**Chapter 1**

**INTRODUCTION**

* 1. **Virtual Telepresence**

The term "virtual telepresence" refers to the use of technology to simulate physical presence and permit interaction with people or settings while being physically distant. It enables people to interact and collaborate in real-time as if they were physically present while electronically present in another place.

Application areas for virtual telepresence include business, education, healthcare, and entertainment. It makes it possible to collaborate remotely, hold virtual meetings, learn remotely, receive medical advice remotely, and have immersive experiences.

People can overcome the restrictions of physical distance and engage in meaningful interactions and experiences with others, regardless of their location, by using virtual telepresence technologies.

**Virtual Reality (VR):** Using cutting-edge technology, immersive digital experiences are created that imitate physical presence in a digitally created environment. Users engage with virtual objects and other users while wearing VR headsets that offer a 360-degree view.

**Augmented Reality (AR):** Augmented reality is the overlaying of digital data or virtual things over the physical world. Users can view and interact with virtual objects through AR applications or devices while still remaining present in the real world.

* 1. **What is Virtual Telepresence Robot?**

With the help of a virtual telepresence robot, a user can view and interact with a remote environment as if they were there in person. The robot has a camera and microphone, and the user can use a computer or mobile device to control the robot's movement and vision.

**1.3 Existing Systems:**

The Double, a well-known telepresence robot, is offered by Double Robotics. It has an extensible pole and a self-balancing base that can handle an iPad or tablet. The robot can be moved around, controlled remotely, and used for video conferences.

Appropriate Technologies Beam: A motorised base, a sizable screen, and a camera make up the telepresence robot Beam. It enables users to engage via video conferencing, navigate in a faraway place, and control the robot remotely.

OhmniLabs: With an emphasis on usability and simplicity, OhmniLabs offers telepresence robots. Their robots have an array of cameras, microphones, and screens that allow for remote presence and communication.

A telepresence robot for corporations and organisations is available from Ava Robotics. Their robot enables remote users to move and engage with others in real-time by fusing mobility with a sizable display.

* 1. **Proposed System**

The suggested system attempts to develop a VR surveillance robot using the idea of telepresence robots that incorporate virtual reality. Our robot is unique in that its motion is controlled by the user's movement in the actual physical space rather than by using standard controllers. The Raspberry Pi, to which a camera module is linked, is the foundation of the VR design. The wireless transmission of the live video from the camera to an Android smartphone running the VR app follows. The user dons a VR headset and inserts their smartphone inside it.

The user's head is turned, for example, to the right or left, and the smart phone reads the accelerometer and magnetometer data. This information is transmitted through Wi-Fi to the modem and the Raspberry Pi board, which then use these values as inputs to the servo motors.

On the user's virtual reality (VR) headset, the image-processing-based obstacle identification is shown together with the acquired images. Through an app that is downloaded to the user's smartphone, the robot can also be moved in any direction (in which the user turns his head, for example, right or left).

The camera is moved by two servo motors, one for vertical movement and the other for horizontal movement. As a result, the Raspberry Pi camera will turn to the right when you swivel your head while wearing a VR headset. Input for navigation or movement of the robot camera is also provided via the smartphone over Wi-Fi. The geared motors and motor driver IC are connected at the navigation circuit's conclusion. The smartphone can use Bluetooth to send the robot's operating instructions. Bluetooth is utilised in this illustration.

**1.5 The main objectives of the project:**

1. To create a robot for telepresence online.
2. Performing the required action on a Raspberry Pi.
3. Using Wi-Fi to transfer data to the modem.
4. This robot has the ability to transmit live pictures and obstacle detection to a VR headset.

**1.6 Applications**

* The robot can be utilised in firefighting and rescue operations, as well as in the medical field to check on patients when a doctor is unable to make rounds.
* The robot provides a real-time visual experience, enabling users to realistically explore properties, as it captures visuals and categorises objects for use in archaeology surveys.

**Chapter 2**

**LITERATURE SURVEY**

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

Time is a valuable resource for all people today. We introduce robots into the world in an effort to remove obstacles and make life easier. In this project, we want to watch over a location that is far away from the user, thus we're introducing a virtual telepresence robot to cut down on journey time between his starting point and the desired monitoring location. Virtual reality and direction-controlled robots are two technologies that we can use to achieve our aspirations. Since a few years ago, virtual reality has been employed extensively in a variety of industries, including the chemical, graphics, and video gaming industries. Virtual reality has a particular effect that enables the user to feel the emotions necessary to perceive it as genuine.

Basically, there are several ways to control a robot. Here, we can use a remote control or the screen of our smartphone to give the robot instructions. As is well known, robots were created to lighten human labour and speed up processes. Robots are non-living objects that function in accordance with the data or programme that was originally installed or stored in them.

In our project, we use a Raspberry Pi, on which we programme and save the programme that controls the robot. This project offers the user the benefits of virtual reality, which provides the user the impression that they are physically there at the desired area without actually being there. They can also control the robot from a distance, and it may move in the direction the user instructs. Here, an RPI camera will be mounted on the robot to take pictures or videos and transmit the information back to the user for viewing on his phone or other device.

**Jingxin Zhang Department of Informatics, University of Hamburg, Germany**

## Humans should be able to walk around a remote area, interact with a remote object, or converse with a remote person via telepresence devices. However, the interaction and exploration of remote environments (REs) are typically not supported by existing telepresence systems in a natural way. The majority of modern telepresence technologies, in particular, use portable webcams with speakers and microphones. Because of this, using a single webcam to record RE gives viewers a relatively small field of view and a restricted sense of spatial presence. The presence of teleoperators is constrained by these two problems. Additionally, a high error rate for teleoperation can result from a lack of visual information about the RE. tasks and online communications. Additionally, in modern telepresence systems, the typical movement controls of mobile platforms are frequently limited to straightforward interaction tools like joysticks, touchpads, mouse, or keyboards. Operators cannot accomplish other jobs with their hands free since they are required to utilise them to control the mobile platform. The user experience as a whole, task performance, and naturalness may all suffer as a result.

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

## The work's initial phase focuses on four-wheeled robots that are built on the Arduino platform. The Arduino board will be connected to four geared motors, which drive the wheels. The Arduino board will also be connected to the Bluetooth module. Using a straightforward Bluetooth-enabled controller would be the first step in testing the robot. The second stage involves constructing the pedometer. Unwanted noise signals can be found in the accelerometer readings. It might negatively impact step counting. To start step counting, the noise signal must be appropriately filtered using the right filters.

## We are utilising Kalman filters for this. It is an ideal estimator that uses the most recent input data as well as the previous state estimate to estimate the unknown value. The foundation of this approach is minimising the mean square deviation. The following is how the Kalman filter works: Pc=P + varProcess, where Pc is the expected covariance, p is the covariance of the sensor data, and varProcess is the covariance of the process. Calculate the Kalman gain using the formula G=P/(Pc + varVolt), where the varVolt-variance and G-Kalman gain are obtained using Excel and reading samples of raw sensor data.

## Update the sensor output covariance: P=(1-G) \*Pc Calculate the sensor voltage Kalman estimate: Xe=G\*(netmag-Xp) + Xp, where netmag is the net magnitude of the sensor output along the x, y, and z axes, and Xe is the Xe-Kalman estimate of the sensor voltage.

## Netmag = x2 + y2 + z2 is the formula for calculating the net magnitude. After receiving the filtered output, we must choose a threshold value that will start step counting. To do this, different users were permitted to wear their smartphones in the band and were instructed to walk in various ways. Each user's accelerometer measurements are continuously tracked, and at the end a threshold value is established and entered into the programme code. Since many people participated in the threshold experiment, it has the benefit that even the smallest motions of the hands or legs will not cause step counting. The magnetometer readings from the user's smartphone are used to rotate the robot in accordance with their movements. The magnetic field of the Earth is used as the basis for all calculations. The robot is programmed to initially check the user's direction when it is powered on and adjust its course accordingly.

## Assume, for instance, that the user is standing facing the north-northeast when the robot is turned on and facing the north. The robot notices a directional mismatch and makes a 45-degree spin to the right to remedy it. It will keep moving in the same direction as the user after matching the directions. In order to determine whether the user has turned right or left after each movement, the robot compares its direction to that of the user and, if there is a mismatch, adjusts its direction accordingly.

## The pedometer is first verified to accurately count steps before being connected to the robot via Bluetooth and having the controller code changed. In the third stage, virtual reality is integrated using a Raspberry Pi board, which is coupled to the camera module. A set of servo motors rotates the camera module. The top of the robot that resembles the user's head has a camera mount attached with two servo motors that allow for both pan and tilt movement.

# The gyroscope values will change as the user turns their head after setting the initial data, and these values are utilised to determine how far the camera mount needs to be moved. In the last stage, a dual screen app is utilised to divide the screen so that users can run two apps simultaneously on their smartphone, which serves as the VR headset's display. In this step, the smartphone and Raspberry Pi are also linked, and the VR headgear is converted into a camera controller.

# **Application of redirected walking in room-scale VR**

## [Eike Langbehn](https://ieeexplore.ieee.org/author/37085793774), [Paul Lubos](https://ieeexplore.ieee.org/author/37085995102), [Gerd Bruder](https://ieeexplore.ieee.org/author/37304364100), [Frank Steinicke](https://ieeexplore.ieee.org/author/37304363900)

## By deftly adjusting the virtual camera, redirected walking (RDW) promises to enable nearly natural walking in an indefinitely vast virtual environment (VE). Previous research revealed that undetected RDW requires a physical radius of at least 22 metres. However, we discovered that this radius may be lowered and RDW applied to room-scale VR, i.e., up to around 5m 5m.

In order to achieve this, the Ve uses curved paths rather than straight ones and connects them in a way that allows for continuous walking. Additionally, the equivalent paths in the real world are set up in a way that seamlessly integrates with room scale virtual reality. In this research demo, participants can explore a VE of roughly 25m by 25m while experiencing RDW in a room-scale head-mounted display VR system.

## 2.1 Embedded Systems

## 2.1.1 Embedded Systems

A computer system called an embedded system is made to carry out one or a small number of specific tasks, frequently under real-time computing limitations. It is included into a whole device, frequently with physical and mechanical components. A general-purpose computer, such as a personal computer (PC), on the other hand, is made to be adaptable and to satisfy a variety of user needs. Today's commonplace devices are controlled by embedded systems.

One or more main processing cores—typically microcontrollers or digital signal processors (DSP)—control embedded systems. Being committed to doing a certain task, which may need for extremely powerful processors, is the crucial quality. For instance, even though they employ mainframe computers and specialised regional and national networks to connect airports and radar sites, air traffic control systems might be considered embedded. (Each radar most likely has a single or more embedded systems.)

Design engineers may maximise the embedded system's performance and reliability while reducing the product's size and cost because it is dedicated to certain functions. Embedded systems are sometimes mass-produced in order to take advantage of economies of scale.

**2.1.2 Real-time problems:**

Because the embedded system is intended to fulfil a specific task, design engineers can optimise it to decrease the product's size and cost while boosting its performance and dependability. Because of the efficiencies of scale, some embedded systems are mass-produced.

**2.1.3 Need for Embedded Systems:**

Because new devices are released daily that make inventive use of embedded computers, the applications for embedded systems are practically endless. Hardware like microprocessors, microcontrollers, and FPGA chips have gotten significantly less expensive recently. It is therefore wiser to simply purchase the generic chip and create your own unique software for it when creating a new form of control. It takes much longer and costs much more money to create a chip specifically designed to tackle a given task or group of tasks.

It's also common for embedded computers to ship with sizable libraries, making "writing your own software" a remarkably simple operation. A computer and an embedded system are very different from the perspective of implementation. Real-Time reaction is frequently demanded in embedded devices. The major characteristics that set embedded systems apart are their dependability and simplicity of debugging.

**2.1.4 Debugging:**

Depending on the facilities provided, embedded debugging can be done at several levels. They can be loosely divided into the following categories, going from the most basic to the most complex:

* Interactive resident debugging utilising Forth and Basic or another simple shell offered by the embedded operating system.
* External debugging via serial port output or logging to track operation using either a flash-based monitor or a debug server like the Remedy Debugger, which even supports heterogeneous multicore systems.
* An in-circuit debugger (ICD), a piece of hardware that interfaces with the CPU using the JTAG or Nexus standard. This enables external control of the microprocessor's functionality, but is often limited to the processor's specialised debugging capabilities.
* An in-circuit emulator substitutes a simulated microprocessor for the real one, giving complete control over the microprocessor's functionality.
* A comprehensive emulator offers a simulation of all facets of the hardware, enabling control over and modification of all of it as well as debugging on a standard PC.
* The programmer can typically load and run software through the tools, inspect the code that is now running in the processor, and start or stop its operation, unless they are confined to external debugging. The code may be viewed as source code or as assembly code.
* Examine error logs and codes: To learn more about the issue, examine the system error codes, logs, and any available output. Pay close attention to any cautions, error messages, or other system feedback.

**2.2 Applications of Embedded Systems**

**Consumer applications:**

We utilise a variety of embedded systems at home, including the microwave, remote control, vcd and dvd players, camera, etc.

**Office Automation :**

We use systems like fax machines, modems, printers, etc.

**Industrial automation:**

Many industries use embedded systems nowadays for process control. In industries, embedded systems are created to carry out a specific task, such as monitoring temperature, pressure, humidity, voltage, current, etc. Based on these levels, additional devices are controlled, and data can be sent to a centralised monitoring station.

 Robots that have been programmed to perform a certain task can be used in businesses that require high levels of human presence to be avoided.

**Fig 2.1: Robot**

**Tele communications:**

Web cams, mobile phones, etc.

**Security:**

Security systems like fire alarms and burglar alarms use embedded systems.

**Aerospace:**

Both spaceships and aircraft use embedded systems.

**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

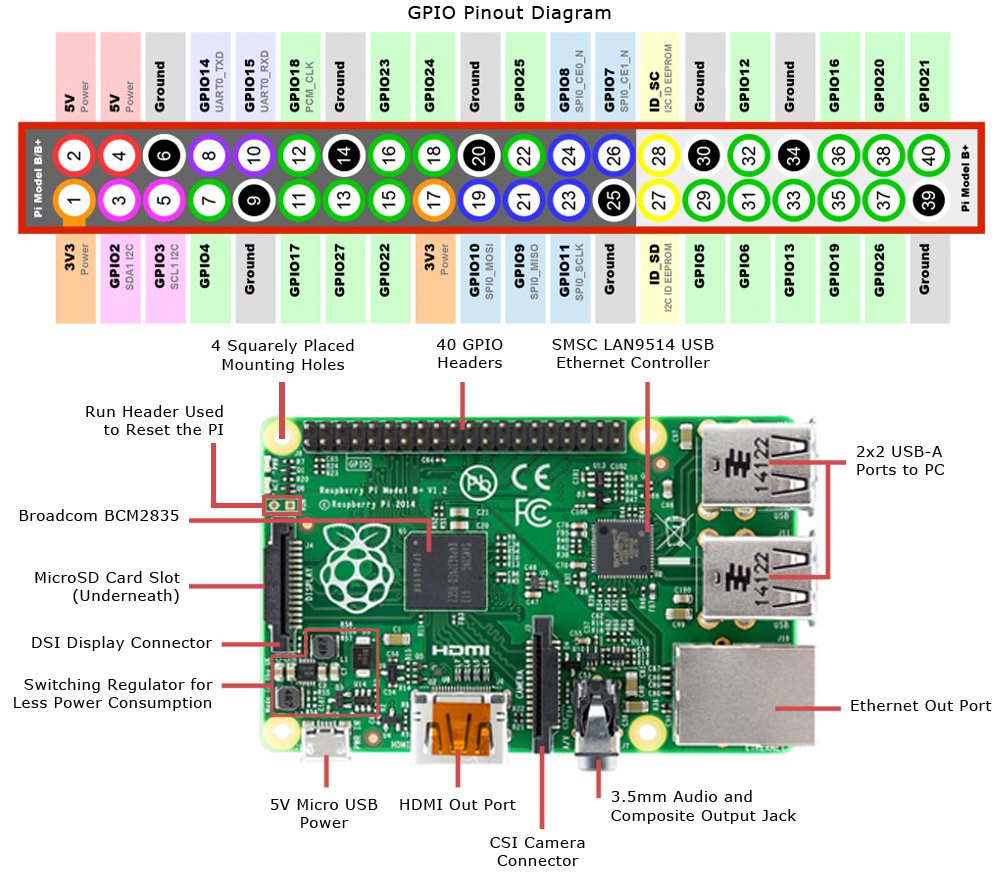
**3.3 Hardware Description**

**3.3.1 About Raspberry pi**

The Raspberry Pi is a single-board computer the size of a credit card that was created in the UK by the Raspberry Pi Foundation to support the teaching of fundamental computer science in schools. Through licenced manufacturing agreements with Newark element14 (Premier Farnell), RS Components, and Egoman, the Raspberry Pi is produced.

The most recent item in the Raspberry Pi 3 line is the Model A+. It features a 64-bit four core processor running at 1.4 GHz, dual-band 2.4 GHz and 5 GHz wireless LAN, and Bluetooth 4.2/BLE, just like the Raspberry Pi 3 Model B+.

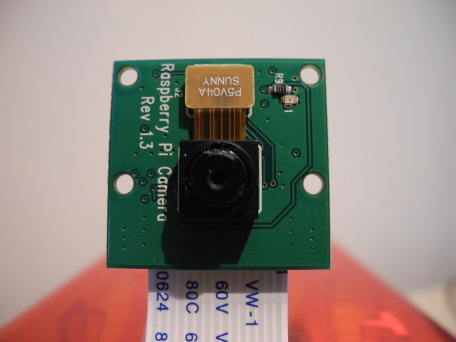
**Pin Description**



**Fig 3.1:** Raspberry Pi pin description

**3.3.2 Camera Module**

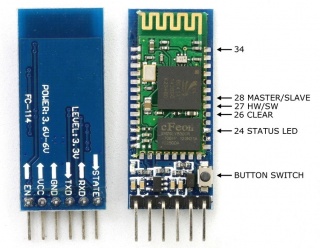
A thin (25mm by 20mm by 9mm) circuit board serves as the camera, and a flexible ribbon cable links it to the Raspberry Pi's Camera Serial Interface (CSI) bus socket. The camera's image sensor has a fixed focus lens and a native resolution of five megapixels. The camera's software supports video resolutions of 1080p30, 720p60, and 640x480p60/90 as well as full resolution still photos up to 2592x1944. Here is a picture of the camera module:



**Fig 3.2: Raspberry Pi Camera**

**3.3.3 Bluetooth module HC – 05**

A MASTER/SLAVE module is what the Bluetooth HC-05 module is. The factory setting is SLAVE by default. Only AT COMMANDS have the ability to configure the module's Role (Master or Slave). Although they can accept connections, the slave modules are unable to establish a connection with another Bluetooth device. The master module has the ability to connect to other components. It can be used as a simple serial port substitute to link an MCU to a GPS, a PC to an embedded project, etc. Simply read the datasheet for additional information.

[](https://wiki.eprolabs.com/index.php?title=File:FC-114.jpg)

**Fig 3.3:** HC – 05 Bluetooth

**3.3.4 Servo Motors**

An angular location, velocity, and acceleration can be precisely controlled using a servomotor, which is a rotary actuator. It consists of an appropriate motor connected to a position feedback sensor. It also needs a rather sophisticated controller, frequently a special module created just for use with servomotors.

Although the word "servomotor" is frequently used to describe a motor appropriate for use in a closed-loop control system, servomotors are not a particular sort of motor.

Applications for servomotors include robotics, CNC equipment, and automated manufacturing.

**3.3.5 DC Motor**

A dc motor converts electrical energy into mechanical energy by interacting magnetic fields with current-carrying conductors, which is normally how it works. the opposite process, using an alternator, generator, or dynamo to convert electrical energy to mechanical energy. The use of electric motors as generators and vice versa is common. A DC motor receives current and voltage as inputs, and produces torque (speed) as an output.

**3.4 Software Description**

**3.4.1 PYTHON**

The ideal programming language to utilise in this project to interact with the Raspberry Pi user is Python. Python is a multipurpose, interactive, object-oriented, and high-level programming language with an interpreter. Between 1985 and 1990, Guido van Rossum designed it. Python's source code is likewise accessible under the GNU General Public Licence (GPL), just like Perl. A popular high-level language for general-purpose programming is Python. Before executing, Python programmes don't need to be compiled. To use them, though, a computer must have the Python interpreter installed. A programme called the Python interpreter reads Python files and runs the code inside. The servo motors on the RPi board are managed via the python\_camera.py programme.

**3.4.2 RASPBIAN OS**

An operating system for the Raspberry Pi based on Debian is called Raspberry Pi OS (formerly Raspbian). It has been the primary operating system for the Raspberry Pi series of small single-board computers since 2015, according to the Raspberry Pi Foundation. As a stand-alone undertaking, Mike Thompson and Peter Green developed the first iteration of Raspbian. The preliminary construction was finished in June 2012. The Raspberry Pi range of small single-board computers with ARM CPUs is designed with Raspberry Pi OS in mind. All Raspberry Pi models save the Pico microcontroller support it. The desktop environment for Raspberry Pi OS is a customised version of LXDE with the Open box stacking window manager and a distinctive look.

**3.4.3 Linux Operating System**

An operating system for computers that is largely POSIX compatible and Unix-like called Linux was created using the model of free and open source software development. The Linux kernel, an operating system kernel that Linus Torvalds first released on October 5, 1991, is what makes Linux unique. GNU/Linux is used by the Free Software Foundation, which has generated considerable debate.

In order to standardise the software system structure, including the file system hierarchy used in the GNU/Linux operating system, numerous Linux distributions have collaborated on the Linux Standard Base (LSB). The Single UNIX Specification, the POSIX specification, and several other open standards serve as the foundation for the LSB, which adds to them in some areas.

**3.4.4 Bluetooth Terminal HC – 05 Application**

The Android app Bluetooth Terminal HC-05 offers a practical and approachable user interface for interfacing with the HC-05 Bluetooth module.

Bluetooth Connection: Users can connect to the HC-05 module by searching for nearby Bluetooth devices using the app. It offers a list of nearby Bluetooth devices and, if necessary, makes pairing easier.

Users can enter commands and get responses from the HC-05 module through the app's main interface, which is modelled after a terminal or console window. It offers a text input box for commands and a display area for the information the module has sent.

Text Input and Output: Using the app's terminal interface, users can type commands or messages to the HC-05 module. Both text-based data and special characters can be sent using it. Users can view the answers, sensor readings, and any other information sent by the module by viewing the data that has been received from it and is presented in the terminal interface.

**Chapter 4**

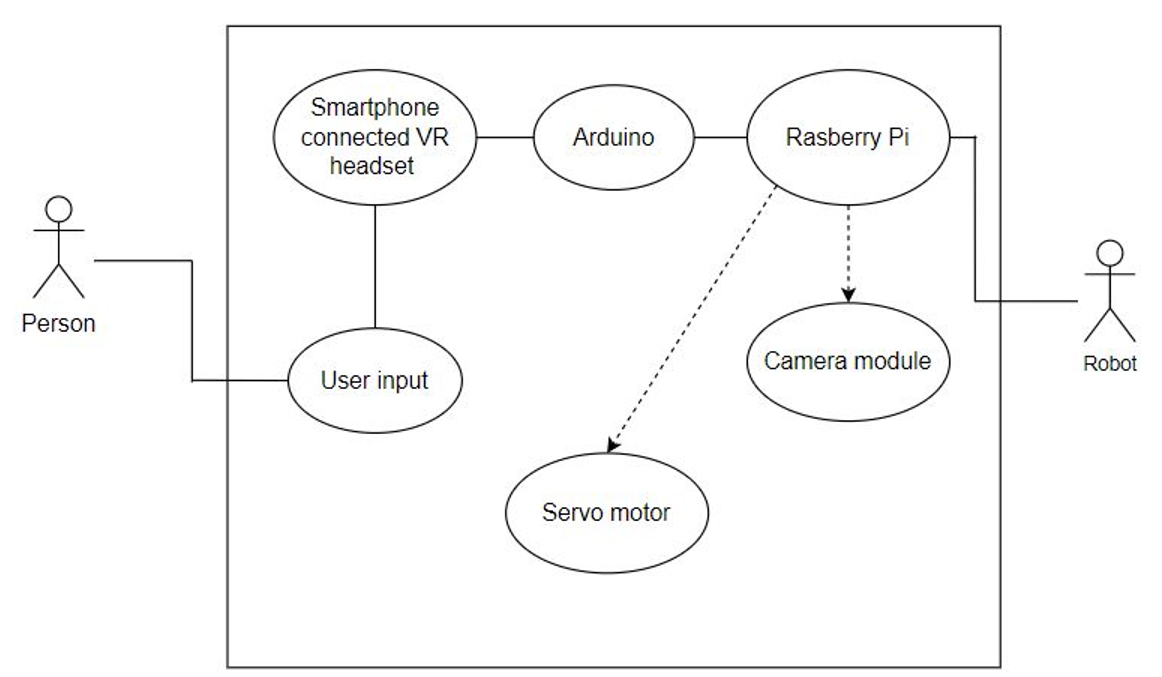
**SYSTEM ANALYSIS**

**4.1 System Analysis**

The process of researching and comprehending complicated systems in order to pinpoint their constituent parts, functions, relationships, and behaviour is referred to as system analysis. In order to obtain understanding of a system's operation and make educated judgements about its design, improvement, or troubleshooting, it entails a systematic and in-depth investigation of the system's structure and processes.

Gaining a thorough understanding of the system's current status, identifying areas for development, and suggesting solutions to satisfy the needs of the stakeholders are the objectives of system analysis. In order to ensure that the system is in line with user requirements and successfully supports the required functionality and objectives, it serves as the foundation for the system design, development, and implementation phases.

**4.2 Use Case Diagram**



**Fig 4.1:** Use case diagram

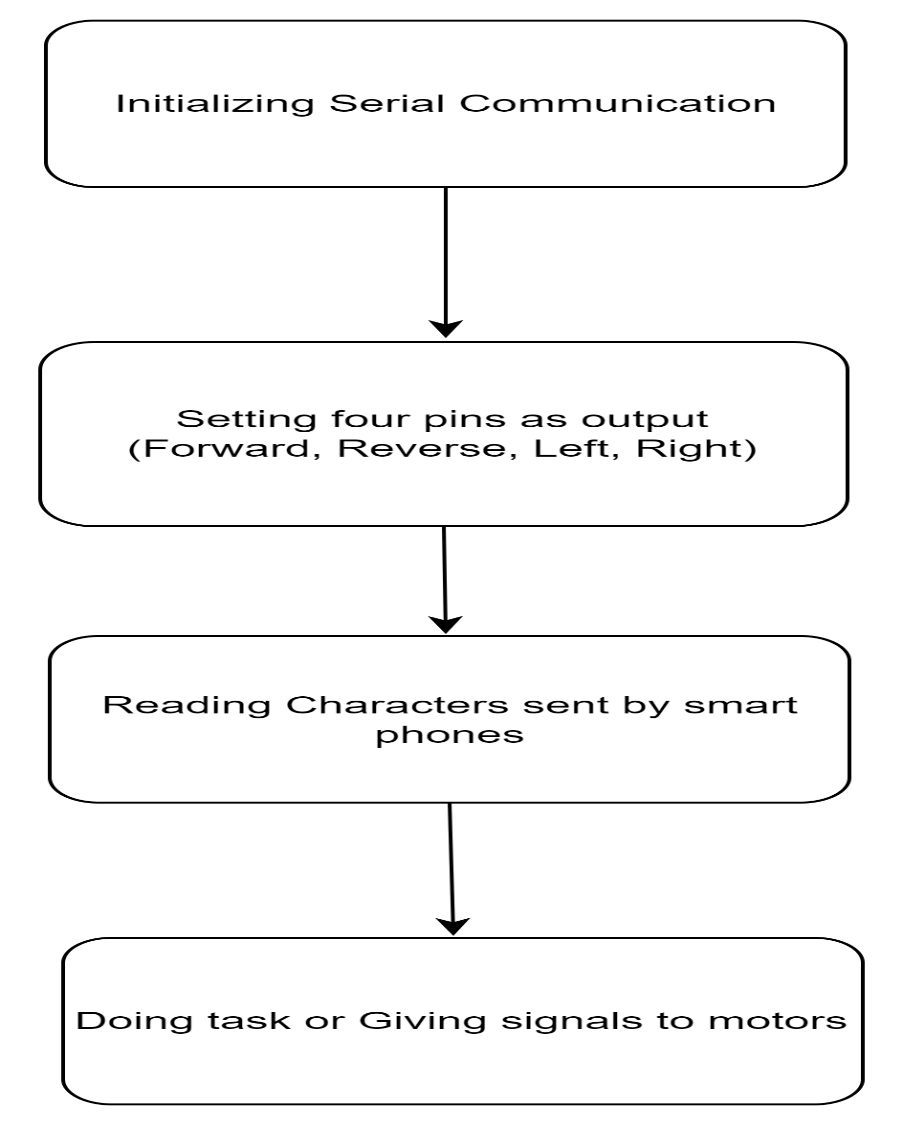
A user can start a virtual presence, control robot motions, observe a remote environment, connect with a remote user, change the camera view, and end a virtual presence. In order to improve their vision and viewpoint of the remote world, the user in this use case modifies the camera angle or view.

The virtual telepresence robot system's interactions and functionalities are outlined in the use case diagram. The user is aided in their understanding of the system's capabilities and the ways in which various actors engage with it, assisting in the design and development process.

**4.3 Data flow diagram**

**4.3.1 Initiation of the process**

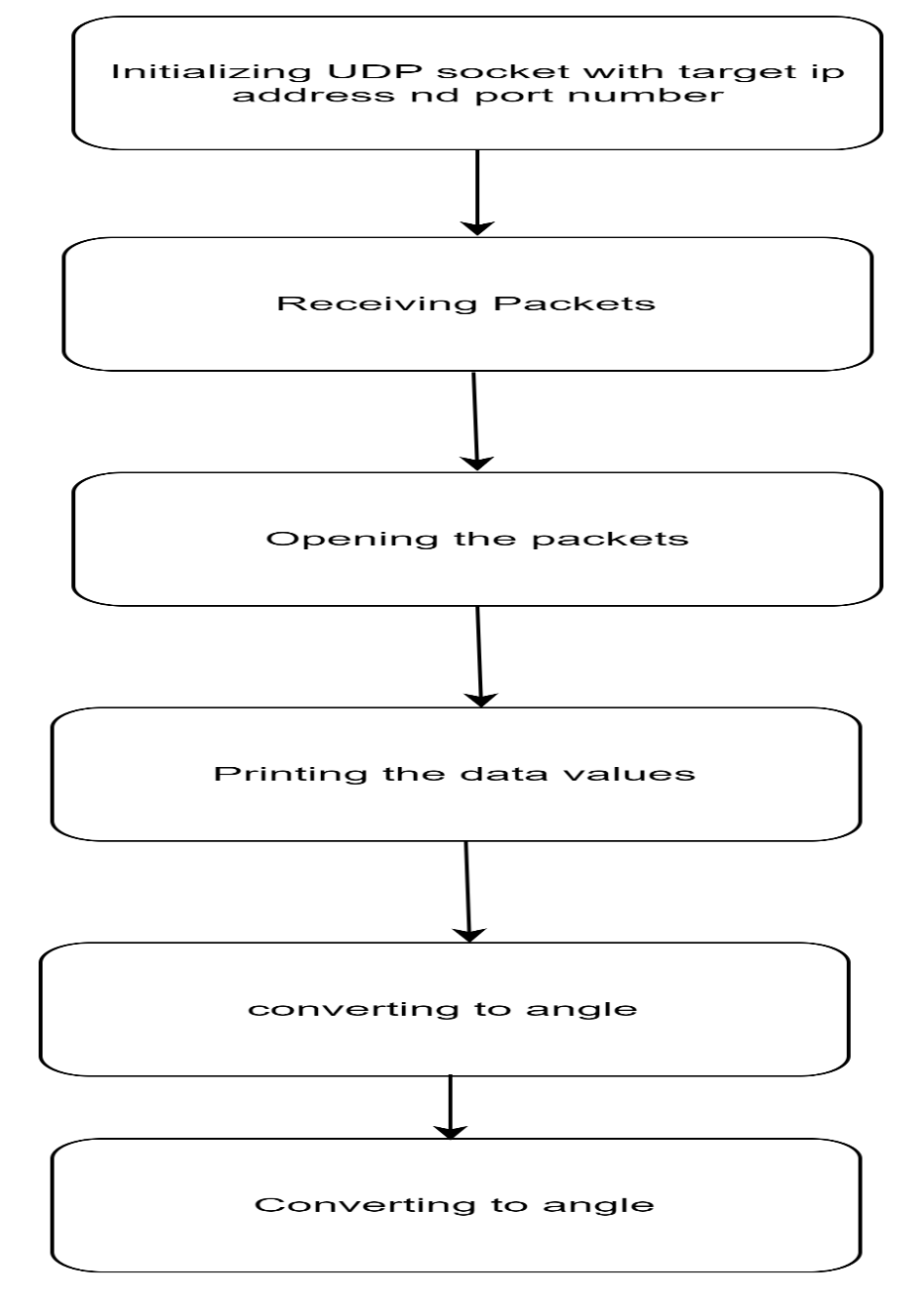
In order to start the user and robot interaction process, the user must first set up serial communication between the Bluetooth module and the Bluetooth Terminal HC-05 application. After that, the user must configure the application's pins in order to supply input to the DC motor that drives the robot. The Bluetooth module then reads the data that the user sends using the Bluetooth Terminal HC-05 application and sends signals to the robot in accordance with the user's input.



**Fig 4.2:** Initiation of process

**4.3. Capturing the live visuals**

The process will start with pairing with the target device and verification of the target IP address and port number. After that, the robot will receive data packets containing values for the camera angle and direction so that it can adjust its angle in response to user input and display images on the screen.



**Fig 4.3:** capturing live video

**4.4 Functional requirement**

1. To create a robot for telepresence online.

2. Performing the required action on a Raspberry Pi.

3. Using Wi-Fi to transfer data to the modem.

4. This robot has the ability to transmit live pictures and obstacle detection to a VR headset.

**4.5 Feasibility Study**

An evaluation of the viability and practicality of a proposed project or initiative is known as a feasibility study. To ascertain whether the project is technically, economically, and operationally feasible, it assesses a number of variables. The study attempts to offer decision-makers unbiased and trustworthy information to help them decide whether to move on with the project or consider other choices.

The viability and practicality of establishing such a system would be determined by a virtual telepresence robot feasibility study. The following main factors would be taken into account in a feasibility assessment for a virtual telepresence robot:

**1. Technical Feasibility:**

Evaluation of the virtual telepresence robot's technical requirements, including the robot's controls, communication, camera, and infrastructure systems.

Evaluation of the proposed technology's compatibility with current networks and systems.

A review of any technological issues or restrictions that might surface during the use of the virtual telepresence robot.

**2. User Acceptance and Social Feasibility:**

Evaluation of user adoption and acceptance of the virtual telepresence robot, taking into account aspects including user experience, usability, and prospective user advantages.

An evaluation of the robot's social influence and acceptance in its intended setting, taking into account societal norms, cultural aspects, and any potential user resistance.

**3. Operational Feasibility:**

evaluation of how the virtual telepresence robot will affect current processes and workflows within the company or setting in which it will be used.

Analysing the human resources, knowledge, and experience needed to maintain and operate the robot efficiently.

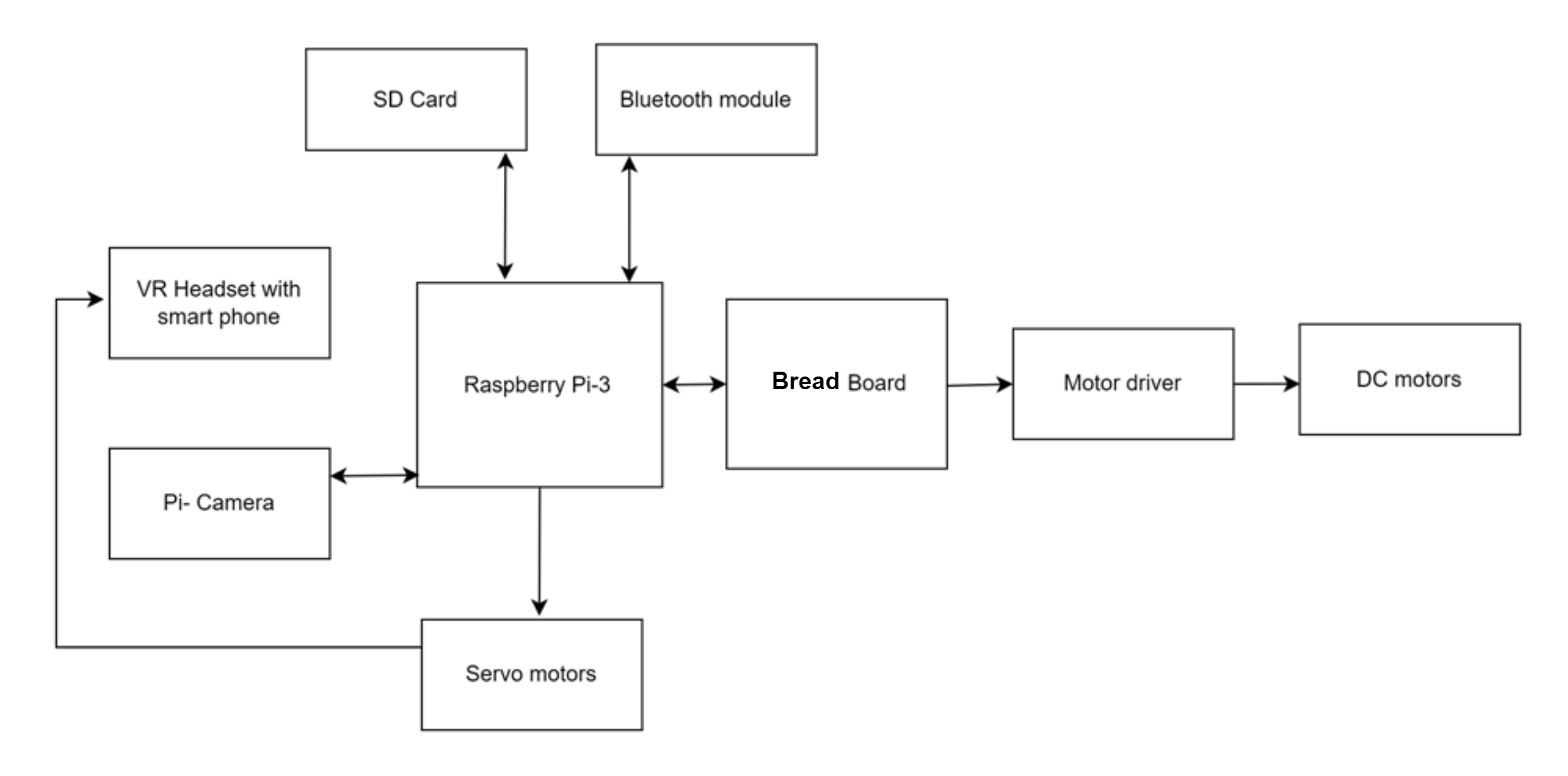
taking into account potential dangers, security issues, and security precautions as well as steps to ensure the seamless functioning and integration of the virtual telepresence robot into the current setup.

**Chapter 5**

**SYSTEM DESIGN**

Robotics, augmented reality, and virtual reality can work together to create cutting-edge solutions for various organisations. In this project, a robot with a camera is sent in a distant location to use a Raspberry Pi (RPi) to visually record the environment. On the user's virtual reality (VR) headset, the collected images are presented. The camera can now move in the direction of the user's head motions thanks to a new feature. If the user is physically there where the virtual tele-presence robot is positioned, this gives him a real-time experience. The user's smartphone can be used to control the virtual telepresence robot's movement in any direction.

**5.1 Architecture Diagram**

****

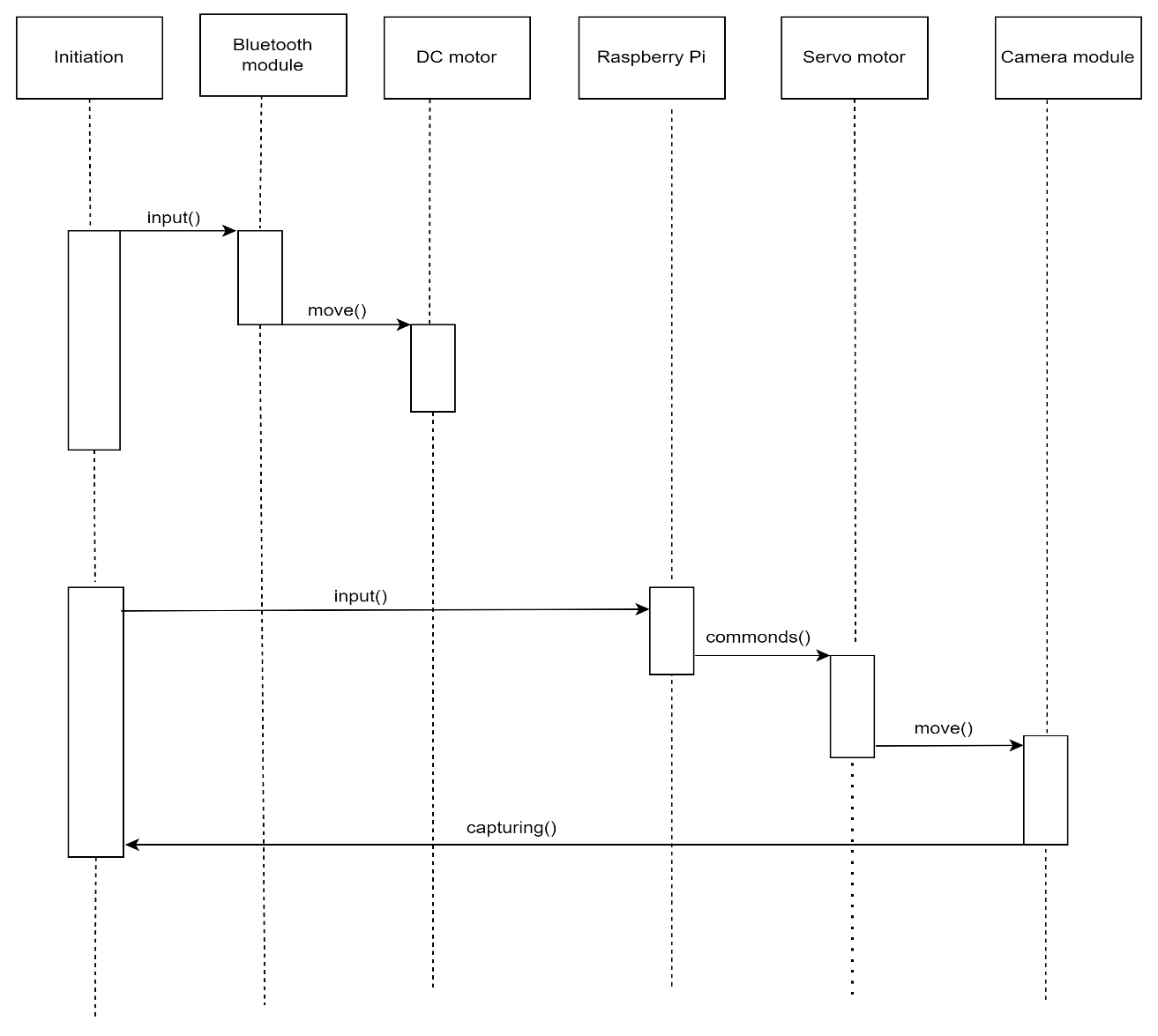
**Fig 5.1:** Architecture diagram

A four-wheeled robot serves as the foundation of the VR telepresence robot. To record live video of the surroundings in which it is operating, a camera mount with servo motors is fastened to the top of the robot. The user's smartphone is mounted in the VR headgear and plays this video. The robot's mobility in that specific environment is mapped to the user's motion.

The Virtual Interactive Reality Telepresence robot's entire system can be divided into VR, robotics, and user sections. A Raspberry Pi 3 Model A+ serves as the brain of the VR component. Using a 5 MP 1080p Raspberry Pi camera module, the video was captured. Two SG-90 servomotors can be used to move the camera module in both the pan and tilt directions. linked to the Raspberry Pi are motors. The user's head movement and this camera movement are in sync. The robotics portion is essentially a robot with four wheels. The 4 wheels are driven by 4 geared motors. An L293D motor driver shield that is powered by the battery drives the motors.

Using information sent over the HC-05 Bluetooth module, the robot's movement is made to match the user's movement. The wearable VR headset holding an Android smartphone running the split screen software makes up the user section. The Raspberry Pi camera module's captured video is displayed on the Android phone. The Wireless IMU app on the smartphone sends the user head movement data to the Raspberry Pi over Wi-Fi. A smartphone with the dabble application installed is another component of the user section. By adjusting the sensor values on the phone, steps are counted. Then, using Bluetooth, this "step data" is wirelessly communicated to the.

**5.2 Sequence Diagram**

****

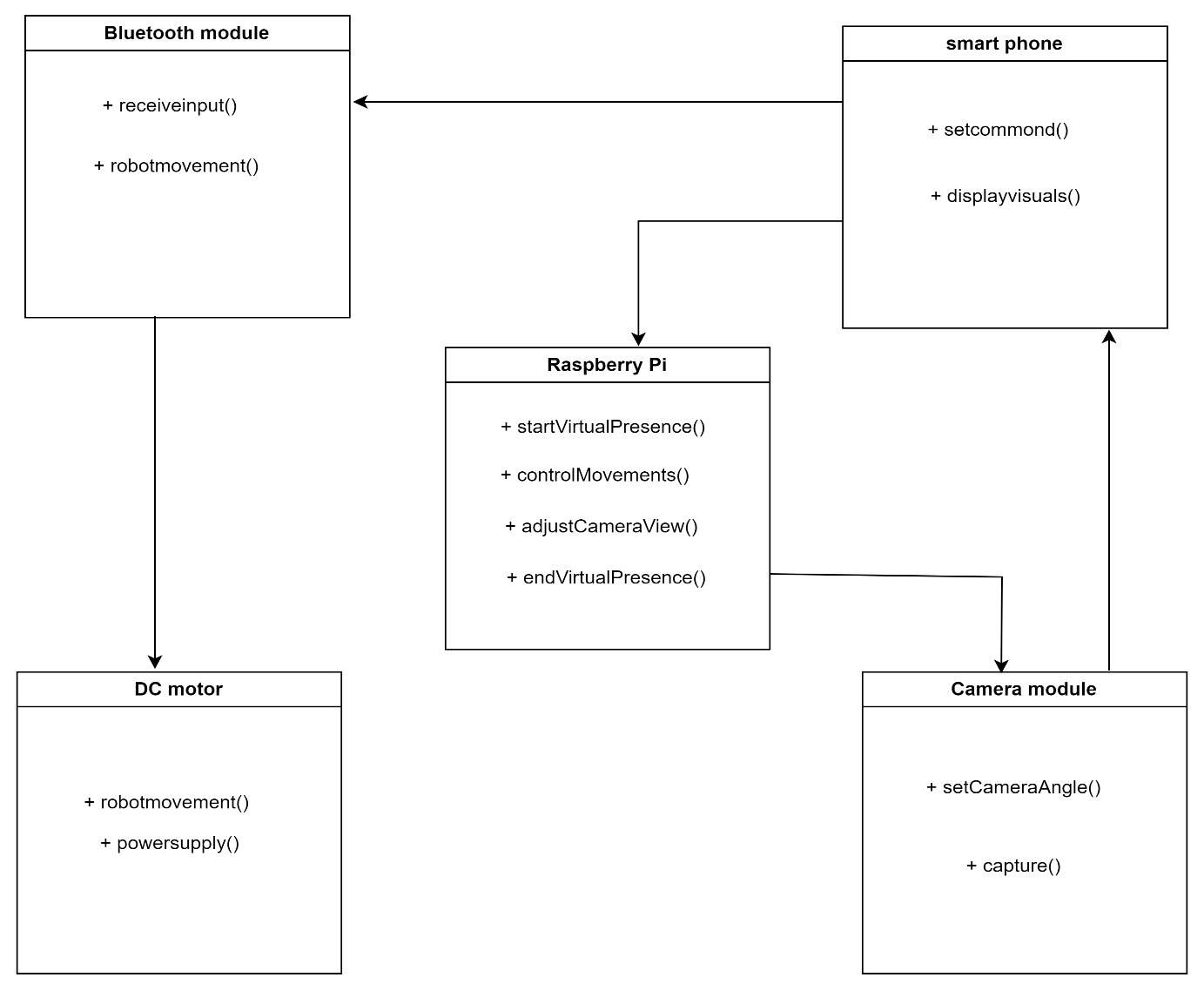
**Fig 5.2**: Sequence diagram

The User starts the virtual presence mode in this flowchart. The Virtual Telepresence Robot can be moved by the User, and it changes its position after processing the movement commands. The manipulator on the robot can be used by the User to interact with things. The camera system is turned on by the virtual telepresence robot. It offers a real-time video feed.

The User can also communicate with distant locations. This entails two-way communication between a remote location and the user. The Virtual Telepresence Robot may alter the camera angle and offer an updated video feed in response to the User's changes to the camera view. Finally, the Virtual Telepresence Robot turns off the camera system and cuts off the connection when the User wishes to discontinue the virtual presence.

This flow chart illustrates the interactions and communications that occur sequentially between the user, the virtual telepresence robot, and various system components at various times when utilising a virtual telepresence robot.

**5.3 Class Diagram**



**Fig 5.3**: class diagram

The Raspberry Pi is the primary class in this class diagram because it performs the majority of the operations. StartVirtualPresence(), controlMovements(), adjustCameraView(), and endVirtualPresence() are among the methods offered by the class.

## The Camera module class, which represents the robot's camera system, provides methods like setcameraangle() and capture() for changing the camera view.

## The Bluetooth module class uses the functions receiveinput() and robotmovement() to receive input from a user's smartphone and to send commands for the robot to move.

## The smartphone class will take visuals from the camera module and display them on the screen using displayVisuals(). It will also use setcommand() to transmit commands to the robot.

## The robotmovement() and powersupply() methods of the DC motor class will be used to control the robot's motions and to draw power from the batteries, respectively.

## 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.

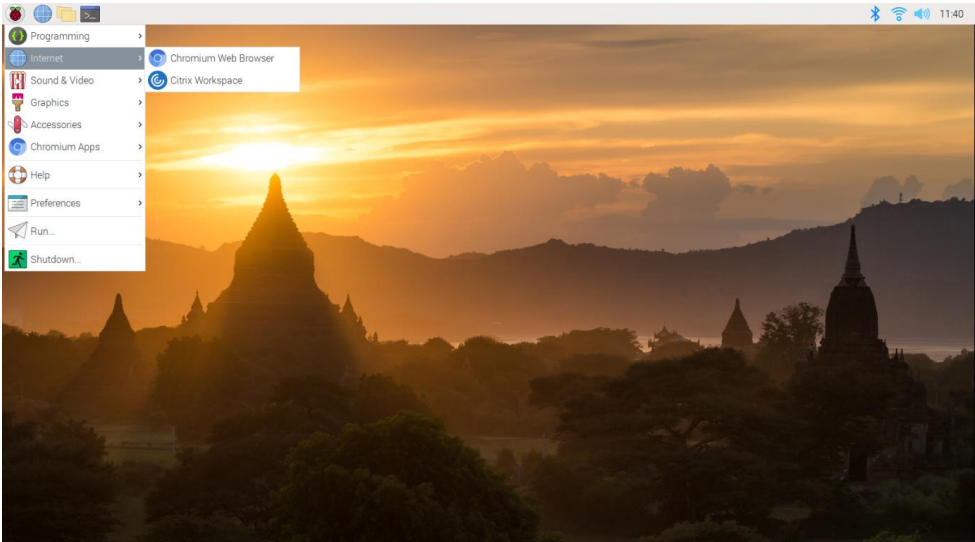
**Control Interface:**

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



**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

## 

## Fig 8.7: Network analyzer applicaton

## 

## Fig 8.8: View of IP address in network analyzer

## 

## 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](http://en.wikipedia.org/wiki/ARM11) CORTEX A5 1.2 Ghz processor Raspberry Pi. The data received by the fingerprint scanner is fed to the[ARM](http://en.wikipedia.org/wiki/ARM11) 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](http://www.allaboutcircuits.com)

## [11] <https://www.microchip.com/>

## [12] [www.howstuffworks.com](http://www.howstuffworks.com)

## [13] <http://openrelief.org/>