Chapter 1

INTRODUCTION

1.1 Virtual Telepresence

Virtual telepresence refers to the use of technology to create a sense of presence and enable remote interaction with people or environments. It allows individuals to virtually be in a different location and engage in real-time communication and collaboration as if they were physically present.

Virtual telepresence finds applications in various fields, including business, education, healthcare, and entertainment. It enables remote collaboration, virtual meetings, remote learning, remote medical consultations, and immersive experiences.

By utilizing virtual telepresence technologies, individuals can transcend the limitations of physical distance and engage in meaningful interactions and experiences with others, regardless of their geographical location.

Virtual Reality (VR): Virtual reality technology creates immersive digital environments that simulate physical presence in a computer-generated world. Users wear VR headsets that provide a 360-degree view and can interact with virtual objects and other participants.

Augmented Reality (AR): Augmented reality overlays digital information or virtual objects onto the real-world environment. Through AR applications or devices, users can see and interact with virtual elements while maintaining their presence in the physical space.

1.2 What is Virtual Telepresence Robot?

A virtual telepresence robot is a remote-controlled robot that allows a user to see and interact with a remote environment as if they were there in person. The robot is equipped with a camera and microphone, and the user can control the robot's movement and view using a computer or mobile device.

1.3 Existing Systems:

Double Robotics: Double Robotics offers the Double, a popular telepresence robot. It features a self-balancing base with an extendable pole that holds an iPad or tablet. Users can control the robot remotely, move it around, and engage in video conferencing.

Suitable Technologies Beam: Beam is a telepresence robot that combines a motorized base with a large screen and camera. It allows users to remotely control the robot, navigate in a remote location, and interact through video conferencing.

OhmniLabs: OhmniLabs provides telepresence robots with a focus on simplicity and ease of use. Their robots feature a screen, camera, and microphone array, enabling remote presence and interaction.

Ava Robotics: Ava Robotics offers a telepresence robot designed for businesses and institutions. Their robot combines mobility with a large display, allowing remote users to navigate and interact with others in real-time.

1.4 Proposed System

The proposed system aims to use the concept of telepresence robots incorporating Virtual Reality to create a VR surveillance robot. The novelty of our robot is that no conventional controllers are used for the motion control rather the user's movement in the physical space itself is mapped to the robot. The VR design is based on a Raspberry Pi to which a camera module is connected. The live footage captured by the camera is then streamed wirelessly to an android smartphone running the VR app. The smartphone is placed in a VR headset worn by the user

The smart phone reads the accelerometer and magnetometer data of the direction in which the user turns his head, say, right or left. This data is sent to the modem over Wi-Fi and to the Raspberry Pi board, which, in turn, provides these values as inputs to the servo motors.

The captured visuals are displayed on the user’s virtual reality (VR) headset along with obstacle detection using image processing. The robot can also be moved in any direction (in which the user turns his head, say, right or left) through an app installed in the user’s smartphone.

Two servo motors are used to move the camera—one for the vertical movement and the other for the horizontal movement. So, when you turn your head along with VR headset to the right side, the Raspberry Pi camera will also turn to the right direction. The smartphone also provides input to the Wi – fi for the purpose of navigation or movement of the robot camera . The motor driver IC and geared motors are connected at the end of the navigation circuit. The commands to run the robot can be sent via Bluetooth from the smartphone. In this example, Bluetooth is used.

1.5 The main objectives of the project:

1. To design a robot for virtual telepresence.
2. Using Raspberry pi to achieve the task.
3. To send data to the modem using Wi-Fi.
4. This robot can send the live visuals along with obstacle detection to the VR headset.

1.6 Applications

- The robot can be used in fire and rescue operations.
- In the Medical case, at times when the doctor is not able to go on rounds, the robot can be used to check the state of the patient.
- As the robot capture the visuals and classify the objects so it can be used in archeology survey.
- The robot gives the real time visual experience, it is helpful for users to view real estate virtually.

Chapter 2

LITERATURE SURVEY

Virtual Telepresence Robot Using Raspberry Pi Srinivas Institute of Technology, Valachil, India (July 2020).

Now-a-days time is precious for all humans. In order to avoid to avoid the difficulties and make the life easiest way we introduce robots to the world. In this project we are planning to monitor a different place which is far away from the user so in order to save the time from travel from his origin place to desired monitoring place we are introducing the virtual telepresence robot. To meet our expectations, we can use both technologies and they are virtual reality and direction controlled robot. Virtual reality has been seen from past few years, they had been widely used in some different fields like chemical field and the field of graphics and video games. The virtual reality gives the person a special effect that makes the person attain the emotions that the user could feel it as real.

Basically robot can be controlled using different methods here we can control our robot using a remote or from our smart phone screen by providing the directions as input. As we already know robot are invented to reduce the work load of human and make the work in faster way. Robots are non living things; they work according to the data or program which is initially installed or stored in it. In our project we are using raspberry pi which we code and store program based on which the robot works. This project gives the user with advantages of virtual reality which makes user as if being present in the desired location without physically being there and control the robot from a far distance the robot can move by the commands of user on which direction he wants to move the robot. Here RPI camera will be placed on the robot for capture the image or video and send it back to the data to the user on his smart phone or display.

Jingxin Zhang Department of Informatics, University of Hamburg, Germany

Telepresence systems should allow humans to move through the remote environment, interact with remote artifacts or communicate with the remote people. However, current telepresence systems usually lack natural ways of supporting interaction and exploration of remote environments (REs). In particular, most current telepresence platforms consist of mobile webcams with speakers and microphones. As a result, the usage of single webcams for capturing the RE provides the users with a very narrow field of view and a limited illusion of spatial presence. Both issues limit the sense of presence of teleoperators. In addition, the deficiency of visual information about the RE can lead to a high error-rate for teleoperation tasks and remote interactions. Furthermore, typical movement controls of mobile platforms in today's telepresence systems are often `restricted to simple interaction devices, such as joysticks, touchpads, mice or keyboards. As such operators have to use their hands in order to control the mobile platform and, therefore, the hands are not free to perform other tasks simultaneously. This may decrease the naturalness, task performance and overall user experience.

Nandagopal Harikrishnan Dept. of ECE Mar Baselios College of Engineering and Technology Thiruvananthapuram, India

The first stage of the work focuses on the Arduino based four wheeled robots. The board that will be used is an Arduino coupled to 4 geared motors which power the wheels. The Bluetooth module will also be connected to the Arduino board. First stage of testing would involve controlling the robot using a basic Bluetooth enabled controller. Building the pedometer comes in the second stage. The values obtained from the accelerometer contain unwanted noise signals. It may badly affect the step counting. The noise signal must be properly filtered using appropriate filters to initiate step counting.

For this purpose, we are using Kalman filters. It is an optimal estimator which creates an estimate of the unknown value using the previous state estimate and the current input data. This algorithm is based on minimizing the mean square deviation. The Kalman filter process is as follows: Predict the next covariance as: Pc=P + varProcess. where Pc-predicted covariance, p-covariance of the sensor data and varProcess-process covariance. Compute the Kalman gain: G=P/ (Pc + varVolt) where G-Kalman gain and varVolt-variance determined using excel and reading samples of raw sensor data. Update the covariance of the sensor output: P=(1-G) *Pc Compute the Kalman estimate of the sensor voltage: Xe=G*(netmag-Xp) + Xp where Xe-Kalman estimate of the sensor voltage, Xpprevious Kalman estimate and netmag is the net magnitude of the sensor output along the x,y and z axes.

The net magnitude is computed as: netmag x2 + y2 + z2 Once we have the filtered output, we need to determine a threshold value to be set to initiate step counting. For that purpose, different users were allowed to wear the smartphone in the band and made to walk in different ways. The accelerometer values are continuously monitored for each user and finally a threshold value is determined and was set in the program code Since the determination of threshold experiment is conducted in many peoples, the advantage is that even the minor movements in the legs or hands will not result in step counting. The rotation of the robot according to the user's motion is controlled using the magnetometer values from the smartphone worn by the user. All the computations are performed with respect to the Earth’s magnetic field. The robot is coded in such a way that when it is powered on, it initially checks the user direction and changes its direction accordingly.

For example, when the robot is powered on, assume the user is standing in the north-east direction while the robot is facing the north direction. The robot detects a mismatch in direction and in order to correct it turns to the right by 45 degrees. After matching the directions, it will continue to move like the user. After every movement the robot direction is compared with the user direction to check whether the user has turned right or left and if any mismatch is found the robot corrects it’s direction accordingly.

Once the pedometer is tested for accurate step counting, it is then connected to the robot via Bluetooth and the code is modified to act as the controller. Third stage involves integrating Virtual Reality using a Raspberry Pi board to which the camera module is connected. The camera module is rotated by a servo motor arrangement. A camera mount with two servo motors to enable both pan and tilt movement is attached at the top of the robot that resembles the user's head.

After setting the initial data, when the user moves his/her head, the gyroscope values will also change and these values are used for computing how much the camera mount to be moved. In the final stage a dual screen app is used to split the screen that allows the users to open two applications at once on the smartphone that acts as the display for the VR headset. This stage also involves connecting the smartphone to the Raspberry Pi and making the VR headset the directional controller for the camera.

Application of redirected walking in room-scale VR

Eike Langbehn, Paul Lubos, Gerd Bruder, Frank Steinicke

Redirected walking (RDW) promises to allow near-natural walking in an infinitely large virtual environment (VE) by subtle manipulations of the virtual camera. Previous experiments showed that a physical radius of at least 22 meters is required for undetectable RDW. However, we found that it is possible to decrease this radius and to apply RDW to room-scale VR, i. e., up to approximately 5m × 5m.

This is done by using curved paths in the Ve instead of straight paths, and by coupling them together in a way that enables continuous walking. Furthermore, the corresponding paths in the real world are laid out in a way that fits perfectly into room scale

VR. In this research demo, users can experience RDW in a room-scale head-mounted display VR setup and explore a VE of approximately 25m × 25m.

2.1 Embedded Systems

2.1.1 Embedded Systems

An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.

Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

2.1.2 Real Time Issues:

Embedded systems frequently control hardware, and must be able to respond to them in real time. Failure to do so could cause inaccuracy in measurements, or even damage hardware such as motors. This is made even more difficult by the lack of resources available. Almost all embedded systems need to be able to prioritize some tasks over others, and to be able to put off/skip low priority tasks such as UI in favor of high priority tasks like hardware control.

2.1.3 Need for Embedded Systems:

The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task or set of tasks costs far more time and money.

Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed. From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real-Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.

2.1.4 Debugging:

Embedded debugging may be performed at different levels, depending on the facilities available. From simplest to most sophisticate they can be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multicore systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in-circuit emulator replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.
- Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as assembly code or source-code.
- Review error codes and logs: Check system error codes, logs, and any available output to gather additional information about the problem. Pay attention to error messages, warnings, and any other feedback that the system provides.

2.2 Applications of Embedded Systems

Consumer applications:

At home we use a number of embedded systems which include microwave oven, remote control, vcd players, dvd players, camera etc..

Office Automation :

We use systems like fax machine, modem, printer etc…

Industrial automation:

Today a lot of industries are using embedded systems for process control. In industries we design the embedded systems to perform a specific operation like monitoring temperature, pressure,humidity ,voltage, current etc.., and basing on these monitored levels we do control other devices, we can send information to a centralized monitoring station.

In critical industries where human presence is avoided there we can use robot**s** which are programmed to do a specific operation.

Fig 2.1: Robot

Tele communications:

Cell phones, web cameras etc.

Security:

Embedded systems are used in security systems such as burglar alarms and fire alarms.

Aerospace:

Embedded systems are used in aircraft and spacecraft.

Chapter 3:

HARDWARE AND SOFTWARE REQUIREMENTS

3.1 Hardware Requirements

- Raspberry Pi
- Raspberry Pi Camera Module
- Bluetooth Module HC - 05
- Servo motors
- VR Headset
- Voltage Regulator
- l293d Motor Driver
- DC motors
- Bread board
- SD Card
- Batteries 12V 2A
- Charger

3.2 Software Requirements

- Raspberry Pi Operating System
- Bluetooth HC – 05 Application
- Dual Browser Application
- Network Analyzer Application
- Wireless IMU Application
- Python

3.3 Hardware Description

3.3.1 About Raspberry pi

The **Raspberry Pi** is a credit-card-sized single-board computer developed in the UK by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools. The Raspberry Pi is manufactured through licensed manufacturing deals with Newark element14 (Premier Farnell), RS Components and Egoman.

The Raspberry Pi 3 Model A+ is the latest product in the Raspberry Pi 3 range. Like the Raspberry Pi 3 Model B+, it boasts a 64-bit quad core processor running at 1.4 GHz, dual-band 2.4 GHz and 5 GHz wireless LAN, and Bluetooth 4.2/BLE.

Pin Description

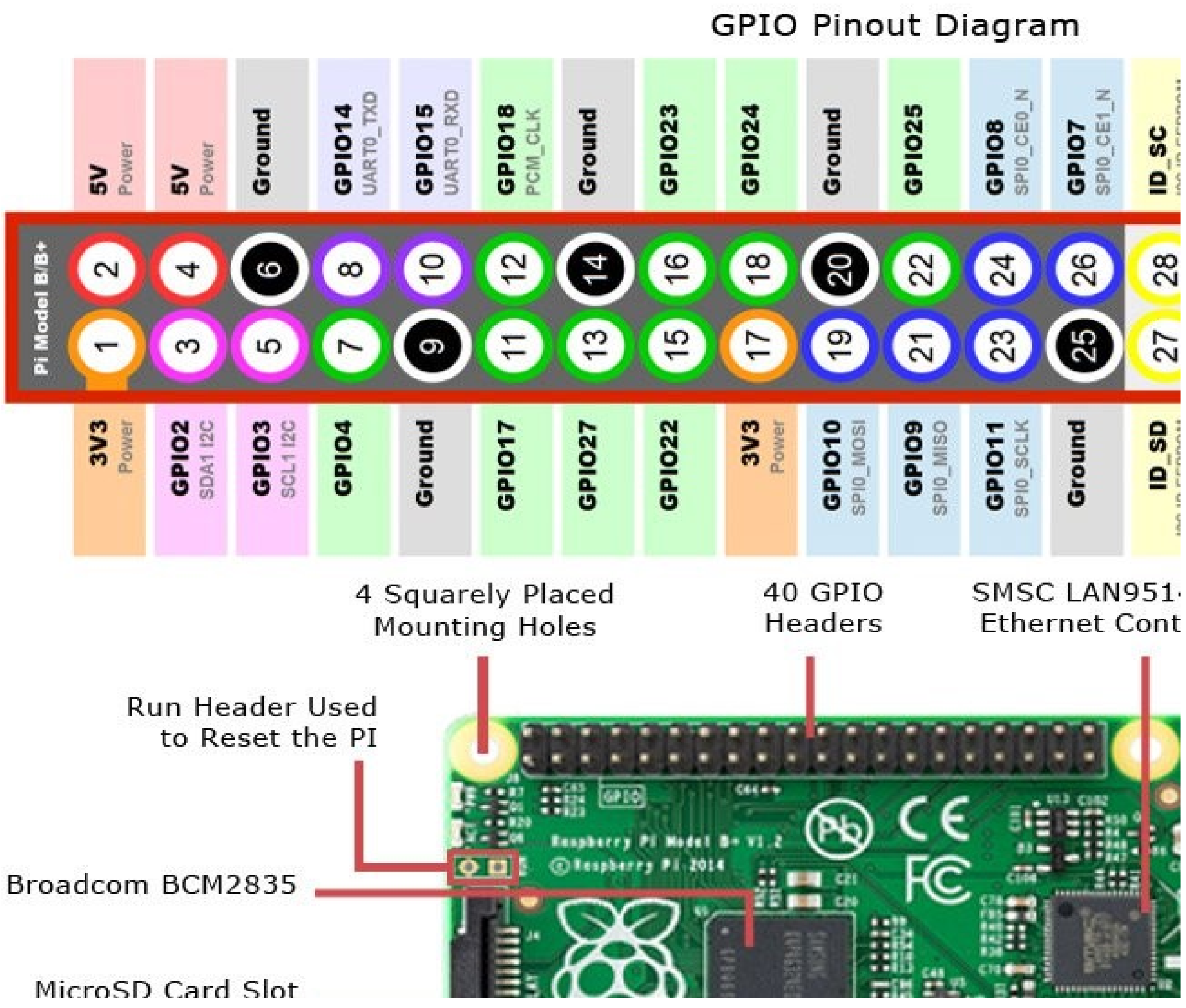


Fig 3.1: Raspberry Pi pin description

3.3.2 Camera Module

The camera consists of a small (25mm by 20mm by 9mm) circuit board, which connects to the Raspberry Pi's Camera Serial Interface (CSI) bus connector via a flexible ribbon cable. The camera's image sensor has a native resolution of five megapixels and has a fixed focus lens. The software for the camera supports full resolution still images up to 2592x1944 and video resolutions of 1080p30, 720p60 and 640x480p60/90. The camera module is shown below:

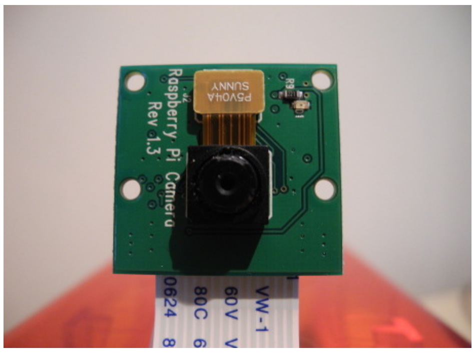


Fig 3.2: Raspberry Pi Camera

3.3.3 Bluetooth module HC – 05

The Bluetooth module HC-05 is a MASTER/SLAVE module. By default the factory setting is SLAVE. The Role of the module (Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc. Just go through the datasheet for more details.

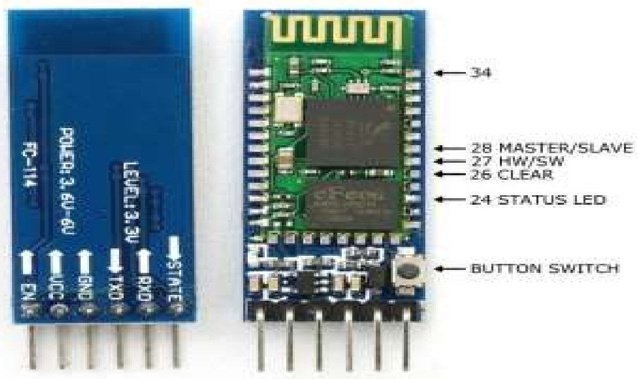


Fig 3.3: HC – 05 Bluetooth

3.3.4 Servo Motors

A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

Servomotors are not a specific class of motor although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system.

Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.

3.3.5 DC Motor

A dc motor uses electrical energy to produce mechanical energy, very typically through the interaction of magnetic fields and current-carrying conductors. The reverse process, producing electrical energy to mechanical energy by an alternator, generator or dynamo. Many types of electric motors can be run as generators, and vice versa. The input of a DC motor is current/voltage and its output is torque (speed).

3.4 Software Description

3.4.1 PYTHON

Python is the best programming language to use in this project to communicate with the user on the Raspberry Pi. Python is a general-purpose interpreted, interactive, object-oriented, and highlevel programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is a widely used high-level programming language for general-purpose programming. Python programs don't need to be compiled before running them. However, the Python interpreter must be installed on the computer to run them. The Python interpreter is a program that reads Python files and executes the code. The python_camera.py code is used in the RPi board to control the servo motors.

3.4.2 RASPBIAN OS

Raspberry Pi OS (formerly Raspbian) is a Debian-based operating system for Raspberry Pi. Since 2015, it has been officially provided by the Raspberry Pi Foundation as the primary operating system for the Raspberry Pi family of compact single-board computers. The first version of Raspbian was created by Mike Thompson and Peter Green as an independent project.

The initial build was completed in June 2012. Raspberry Pi OS is highly optimized for the Raspberry Pi line of compact single-board computers with ARM CPUs. It runs on every Raspberry Pi except the Pico microcontroller. Raspberry Pi OS uses a modified LXDE as its desktop environment with the Open box stacking window manager, along with a unique theme.

3.4.3 Linux Operating System

Linux is a Unix-like and mostly POSIX-compliant computer operating system assembled under the model of free and open source software development and distribution. The defining component of Linux is the Linux kernel, an operating system kernel first released on 5 October 1991 by Linus Torvalds. The Free Software Foundation uses the name GNU/Linux, which has led to some controversy.

The Linux Standard Base (LSB) is a joint project by several Linux distributions under the organizational structure of the Linux Foundation to standardize the software system structure, including the file system hierarchy used in the GNU/Linux operating system. The LSB is based on the POSIX specification, the Single UNIX Specification, and several other open standards, but extends them in certain areas.

3.4.4 Bluetooth Terminal HC – 05 Application

The Bluetooth Terminal HC-05 application is an Android app that provides a convenient and user-friendly interface for interacting with the HC-05 Bluetooth module.

Bluetooth Connection: The app allows users to scan for available Bluetooth devices and establish a connection with the HC-05 module. It provides a list of nearby Bluetooth devices and facilitates the pairing process if required.

Terminal Interface: The main interface of the app resembles a terminal or console window where users can enter commands and receive responses from the HC-05 module. It provides a text input field for entering commands and a display area to show the received data from the module.

Text Input and Output: The app allows users to type and send commands or messages to the HC-05 module via the terminal interface. It supports sending text-based data and special characters. The received data from the module is displayed in the terminal interface, allowing users to view the responses, sensor readings, or any other information sent by the module.

Chapter 4

SYSTEM ANALYSIS

4.1 System Analysis

System analysis refers to the process of studying and understanding complex systems to identify their components, functions, interactions, and behavior. It involves a systematic and detailed examination of a system's structure and processes to gain insights into its operation and make informed decisions about its design, improvement, or troubleshooting.

The goal of system analysis is to gain a comprehensive understanding of the system's current state, identify areas for improvement, and propose solutions to meet the stakeholders' needs. It forms the basis for system design, development, and implementation phases, ensuring that the system is aligned with user requirements and effectively supports the desired functionality and objectives.

4.2 Use Case Diagram

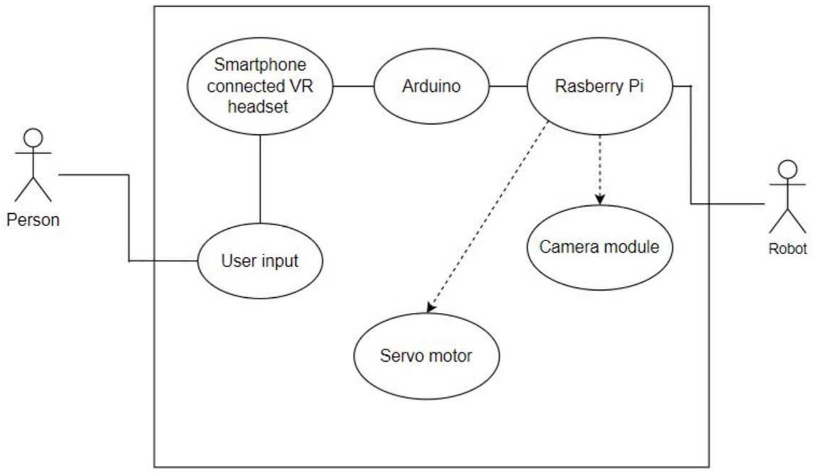


Fig 4.1: Use case diagram

The user initiates and interacts with multiple use cases, such as start virtual presence, control robot movements, view remote environment, communicate with remote users, adjust camera view, and end virtual presence. In this use case the user adjusts the camera angle or view to optimize their visibility and perspective of the remote environment.

The use case diagram provides an overview of the functionalities and interactions of the virtual telepresence robot system. It helps user to understand the system's capabilities and how different actors interact with it, supporting the development and design process.

4.3 Data flow diagram

4.3.1 Initiation of the process

The initiation of process of user and robot consist of that the user need to first initialize the serial communication between the Bluetooth module and the Bluetooth Terminal HC – 05 application and setting the pins in the application in order to provide input to the DC motor for the movement of the robot. Then user send the data through Bluetooth Terminal HC – 05 application and these data characters are read by the Bluetooth module and it will send signals to the robot according to the user input.

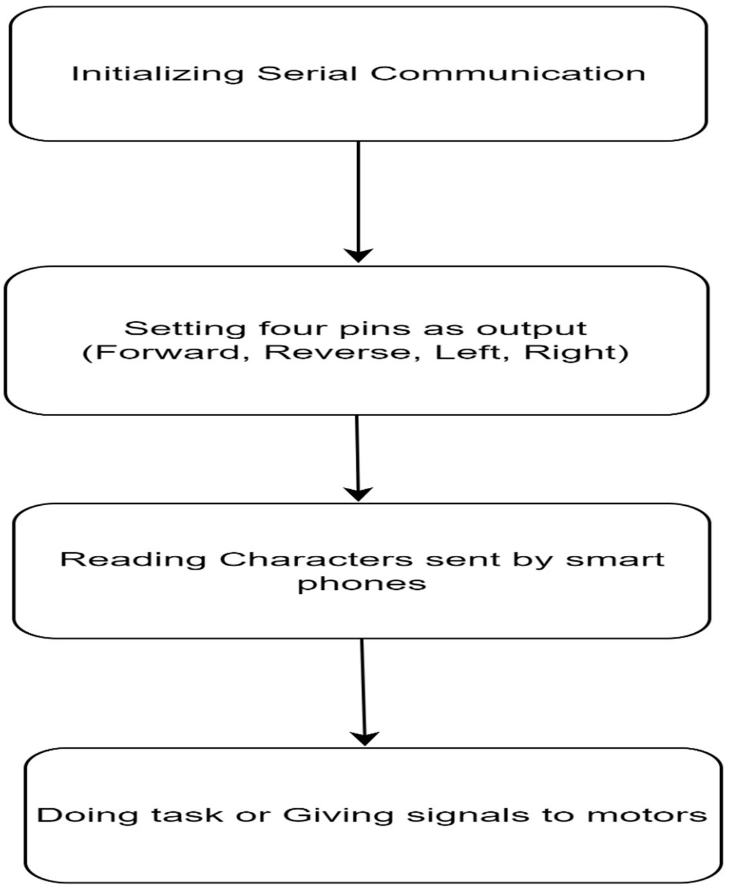

Initializing Serial Communication
Setting four pins as output (Forward, Reverse, Left, Right)
Reading Characters sent by smart phones
Doing task or Giving signals to motors

Fig 4.2: Initiation of process

4.3. Capturing the live visuals

The process will begin by pairing with the target device with the verification of target IP address and port number then the robot will receive the data packets and these packets contain values with respect to camera angle and direction so according to the users input the camera angle will change and visuals are displayed on the screen.

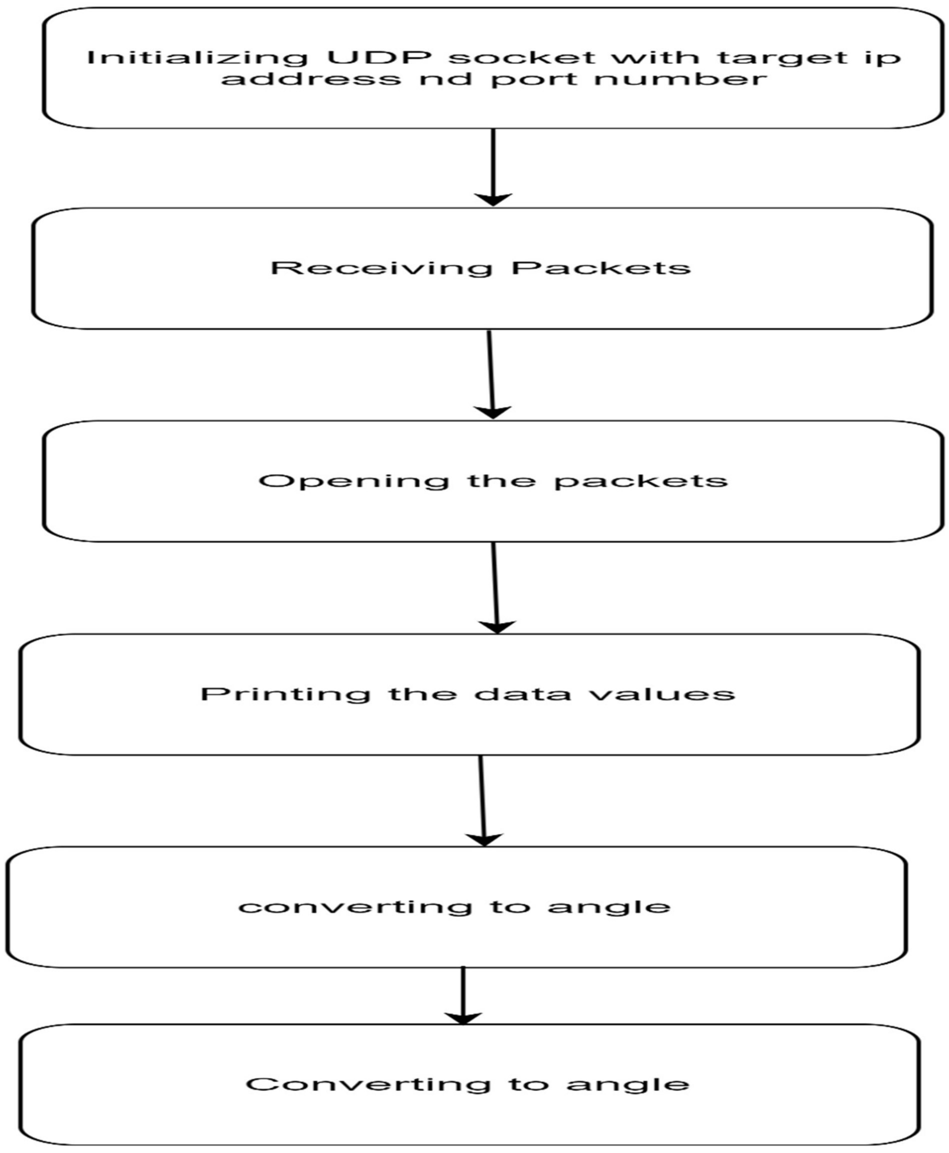


Fig 4.3: capturing live video

4.4 Functional requirement

1. To design a robot for virtual telepresence.
2. Using Raspberry pi to achieve the task.
3. To send data to the modem using Wi-Fi.
4. This robot can send the live visuals along with obstacle detection to the VR headset.

4.5 Feasibility Study

A feasibility study is an assessment conducted to determine the practicality and viability of a proposed project or initiative. It evaluates various factors to determine whether the project is technically feasible, economically viable, and operationally achievable. The study aims to provide decision-makers with objective and reliable information to determine whether to proceed with the project or explore alternative options.

A feasibility study for a virtual telepresence robot would assess the viability and practicality of implementing such a system. Here are the key aspects that would be considered in a feasibility study for a virtual telepresence robot:

1. Technical Feasibility:

Evaluation of the required technology and infrastructure for the virtual telepresence robot, including the robot hardware, communication systems, camera systems, and control mechanisms.

Assessment of the compatibility of the proposed technology with existing systems and networks.

Examination of any technical challenges or limitations that may arise during the implementation of the virtual telepresence robot.

2. User Acceptance and Social Feasibility:

Evaluation of user acceptance and adoption of the virtual telepresence robot, considering factors such as user experience, ease of use, and potential benefits for users.

Assessment of the social impact and acceptance of the robot within the intended environment, including cultural factors, social norms, and potential concerns or resistance from users.

3. Operational Feasibility:

Assessment of the impact on existing operations and workflows within the organization or environment where the virtual telepresence robot will be deployed.

Evaluation of the required human resources, skills, and training necessary to operate and maintain the robot effectively.

Consideration of potential risks, security concerns, and measures for ensuring the smooth operation and integration of the virtual telepresence robot into the existing setup.

By conducting a comprehensive feasibility study covering these aspects, organizations or stakeholders can make informed decisions about the viability and potential success of implementing a virtual telepresence robot.

Chapter 5

SYSTEM DESIGN

Virtual reality, robotics, and Augmented reality can team up to develop innovative applications for various organizations. In this project a robot with a camera is placed in a remote location to capture the environment in visual form using Raspberry Pi (RPi). The captured visuals are displayed on the user's virtual reality (VR) headset. An added feature allows the camera to move in the direction of the user's head movements. This gives the user a real time experience a if he is present where the virtual tele-presence robot is located. The virtual telepresence robot can also be moved in any direction through an app installed in the user's smartphone.

5.1 Architecture Diagram

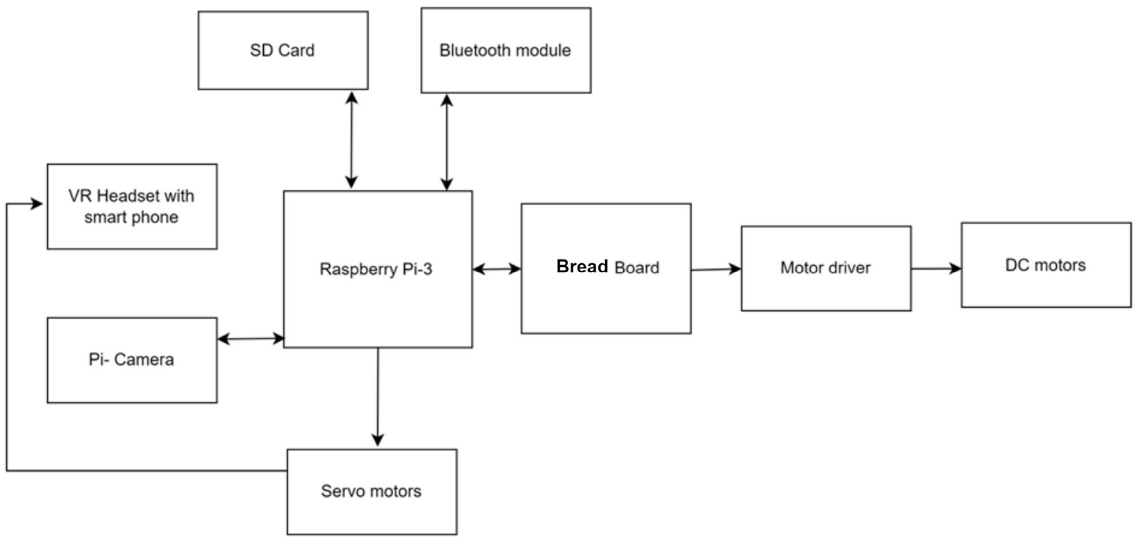


Fig 5.1: Architecture diagram

The VR telepresence robot consists of a 4 wheeled robot as the basic structure. A camera mount with servo motors is attached at the top of the robot in order to capture the live video footage of the environment where it is present. This video is displayed on the smartphone placed in the VR headset worn by the user. The user's motion is mapped into the robot for its movement in that particular environment.

The overall system of the Virtual Interactive Reality Telepresence robot can be split up into VR section, robotics section and user section. The VR section has a Raspberry Pi 3 Model A+ at its heart. The Video footage is taken by a 5 MP 1080p Raspberry Pi camera module. The pan and tilt movement of the camera module can be achieved with the help of two SG-90 servo motors connected to the Raspberry Pi. This camera movement corresponds to the user's head movement. The Robotics section is basically a 4 wheeled Robot. It uses 4 geared motors to drive the 4 wheels. The motors are driven by an L293D motor driver shield which is powered by the

The robot movement is made to correspond to the user movement using data transmitted via HC-05 Bluetooth module. The User Section consists of the wearable VR headset holding an android smartphone running the dual screen app. The Android phone displays the camera footage received from the camera module of the Raspberry Pi. The user head movement data is sent from the Wireless IMU app in the smartphone to the Raspberry Pi via Wi-Fi. The User Section also comprises a smartphone installed with the dabble application. Step count is taken by manipulating the phone sensor values. This "step data" is then transmitted to the wirelessly via Bluetooth.

5.2 Sequence Diagram

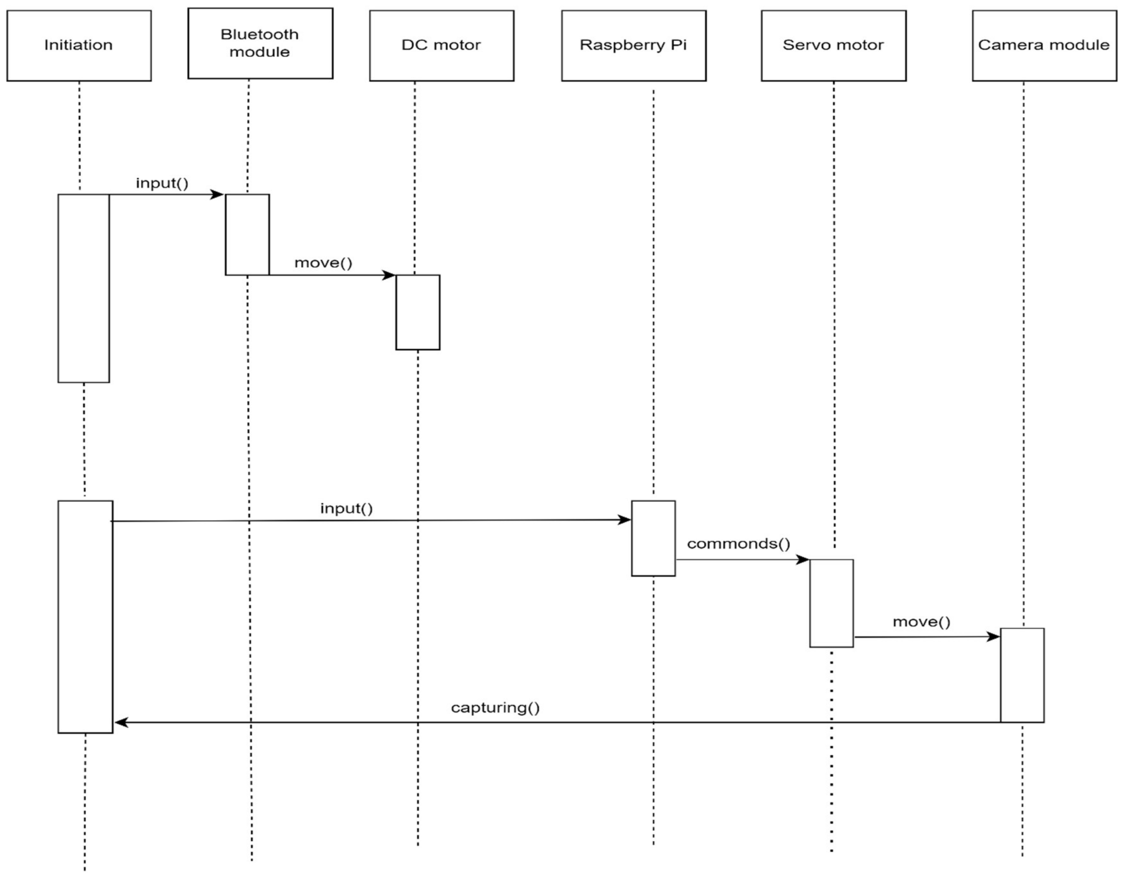


Fig 5.2: Sequence diagram

In this sequence diagram, the User initiates the virtual presence mode. The User can control the robot's movements, and the Virtual Telepresence Robot processes the movement commands and updates the robot's position. The User can also interact with objects using the robot's manipulator. The Virtual Telepresence Robot acknowledges the activates the of camera system. It provides a live video feed.

Furthermore, the User can communicate with remote location. This involves a bidirectional conversation between the User and a Remote location. The User can adjust the camera view, and the Virtual Telepresence Robot updates the camera angle and provides an updated video feed accordingly. Finally, when the User decides to end the virtual presence, the Virtual Telepresence Robot deactivates the camera system terminates the connection.

This sequence diagram demonstrates the sequential flow of interactions and messages between the User, Virtual Telepresence Robot, and various system components during different stages of using a virtual telepresence robot.

5.3 Class Diagram

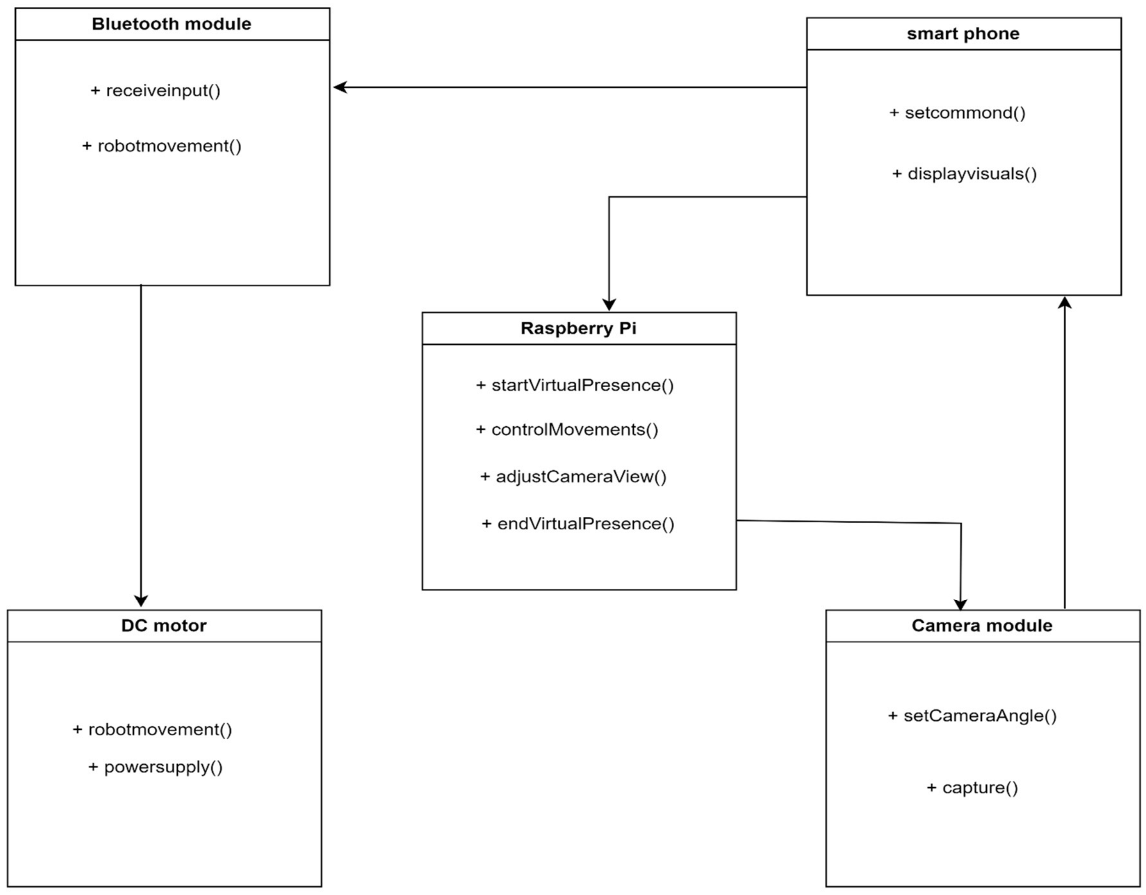


Fig 5.3: class diagram

In this class diagram, the main class is the Raspberry pi, which operates the most of the operations. The class provides methods to perform actions like startVirtualPresence(), controlmovements(), adjustcameraview() and endVirtualPresence().

- The Camera module class represents the camera system of the robot and has method like setcameraangle() to change the camera view and capture() to capture the visuals.
- The Bluetooth module class receives the input from users smartphone using receiveinput() and give instructions for the robot movement using robotmovement().
- The smartphone class will send commands to the robot using setcommand() and receive visuals from the camera module and displayed on the screen using displayVisuals().
- The DC motor class will perform movements of the robot using robotmovement() and receive power supply from the batteries using powersupply().

Chapter 6:

IMPLEMENTATION

Implementing a virtual telepresence robot involves several components, including a user interface, video streaming, robot control, and communication between the user and the robot.

Video Streaming: Set up a video streaming system that captures video and audio from the robot's camera and microphone and sends it to the user in real-time. You can use libraries like OpenCV or FFmpeg to handle video capture, encoding, and streaming.

Robot Control: Implement the code to control the robot's movement based on user commands. This typically involves mapping user inputs (e.g., button clicks or keyboard commands) to appropriate robot movements (e.g., forward, backward, turn left, turn right). Depending on your hardware setup, you may need to interface with motor controllers or other components to control the robot's actuators.

Communication: Establish a communication channel between the user interface and the robot control system. This can be done using a protocol such as HTTP or WebSocket. The user interface can send commands to the robot control system, and the control system can send video and audio data back to the user interface.

Integration: Combine all the components together, ensuring they work seamlessly. The user interface should display the video stream, provide controls for robot movement, and enable audio communication between the user and the robot. The robot control system should receive commands from the user interface, control the robot's movement, and send video and audio data back to the user.

Testing and Refinement: Test the virtual telepresence robot system thoroughly, identify any issues or bugs, and refine the implementation as needed to improve performance, usability, and reliability.

6.1 Raspberry Pi Setup:

Setting up a Raspberry Pi in a virtual telepresence robot involves configuring the Raspberry Pi, connecting necessary hardware components, and installing the required software. Here are the general steps to follow:

Raspberry Pi Setup:

Download the Raspberry Pi operating system (Raspberry Pi OS) from the official Raspberry Pi website.

Write the operating system image to an SD card using a tool like Etcher.

Insert the SD card into the Raspberry Pi and power it up.

Follow the on-screen instructions to complete the initial setup, including setting up the network connection and enabling SSH (if required).

Hardware Connections:

Connect the necessary hardware components to the Raspberry Pi. This may include a camera module, microphone, speaker, motor controllers, and any other sensors or actuators required for your robot.

Make sure to follow the hardware documentation and pinout diagrams specific to your Raspberry Pi model and the hardware components you're using.

Software Configuration:

Install the required software packages on the Raspberry Pi. This may include libraries for camera and audio handling, motor control libraries, and any additional software dependencies.

Configure the software to interface with the connected hardware components. This involves setting up camera drivers, configuring audio devices, and ensuring the motor control libraries are properly configured.

Network Configuration:

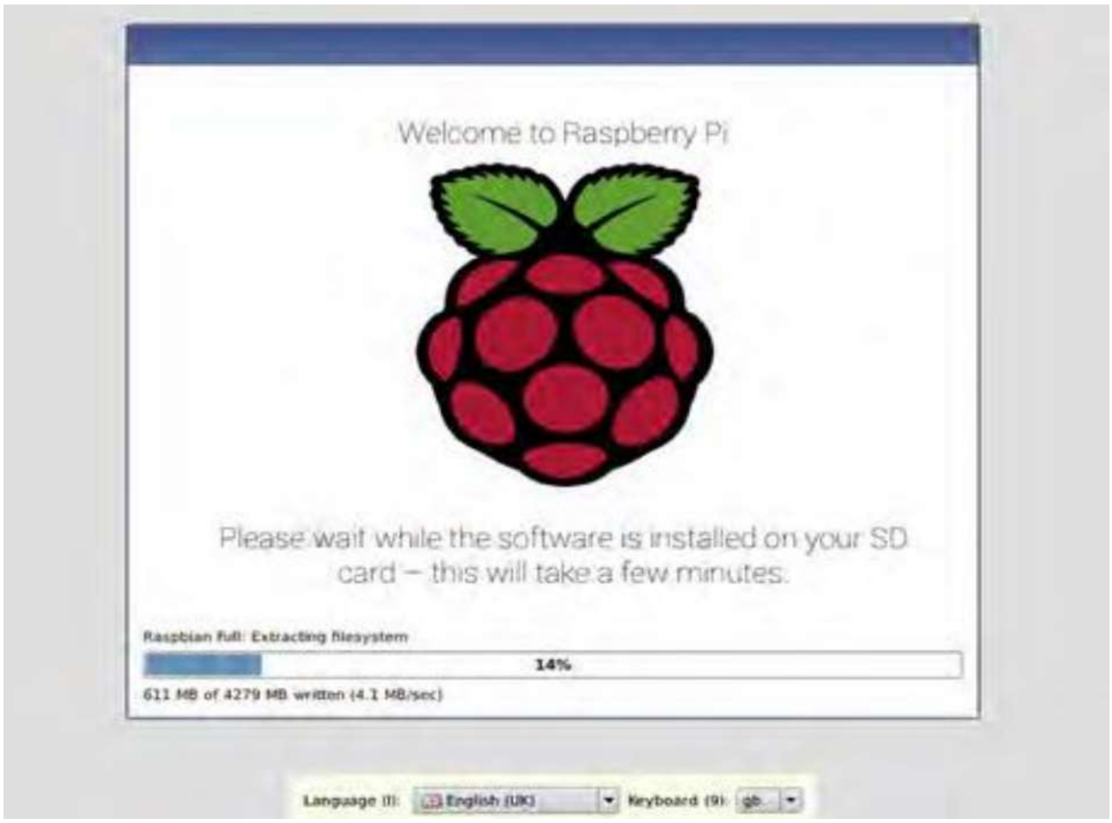
Ensure that the Raspberry Pi is connected to the same network as the user interface device (e.g., PC, laptop, or mobile device) to establish communication between them.

Determine the IP address of the Raspberry Pi and make a note of it for later use.

Control Interface:

Develop or configure a user interface that allows users to control the robot remotely. This can be a web-based interface, a mobile app, or a dedicated control software running on the user interface device.

Implement the necessary controls (e.g., buttons, joystick) in the user interface to send commands to the Raspberry Pi for controlling the robot's movement



Welcome to Raspberry Pi

Please wait while the software is installed on your SD card – this will take a few minutes.

Raspbian Full: Extracting filesystem

14%

611 MB of 4279 MB written (4.1 MB/sec)

Language (l): English (UK) Keyboard (9): gb

Fig 6.1: Installation of Raspberry pi Os

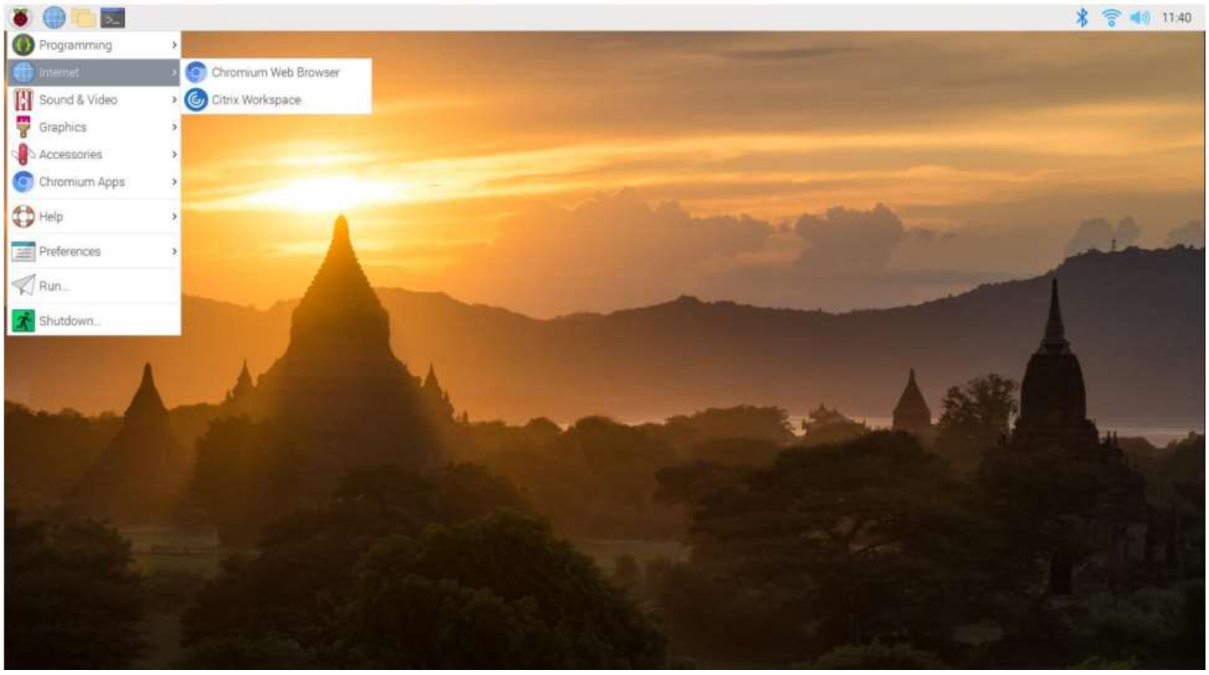


Fig 6.2: Raspbian Desktop

6.2 Bluetooth Module set up

Choose a Bluetooth Module: Select a Bluetooth module that is compatible with your hardware and has the necessary features for your virtual telepresence robot. Bluetooth modules such as HC-05 or HC-06 are commonly used with Arduino or Raspberry Pi.

Connect the Bluetooth module to your microcontroller (e.g., Arduino or Raspberry Pi). This typically involves connecting the VCC pin to a power source (e.g., 3.3V or 5V), GND pin to ground, and the TX and RX pins to the corresponding UART pins (e.g., RXD and TXD) of the microcontroller.

6.3 Wi – fi Module set up

Choose a Wi-Fi Module: Select a Wi-Fi module that is compatible with your hardware and meets the requirements of your virtual telepresence robot. Common Wi-Fi modules used with microcontrollers include ESP8266 and ESP32.

Connect the Wi-Fi module to your microcontroller (e.g., Arduino or Raspberry Pi). This typically involves connecting the VCC pin to a power source (e.g., 3.3V or 5V), GND pin to ground, and the TX and RX pins to the corresponding UART pins (e.g., RXD and TXD) of the microcontroller.

For Raspberry Pi, you can use the appropriate Wi-Fi libraries or modules to connect to your Wi-Fi network and handle communication over TCP/IP or UDP. Additionally, you can utilize other communication protocols like WebSocket or MQTT if needed.

6.4 Power Supply setting up

Determine Power Requirements: Check the power requirements of your Raspberry Pi model. The Raspberry Pi typically requires a 5V DC power supply, and the current requirements vary depending on the model and connected peripherals. Refer to the official Raspberry Pi documentation or the specifications of your specific model to determine the power requirements.

To ensure stable power supply and protect the Raspberry Pi from voltage fluctuations or spikes, consider using a voltage regulator or a power supply module with built-in voltage regulation. This helps to provide a consistent and regulated 5V power supply to the Raspberry Pi.

6.5 Code for robot movement

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
left_motor_pin1 = 11
left_motor_pin2 = 12
right_motor_pin1 = 13
right_motor_pin2 = 15
GPIO.setup(left_motor_pin1, GPIO.OUT)
GPIO.setup(right_motor_pin2, GPIO.OUT)
def move_forward():
    GPIO.output(left_motor_pin1, GPIO.HIGH)
    GPIO.output(left_motor_pin2, GPIO.LOW)
    GPIO.output(right_motor_pin1, GPIO.HIGH)
    GPIO.output(right_motor_pin2, GPIO.LOW)
def move_backward():
    GPIO.output(left_motor_pin1, GPIO.LOW)
    GPIO.output(left_motor_pin2, GPIO.HIGH)
    GPIO.output(right_motor_pin1, GPIO.LOW)
    GPIO.output(right_motor_pin2, GPIO.HIGH)
def stop_robot():
    GPIO.output(left_motor_pin1, GPIO.LOW)
    GPIO.output(left_motor_pin2, GPIO.LOW)
    GPIO.output(right_motor_pin1, GPIO.LOW)
    GPIO.output(right_motor_pin2, GPIO.LOW)
```

6.6 Object detection code using flask

```
from flask import Flask, render_template, Response

import cv2

app = Flask(__name__)

camera = cv2.VideoCapture(0)

net=cv2.dnn.readNetFromDarkne("yolov3.cfg","yolov3.weights")

layer_names = net.getLayerNames()

output_layers=[layer_names[i[0]-1]

for i in net.getUnconnectedOutLayers()]

def detect_objects():

    while True:

        _,frame = camera.read()

        ret, buffer = cv2.imencode('.jpg', frame)

        frame = buffer.tobytes()

        yield(b'--frame\r\n'b'Content-Type: 
image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/')

def index():

    return render_template('index.html')

@app.route('/video_feed')

def video_feed():

    return   Response(detect_objects(),mimetype='multipart/x-
mixed-replace;

if __name__ == '__main__':

  app.run(host='0.0.0.0', port=5000, threaded=True)
```

6.7 Code for the servo motors movement

```
  import RPi.GPIO as GPIO

  import time

  GPIO.setmode(GPIO.BOARD)

  servo_pin = 11

  frequency = 50

  duty_cycle_min = 2

  duty_cycle_max = 12

  GPIO.setup(servo_pin, GPIO.OUT)

  pwm = GPIO.PWM(servo_pin, frequency)

  def set_angle(angle):

      duty_cycle=(angle/180.0)*(duty_cycle_max-duty_cycle_min)
  + duty_cycle_min

      pwm.ChangeDutyCycle(duty_cycle)

  try:

      pwm.start(duty_cycle_min)

      set_angle(0)

      time.sleep(1)

      set_angle(90)

      time.sleep(1)

      set_angle(180)

      time.sleep(1)

  except KeyboardInterrupt:

      pass

  pwm.stop()

GPIO.cleanup()
```

Chapter 7

SYSTEM TESTING

System testing for a virtual telepresence robot involves evaluating the overall functionality, performance, and reliability of the system as a whole. It aims to ensure that the robot meets the specified requirements and operates correctly in various scenarios.

The goal of system testing is to validate the system's compliance with functional and non-functional requirements, its behavior under various scenarios, and its ability to deliver the desired functionality to end users. It aims to identify defects, errors, and inconsistencies that may arise due to interactions between different components or modules within the system.

7.1 Testing of Robot movement controls.

Robot movement controls have mainly four test cases. The forward button test cases is implemented to check weather the robot move forward. The backword button test cases is implemented to check whether the robot move backward. Similarly the other two test cases is implemented to check left and right movements. All the test cases are passed.

Sl No.	**Test Case**	**Expected Outcome**	**Result**
1	Press forward button	Move forward	Pass
2	Press Backward button	Move backward	Pass
3	Press left button	Move left	Pass
4	Press right button	Move right	Pass

7.2 Testing of video streaming according to user's VR head set movement.

The video streaming mainly has four test cases according to user VR headset movement the test case head movement up is implemented to check the upward movement of camera. The test case head movement down is implemented to check the downward movement of camera similarly the other two test cases is implemented to check left and right movements of camera. All the test cases are passed.

Sl. No	Test Case	Expected Outcome	Result
1	Head movement – up	Camera movement - up	Pass
2	Head movement – down	Camera movement–down	Pass
3	Head movement – left	Camera movement – left	Pass
4	Head movement – right	Camera movement- right	Pass

7.3 Testing of power supply

The power supply has mainly two test cases the test case supply power is implemented to check the charging of the battery the test case stop power supply is implemented to check whether it stops charging. All the test cases are passed.

Sl. No	Test Case	Expected Outcome	Result
1	Supply power	charged	Pass
2	Stop power supply	Stops charging	Pass

7.4 Testing of voltage regulator

The voltage regulator has mainly two test cases the test case regulate the voltage is implemented to check whether the robot will receive 5 volts of supply. The test case receive power supply from battery is implemented to check whether the voltage regulator will receive power. All the test cases are passed.

Sl. No	Test Case	Expected Outcome	Result
1	Regulate the voltage to 5 volts	Robot receives 5volts	Pass
2	Receive power supply from battery	Voltage regulator receives power	Pass

7.5 Testing of Raspberry Pi

The Raspberry Pi has mainly two test cases, the test case receiving input from Bluetooth module and direct it to DC motors is implemented to check movement of robot and the second test case is implementation of Object detection programme in SD card to check whether Object detection visuals displayed by Pi camera. All the test cases are passed.

Sl. No	Test Case	Expected Outcome	Result
1	Receiving input from Bluetooth module and direct it to DC motors	Movement of robot	Pass
2	Implementation of Object detection programme in SD card	Object detection visuals displayed by Pi camera	Pass

Chapter 8

Results

After assembling the hardware and installing and running the software we successfully demonstrated that the telepresence robot provides us the video streaming in the direction of our requirement.

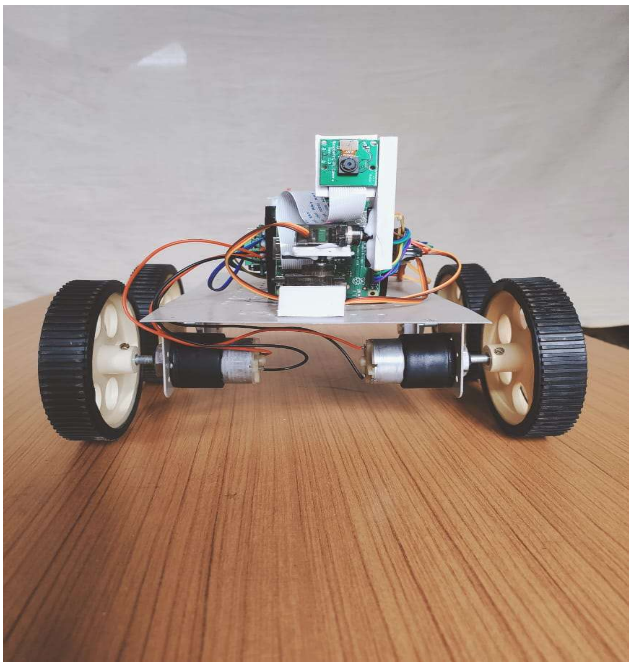


Fig 8.1: front view of robot

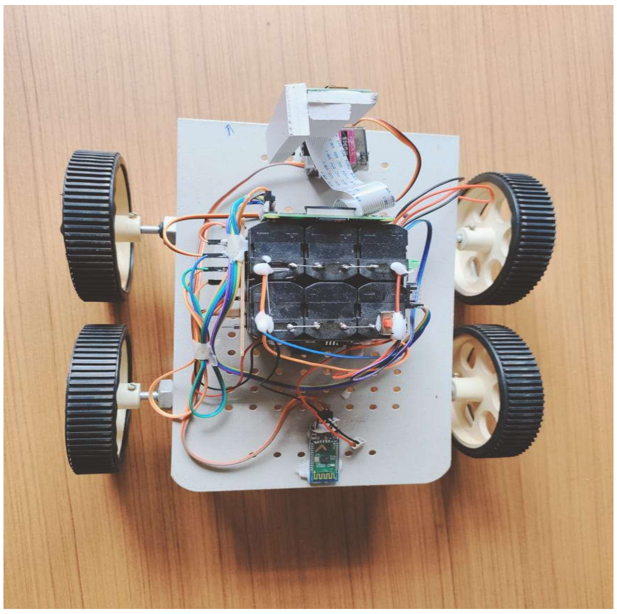


Fig 8.2: Top view of robot

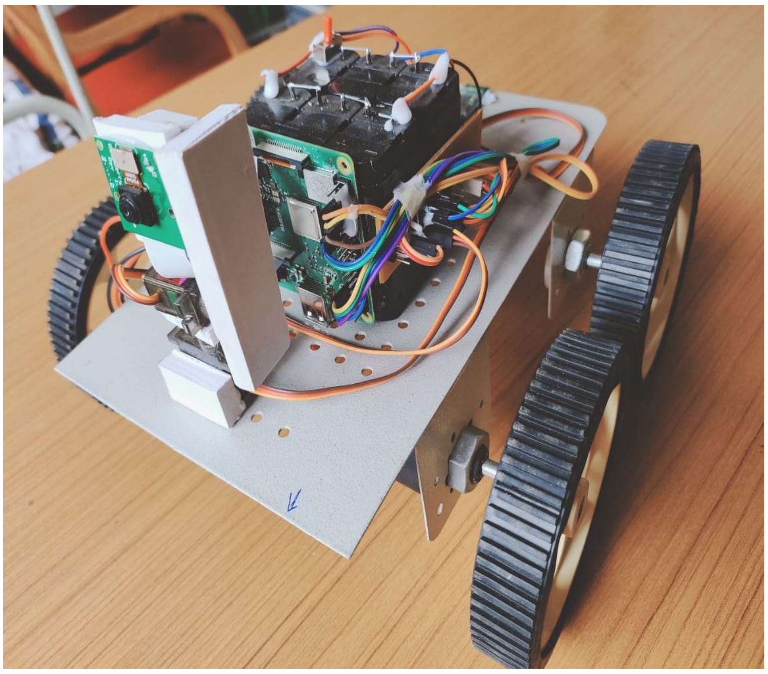


Fig 8.3: View of camera and raspberry pi modules

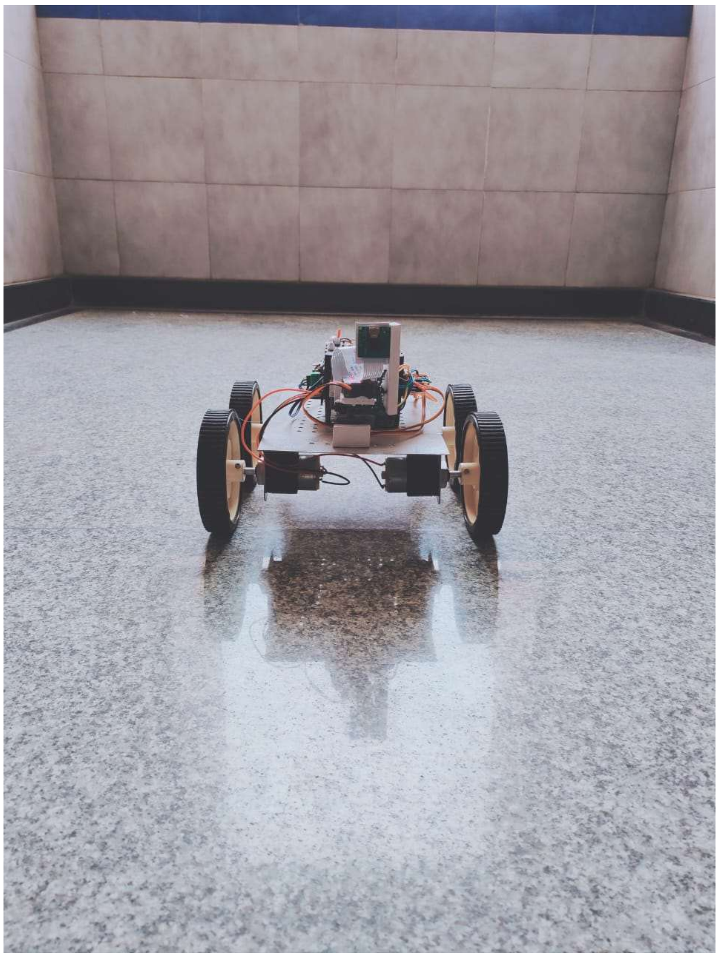


Fig 8.4: Picture captured during the movement of robot

Fig 8.5: VR Headset

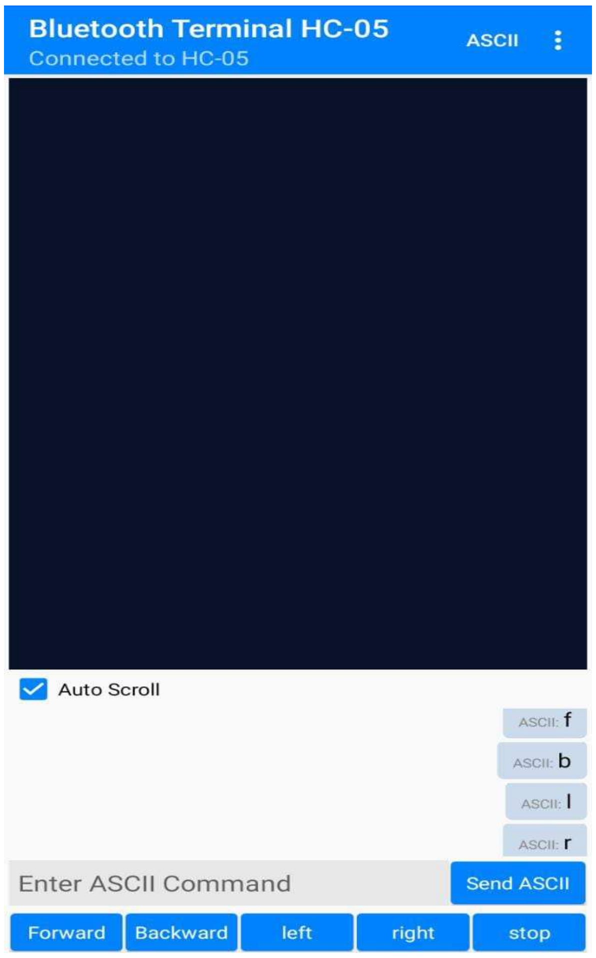
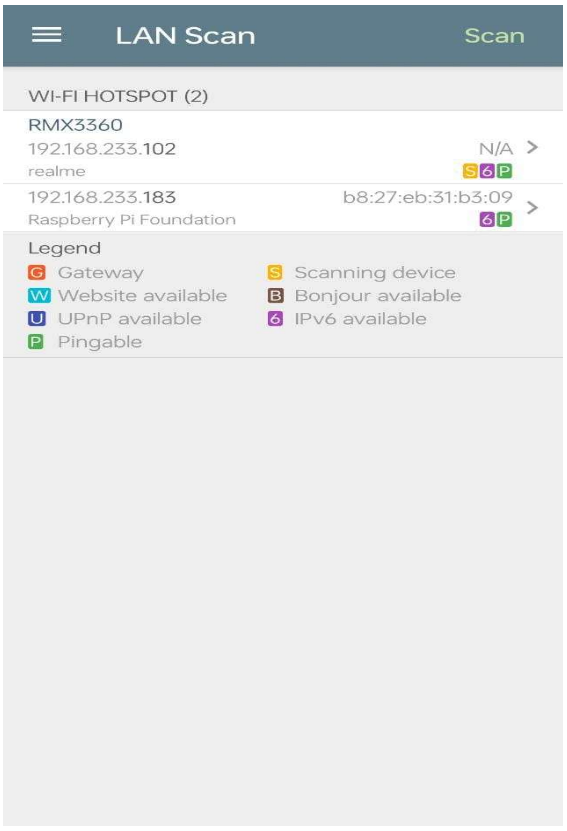


Fig 8.6: Bluetooth Terminal HC – 05 Application



LAN Scan

Scan

WI-FI HOTSPOT (2)

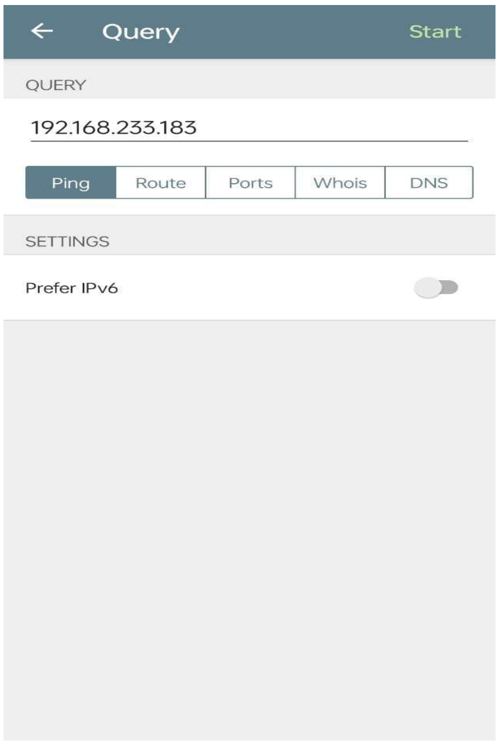
RMX3360
192.168.233.102 N/A
realme S 6 P

192.168.233.183 b8:27:eb:31:b3:09
Raspberry Pi Foundation 6 P

Legend

G Gateway	S Scanning device
W Website available	B Bonjour available
U UPnP available	6 IPv6 available
P Pingable	

Fig 8.7: Network analyzer applicaton



Query

Start

QUERY

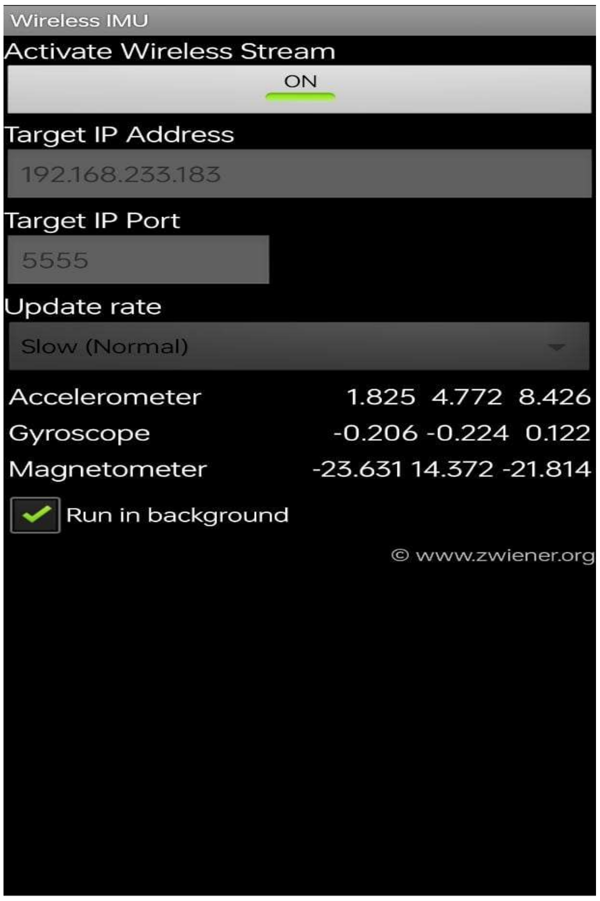
192.168.233.183

Ping | Route | Ports | Whois | DNS

SETTINGS

Prefer IPv6

Fig 8.8: View of IP address in network analyzer



Wireless IMU

Activate Wireless Stream

ON

Target IP Address

192.168.233.183

Target IP Port

5555

Update rate

Slow (Normal)

```
Accelerometer        1.825  4.772  8.426
Gyroscope           -0.206 -0.224  0.122
Magnetometer       -23.631 14.372 -21.814
```

✓ Run in background

© www.zwiener.org

Fig 8.9: Wireless IMU application

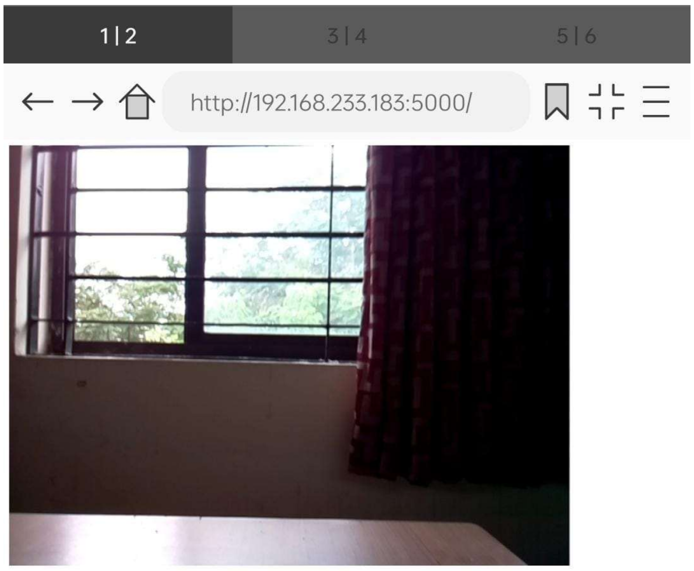


Fig 8.10: Visuals of camera in dual browser applcation

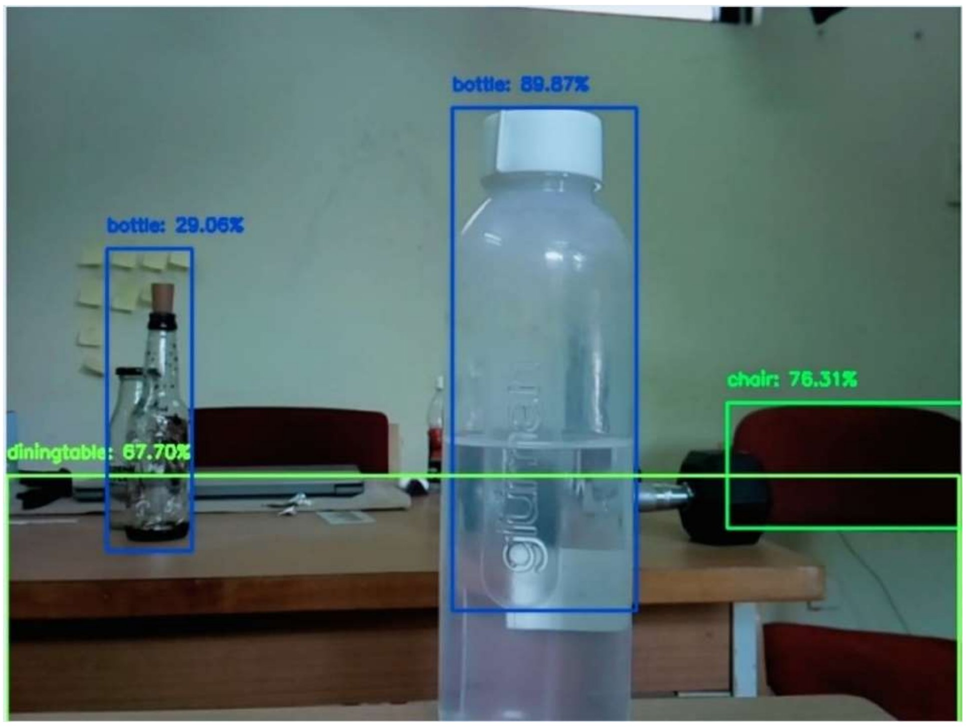


Fig 8.11: Object detection view

Chapter 9

CONCLUSION

9.1 Conclusion

Integrating features of all the hardware components used have been developed in it. Presence of every module has been reasoned out and placed carefully, thus contributing to the best working of the unit. Secondly, using highly advanced IC's with the help of growing technology, the project has been successfully implemented. Thus the project has been successfully designed and tested.

9.2 Future Enhancement

Our project "Virtual Telepresence Robot Using Raspberry Pi" is mainly intended to give direction to the robot using VR head set

The controlling device of the whole system is a Raspberry Pi processor. Arduino, Pi Camera, Servomotor, Bluetooth module are interfaced to the ARM CORTEX A5 1.2 Ghz processor Raspberry Pi. The data received by the fingerprint scanner is fed to theARM CORTEX A5 1.2 Ghz processor. The processor acts accordingly and either opens or does not open the door. In achieving the task the controller is loaded with a program written using Embedded Linux programming language.

This project can be extended using high efficiency GSM module using which we can send the user about the unauthorized access. The GSM module gives the SMS messages of the authentication through SMS. Using GSM we can know which direction the camera is moved and we can get direction in the form of message.

REFERENCES

[1] Dhahran Jadhav, Parth Shah, Henil Shah, "A Study to design VI classrooms using Virtual Reality aided Telepresence", IEEE 18th International Conference on Advanced Learning Technologies, 2018, pp. 319-321.

[2] Keerthi Premkumar. K Gerard Joe Mr, Nigel, Wi-Fi Android Smart Phone Based Robotic Arm Control Using Raspberry Pi IEEE Sponsored 2nd International Conference on Innovations in Information Embedded and Communication System.

[3] Kwon, O.H., Koo, S.Y., Kim, Y.G. and Kwon, D.S., 2010, October. Telepresence robot system for English tutoring. In 2010 IEEE workshop on advanced robotics and its social impacts (pp. 152-155). IEEE.

[4] Documentation by https://www.raspberrypi.org

[5] Stack overflow website for programming concepts

[6] Github –Flask video streaming

[7] Hossain, Mohammad Tanzir Kabir, "A Real-time Surveillance Nazmul Mini-rover Based on OpenCV-Python J AVA Using Raspberry Pi 2", 2015 IEEE International Conference on Control System, Computing and Engineering, 27 - 29 November 2015, Penang, Malaysia

[8] Raj kamal –Microcontrollers Architecture, Programming, Interfacing and System Design. 238

[9] PCB Design Tutorial –David.L.Jones.

[10] www.allaboutcircuits.com

[11] https://www.microchip.com/

[12] www.howstuffworks.com

[13] http://openrelief.org/