Remotely control Robot & Video Streaming over same Network using Raspberry pi

Harsh Chotaliya
EC Department,
Nirma University,
20bec039@nirmauni.ac.in

Dixit Dudhat
EC Department,
Nirma University,
20bec033@nirmauni.ac.in

***Abstract*—This project aims to develop a remotely controlled robot that can be controlled over a network using a Raspberry Pi. The Raspberry Pi is used to stream live video from the robot's camera to a remote user, who can control the robot's movements through a web-based interface. The project combines robotics, computer networking, and video streaming technologies to enable a seamless remote-control experience. The abstract summarizes the key aspects of the project, including the use of Raspberry Pi, the remote-control interface, and the live video streaming capabilities.**

I. INTRODUCTION

In recent years, the use of robotics has become increasingly prevalent in various fields, including healthcare, manufacturing, and education. With the advent of powerful minicomputers such as the Raspberry Pi, it has become easier to control and program robots for various applications. One of the challenges in robotics is the ability to remotely control robots over a network, which can be useful in scenarios where a human operator cannot be physically present with the robot.

In this project, we aim to develop a remotely controlled robot that can be controlled over a network using a Raspberry Pi. The robot will be equipped with a camera that can stream live video, enabling the user to remotely view the robot's surroundings. The user will also be able to control the robot's movements using a web-based interface.

The use of Raspberry Pi enables us to develop a cost-effective solution that can be easily replicated for various applications. The web-based interface makes it easy to control the robot using a smartphone, tablet, or computer, eliminating the need for specialized hardware or software. Live video streaming capabilities provide a real-time view of the robot's surroundings, enabling the user to make informed decisions about the robot's movements.

Overall, this project combines robotics, computer networking, and video streaming technologies to enable a seamless remote-control experience. The next sections will discuss the hardware and software requirements for this project, followed by the detailed steps for implementing the remote-control interface and live video streaming capabilities.

A. Raspberry pi working

In this project, Raspberry Pi is used as the central computing platform for controlling the robot and streaming live video over a network. The Raspberry Pi is a powerful, low-cost, credit-card-sized computer that can be used for a wide range of applications, including robotics.

The Raspberry Pi is used to control the robot's movements through GPIO (General Purpose Input Output) pins, which can be programmed to send signals to the robot's motors. The Raspberry Pi also connects to the robot's camera, which streams live video to the user over a network.

To enable remote control of the robot, a web-based interface is developed using Flask, a Python web framework. The interface allows the user to send commands to the Raspberry Pi, which then sends signals to the robot's motors to move the robot. The web interface can be accessed from any device with a web browser, including smartphones, tablets, and computers.

In addition to the remote-control interface, the Raspberry Pi also streams live video from the robot's camera over a network using the Raspberry Pi Camera Module. The camera module connects to the Raspberry Pi's CSI (Camera Serial Interface) port, which allows the Raspberry Pi to capture high-quality images and video from the robot's camera. The video stream is then broadcast over a network using the RTSP (Real-Time Streaming Protocol) protocol.

Overall, Raspberry Pi plays a central role in this project by providing a low-cost, powerful computing platform for controlling the robot and streaming live video over a network.

we are using a USB camera to capture live video from the robot's surroundings. The USB camera is connected to the Raspberry Pi, which processes the video stream and broadcasts it over a network using the RTSP (Real-Time Streaming Protocol) protocol.

To enable remote control of the robot, a web-based interface is developed using Flask, a Python web framework. The interface allows the user to send commands to the Raspberry Pi, which then sends signals to the robot's motors to move the robot.

The Raspberry Pi plays a central role in this project by providing a low-cost, powerful computing platform for controlling the robot and streaming live video over a network. The use of a USB camera instead of a Raspberry Pi Camera Module provides greater flexibility in selecting a camera with desired features, such as higher resolution or better low-light performance.

Overall, this project combines robotics, computer networking, and video streaming technologies to enable a seamless remote-control experience using a USB camera and Raspberry Pi.

To control the GPIO (General Purpose Input/Output) pins of a Raspberry Pi via a web page, we used a Python web framework like Flask or Django. we wrote a Python script that runs on the Raspberry Pi and listens for requests from the web page. When the user interacts with the web page, such as by clicking a button, the web page sends an HTTP request to the Python script running on the Raspberry Pi.

The Python script can then use a library like RPi. GPIO to control the GPIO pins on the Raspberry Pi. For example, we wrote code to turn an LED on or off by setting the corresponding GPIO pin to high or low. The Python script then sends a response back to the web page to confirm that the action has been taken.

To make the web page accessible from other devices on the same network, we have to run the Python script on the Raspberry Pi and host the web page on a web server. One popular option is to use Apache or Nginx as the web server and run the Python script using WSGI (Web Server Gateway Interface).

Overall, the Raspberry Pi plays a crucial role in this project by acting as both the web server and the hardware controller. By using Python and the appropriate libraries, you can create a responsive and intuitive web page for controlling the GPIO pins on the Raspberry Pi.

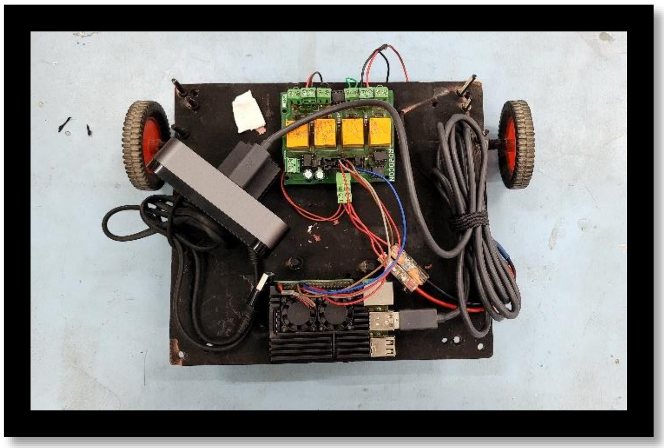
III. Differential Drive

In this project, we are using a differential drive system in the robot, which allows for precise control over the robot's movements. The differential drive system consists of two motors that are connected to the wheels on either side of the robot. By varying the speed and direction of each motor, the robot can move forward, backward, turn, and rotate in place.

To control the motors, we are using relays that are connected to the Raspberry Pi's GPIO pins. The relays allow the Raspberry Pi to send signals to the motors, which in turn control the robot's movements. By varying the signals sent to the motors, the Raspberry Pi can control the speed and direction of the robot's movements.

The use of relays to control the motors provides a simple and reliable way to interface with the robot's electrical system. The relays can be easily controlled using the Raspberry Pi's GPIO pins, which are well-suited for interfacing with external devices.

Overall, the combination of a differential drive system and relay control provides a powerful and flexible control system for the robot. By varying the signals sent to the motors, the robot can be precisely controlled to navigate through complex environments and perform a variety of tasks.



A. *Abbreviations and Acronyms*

1. GPIO - General Purpose Input Output
2. RTSP - Real-Time Streaming Protocol
3. CSI - Camera Serial Interface
4. USB - Universal Serial Bus
5. Vl - Velocity of the left wheel
6. Vr - Velocity of the right wheel

7. V - Velocity of the robot
8. ωL - Angular velocity of the left wheel
9. ωR - Angular velocity of the right wheel
10. R - Radius of the wheels

B. Units

- Vl - m/s (meters per second) or cm/s (centimetres per second) or ft/s (feet per second)
- Vr - m/s (meters per second) or cm/s (centimetres per second) or ft/s (feet per second)
- V - m/s (meters per second) or cm/s (centimetres per second) or ft/s (feet per second)
- ωL - rad/s (radians per second) or deg/s (degrees per second)
- ωR - rad/s (radians per second) or deg/s (degrees per second)
- R - m (meters) or cm (centimetres) or ft (feet)

C. Equations

The differential drive system in a robot is typically described using the following equations:

$$Vl = (2V - \omega L) / 2R$$

$$Vr = (2V - \omega R) / 2R$$

where Vl and Vr are the velocities of the left and right wheels, respectively, V is the velocity of the robot, ωL and ωR are the angular velocities of the left and right wheels, respectively, and R is the radius of the wheels.

These equations relate the velocities of the wheels to the overall velocity and angular velocity of the robot. By controlling the angular velocities of the wheels, the robot can be precisely controlled to move in different directions and turn in place. The use of these equations, along with the relay control system, provides a powerful and flexible control system for the robot.

D. Some Common Mistakes

One of the most common mistakes that people make when working on this project is not testing the hardware before starting the software development. It's important to ensure that all of the components, such as the robot, camera, and Raspberry Pi, are working properly before you begin writing code. Another common mistake is failing to document the code. When working on a complex project like this, it's crucial to document your code to make it easier to understand and maintain. By avoiding these mistakes, you can ensure that your project is successful and that you are able to achieve your goals.

IV. Conclusion

In conclusion, building a remotely controlled robot that streams video over the network using a Raspberry Pi is a challenging but rewarding project that combines hardware and software development. By using the right components, including a USB camera and a differential drive for the robot's motor control, and writing clean, well-documented code, you can create a system that is not only functional but also elegant and easy to use. With careful planning and attention to detail, you can successfully build a project that showcases your skills and creativity while providing a valuable tool for exploration and experimentation in robotics and computer vision.

Acknowledgment

I would like to express my gratitude to all those who have helped me throughout this project. Firstly, I would like to thank my supervisor, Prof. Vijay Savani, for providing me with guidance and support throughout the project. Their expertise and feedback were invaluable in helping me to develop a high-quality system. I would also like to thank Harsh Chotaliya, Dixit Dudhat who worked on the project, for their hard work and dedication. Their contributions were essential in making this project a success. Additionally, I would like to thank [Name of Friends or Family Members], who provided me with emotional support and encouragement throughout the project. Finally, I would like to thank Nirma University who provided me with the resources and facilities necessary to complete this project. Without their support, this project would not have been possible.

References

[1]"Raspberry Pi: A Quick-Start Guide" by Maik Schmidt, published by Pragmatic Bookshelf in 2016.

[2]"Programming Robots with ROS: A Practical Introduction to the Robot Operating System" by Morgan Quigley, Brian Gerkey, and William D. Smart, published by O'Reilly Media in 2015.

[3]"Learning Robotics Using Python" by Lentin Joseph, published by Packt Publishing in 2018.

[4]"Computer Vision: Algorithms and Applications" by Richard Szeliski, published by Springer in 2010.

[5] "Mastering ROS for Robotics Programming" by Lentin Joseph, published by Packet Publishing in 2018.

"GitHub Link of our Project" https://github.com/harshchotaliya7/Remotely-control-Robot-Video-Streaming-over-same-Network-using-Raspberry-pi.git