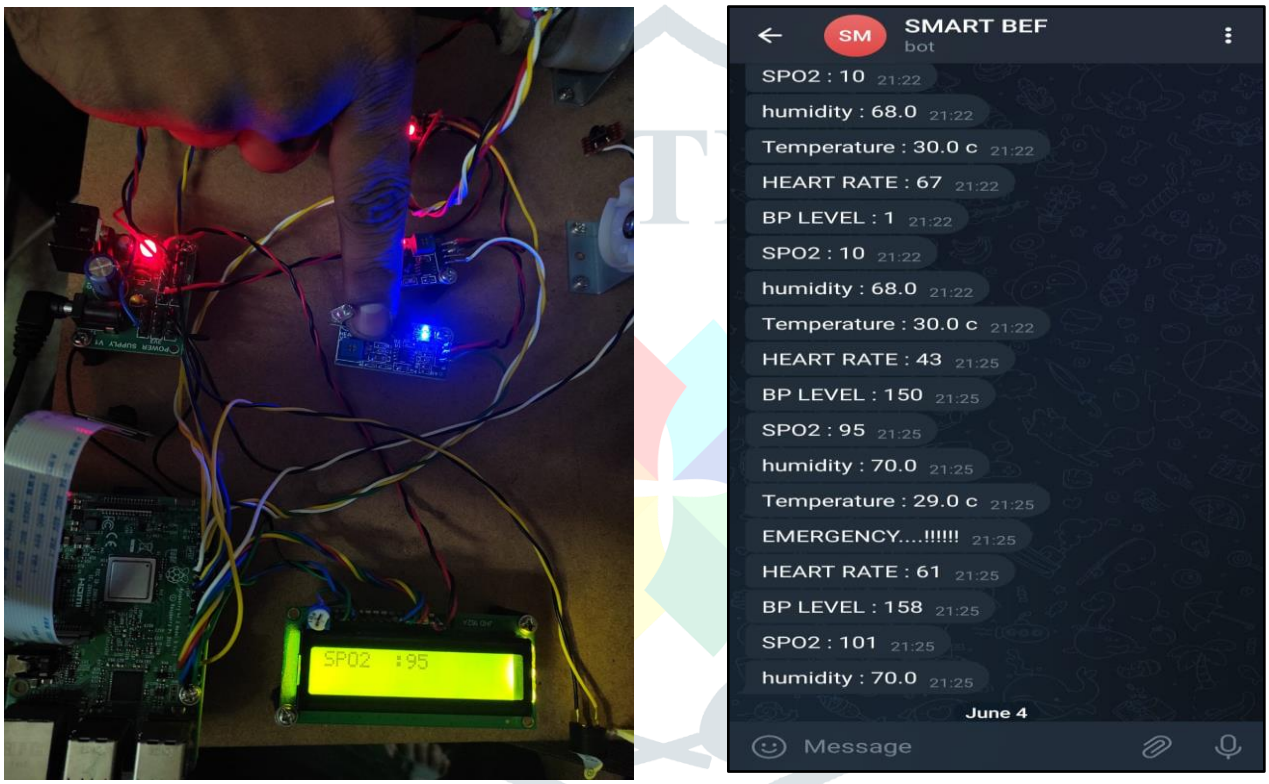


Fig. 11 Sensors is activated my placing finger on heart rate sensor & all parameters is shown on lcd and on telegram

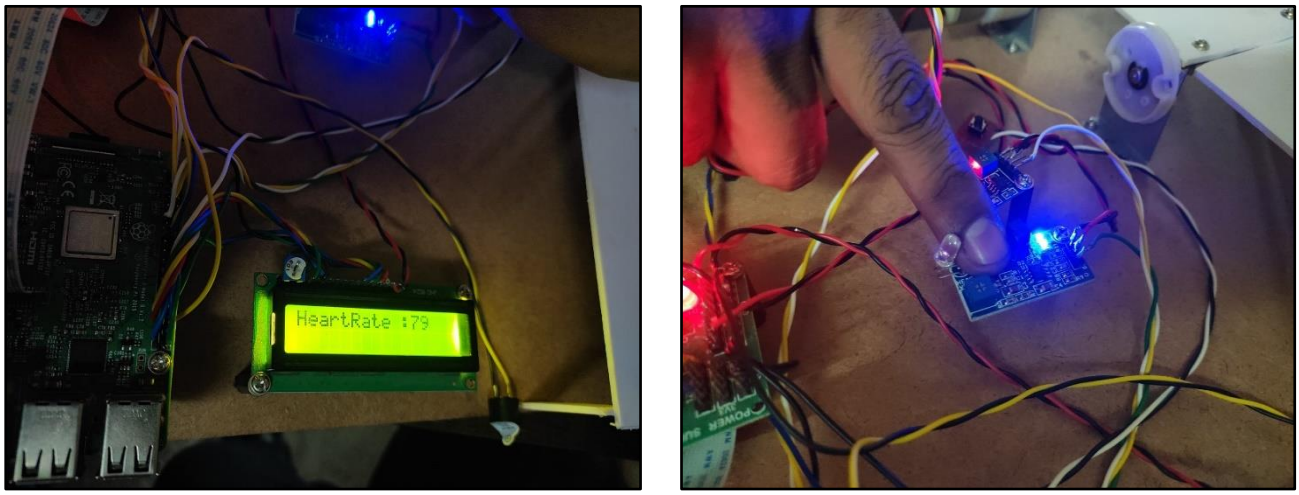


Fig. 12 placing finger on heart rate sensor & is shown on lcd

Fig. 13 Hand Gestures

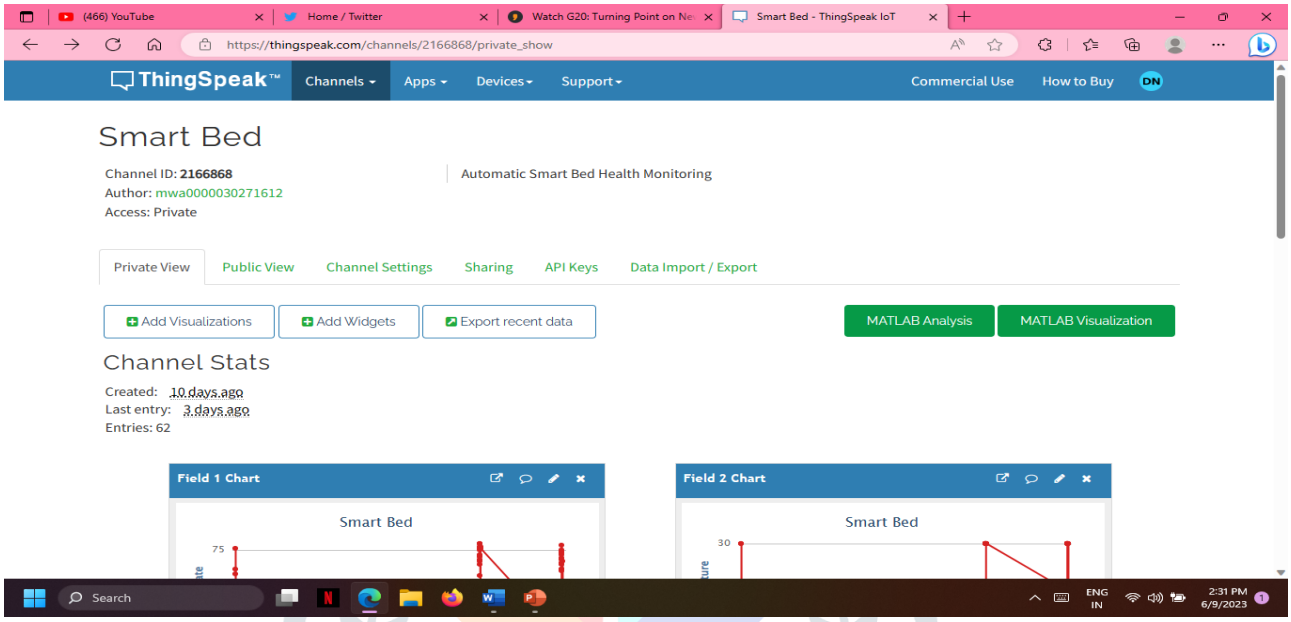
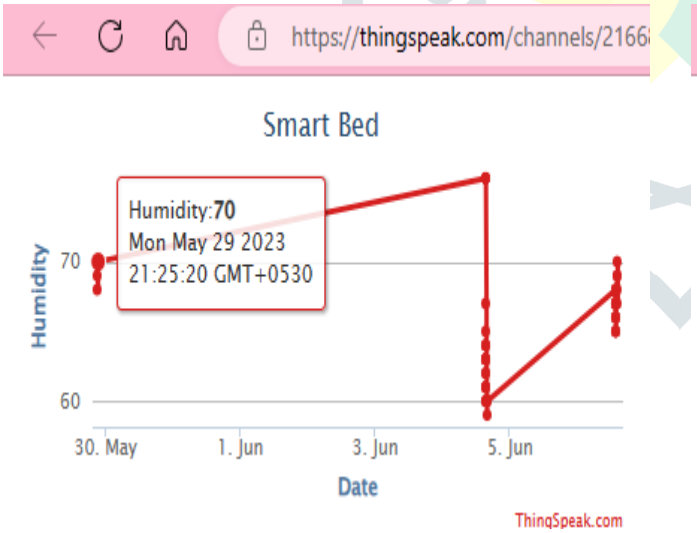
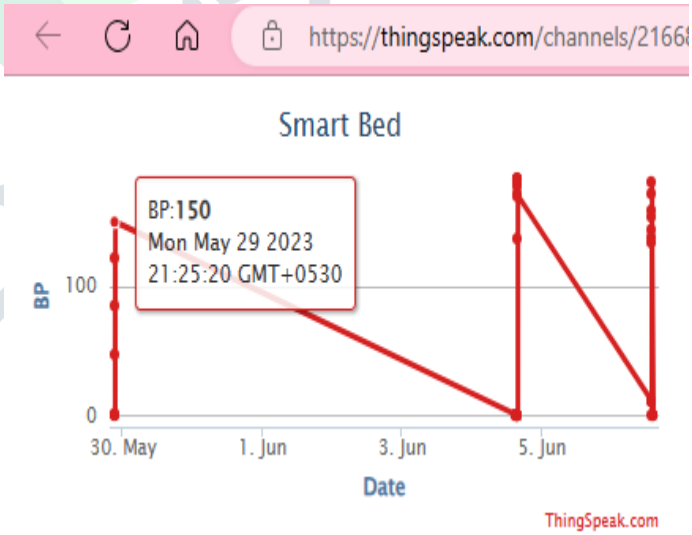
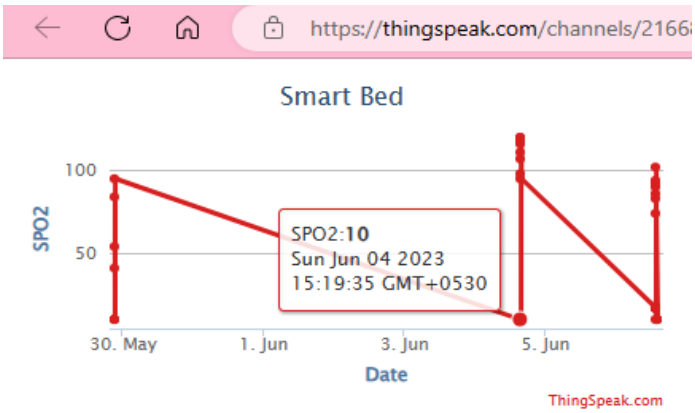
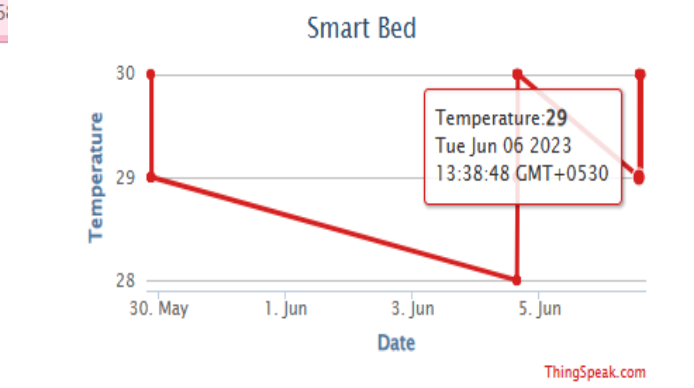


Fig 14 ThingSpeak SS









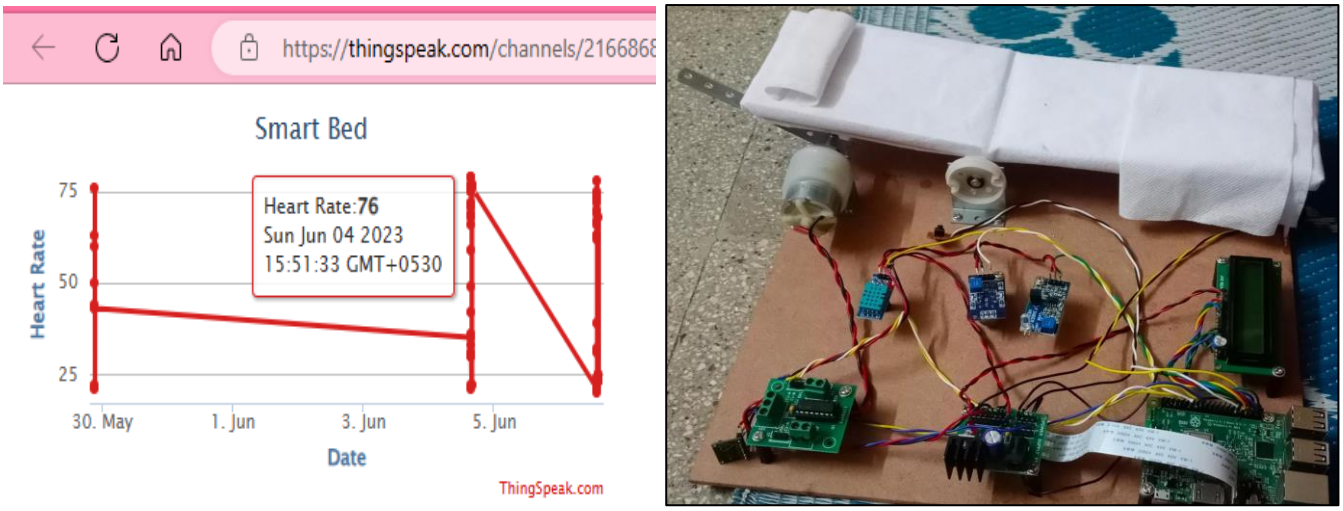


Fig. 15 Thingspeak graph snapshot & Hardware Prototype of proposed system

Fig. 16 Heart Rate sensor & Bed Movement using Hand Gestures for up and down movement

V. Conclusion & Future Work

A. Conclusion

In conclusion, the proposed project aims to develop a bed control system that enables gesture recognition and voice recognition for enhanced user convenience and patient care. The integration of various blocks, including gesture recognition, voice recognition, DC motor control, sensor network, and Raspberry board, allows for intuitive and efficient control of the bed's movements. Additionally, the continuous monitoring of vital parameters and environmental conditions adds an extra layer of safety and care for the patients. The system provides a user-friendly interface for controlling the bed through hand gestures and voice commands, reducing the need for physical buttons or manual adjustments. The implementation of advanced algorithms and the utilization of sensors ensure accurate and reliable performance. The project contributes to the field of healthcare technology by providing an innovative solution for bed control and patient monitoring. Future work can focus on expanding the capabilities of the system, integrating additional sensors and functionalities, and conducting extensive testing and validation to ensure its effectiveness and usability in real-world healthcare settings. Overall, the project holds great potential to enhance patient comfort, caregiver efficiency, and the overall quality of care in medical facilities.

B. Future Work

In terms of future work, there are several areas that can be explored to further enhance the proposed bed control system. Firstly, integrating machine learning algorithms can improve the accuracy and robustness of gesture recognition and voice recognition. This would enable the system to adapt to individual users' gestures and speech patterns, improving overall user experience. Secondly, expanding the sensor network to include additional health monitoring sensors, such as ECG and respiration rate sensors, can provide a more comprehensive view of the patient's health status. Thirdly, incorporating wireless communication capabilities, such as Bluetooth or Wi-Fi, can enable seamless connectivity with other devices or healthcare systems for data sharing and remote monitoring.

Lastly, conducting user studies and gathering feedback from healthcare professionals and patients would help to refine the system's design and usability, ensuring it meets the specific needs of different healthcare environments. By addressing these areas, the future work can further optimize the functionality, performance, and usability of the bed control system, ultimately enhancing the overall patient care experience.

REFERENCES

[1] Sajida Fayyaz, Sidra Tahir. "Adjustment of Bed for a Patient through Gesture Recognition: An Image Processing Approach". IEEE 2018.

[2] Hui-Shyong Yeo & Byung-Gook Lee & Hyotaek Lim. "Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware" Springer Science+Business Media New York 2013.

[3] Diego A. Socolinsky, Andrea Selinger, and Joshua D. Neuheisel. "Face recognition with visible and thermal infrared imagery". Computer Vision and Image Understanding 91 (24 march 2003) 72–114.

[4] Elli Angelopoulou, Rana Molana, and Kostas Daniilidis. "Multispectral Skin Color Modeling". Computer Information Science Department, University of Pennsylvania Philadelphia, *PA.*

[5] Mokhtar M.Hasan, Pramod K. Mishra. "HSV Brightness Factor Matching for Gesture Recognition System". International Journal of Image Processing (IJIP), Volume (4): Issue (5) .Department of Computer Science Faculty of Science BHU-Varanasi
Uttar Pradesh-221005, India.

[6] Gaglio. S, G.L. Re, and M. Morana. "Human activity recognition process using 3-D posture data". IEEE Transactions on Human-Machine Systems. vol. 45, no.5, pp. 586-597, 2015.

Plagiarism reports

Development of Health Monitoring System with Automated Bed Control using Hand Gesture & Voice Recognition

ORIGINALITY REPORT

15%	7%	12%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	M. Neela Harish, K. Praveen Kumar, N. Athaf Khanam, P. R. Akshaya, A. Karshima, Kiran Bihari, V. Aparna, A. Bhuvaneshwari. "Flex sensor controlled assist device for COVID-19 patients", AIP Publishing, 2022 Publication	3%
2	Submitted to BMS College of Engineering Student Paper	2%
3	Kalpana Lamb, Swati Madhe. "Automatic bed position control based on hand gesture recognition for disabled patients", 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), 2016 Publication	2%
4	E Spandana, M Rajasekar, N Sandhya. "Care-giver alerting for bedridden patients using hand gesture recognition system", Journal of Physics: Conference Series, 2021 Publication	1%

5	iopscience.iop.org Internet Source	1 %
6	Sajida Fayyaz, Rubab Bukhsh, Muhammad Adnan Khan, Hafiz Ali Hamza Gondal, Sidra Tahir. "Adjustment of Bed for a Patient Through Gesture Recognition: An Image Processing Approach", 2018 IEEE 21st International Multi-Topic Conference (INMIC), 2018 Publication	1 %
7	www.researchgate.net Internet Source	1 %
8	bmcmedinformdecismak.biomedcentral.com Internet Source	1 %
9	Submitted to National Research University Higher School of Economics Student Paper	<1 %
10	ecsjournal.org Internet Source	<1 %
11	www.ijfrcsce.org Internet Source	<1 %
12	Submitted to Herzing University Student Paper	<1 %
13	Submitted to University of Wolverhampton Student Paper	<1 %

14	atna-mam.utcluj.ro Internet Source	<1 %
15	www.ijirset.com Internet Source	<1 %
16	link.springer.com Internet Source	<1 %
17	ru.scribd.com Internet Source	<1 %
18	G. Pavithra, T. C. Manjunath, Krishnananda, Dharmanna Lamani. "Detection of secondary glaucoma in human eyes using sophisticated bio-medical image processing algorithms", 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), 2017 Publication	<1 %
19	Submitted to SASTRA University Student Paper	<1 %
20	nlist.inflibnet.ac.in Internet Source	<1 %
21	repub.eur.nl Internet Source	<1 %
22	"HCI in Business, Government and Organizations", Springer Science and Business Media LLC, 2022 Publication	<1 %

23	utpedia.utp.edu.my Internet Source	<1 %
24	G Pavithra, Anusha Tugashetti, T. C. Manjunath, L Dharmanna. "Investigation of primary glaucoma by CDR in fundus images", 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017 Publication	<1 %
25	M. Rohith, R Sainivedhana, N. Sabiyath Fatima. "IoT Enabled Smart Farming and Irrigation System", 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), 2021 Publication	<1 %
26	repository.javeriana.edu.co Internet Source	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

CO-PO Mapping Justification Sheets

CO-PO Mapping:

Mapping of Course Outcomes to Program Outcomes and Program Specific Outcomes:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	3	2	3	3	2	3	3	3	3	3	3	2
CO2	3	2	2	3	3	2	3	2	2	3	2	3	2	3
CO3	2	3	3	2	3	3	3	3	3	3	3	2	3	2
CO4	3	3	3	2	2	3	3	2	3	2	2	3	3	2
CO5	3	2	2	3	2	3	2	3	2	3	2	2	2	3
CO6	2	3	2	2	3	2	3	2	3	3	3	2	3	2
CO AVG	2.67	2.67	2.5	2.34	2.67	2.67	2.67	2.5	2.67	2.83	2.5	2.5	2.67	2.34

*High=3, Medium=2, Low=1

Justification

CO1	Applied knowledge of image processing, computer vision & python programming
CO2	Studied various papers & identified the problems with current system & developed solutions
CO3	Developed integrated solution including various technologies & concepts
CO4	Builds software & hardware for the proposed project & developed programming logic for hand gesture and voice recognition
CO5	Developed solution for disabled patients as well as elderly people
CO6	Completed the work (Hardware & Software) within the stipulated time

Budget Estimation Sheets

Sl. No.	Particulars	Estimated Cost in Rs.
1	Raspberry pi 3 B	7000
2	H-Bridge L293D	200
3	DC Stepper Motor	300
4	Power Supply 5V & 12V V1.2 Module	200
5	Raspberry pi Camera	700
6	Bluetooth bc417 Module	600
7	Power Supply Code 2A	200
8	USB 2.0 UART Serial Converter	200
9	Soldering & Setup	200
10	B Type USB Cable Cord	200
11	Temperature Sensor & Gas Sensor	260+225=485
12	Max30100 Spo2 & Heart Rate Sensor	550+215=765
	Total	11050/-