Autonomous Object Retrieval Robot

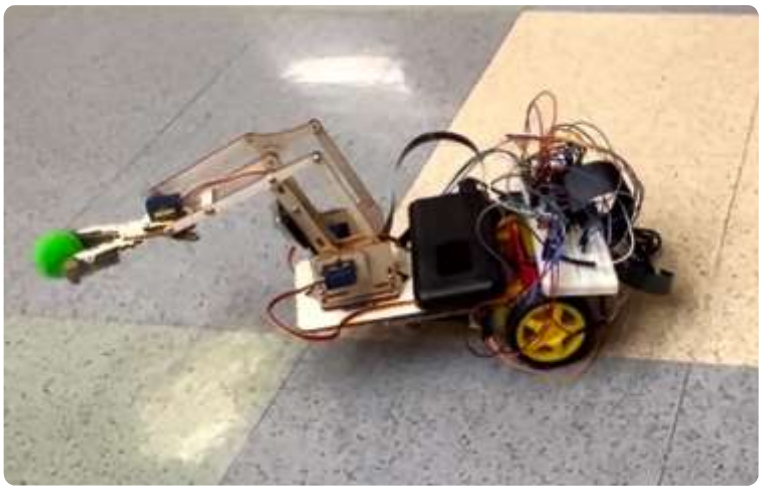
ECE5725 2024Fall Project

A Project By Chenxin Xun and Pengru Lung.

Demonstration Video

Introduction

This project aims to design and build an autonomous object retrieval robot capable of identifying and retrieving objects based on their color. The system utilizes a pair of Raspberry Pi devices, both equipped with cameras, and a robot with an arm featuring servos for object picking. The robot interacts with the user by receiving a color signal through a glove equipped with a camera, searching for the corresponding object, picking it up, and delivering it to the user.



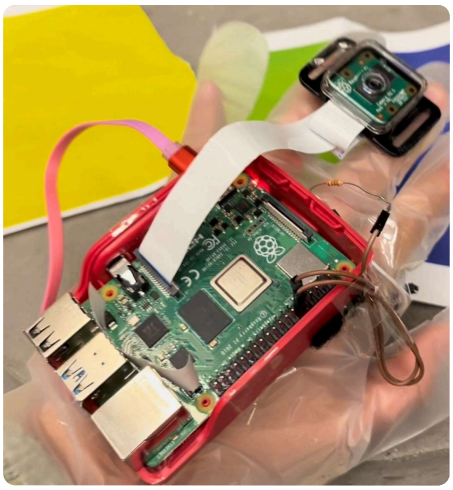
Project Objective:

- Design an autonomous robot capable of identifying objects based on their color.
- Implement a robotic arm with servos for precise object picking and placement.
- Utilize two Raspberry Pi devices, both equipped with cameras, for message transmission and system control.
- Enable user interaction via a camera-equipped glove to specify the desired object color.
- Ensure smooth and reliable object detection, retrieval, and delivery to the user.

Design

Hardware Design

Glove Design



We have mounted the Raspberry Pi 4 and a camera onto the glove. Also, we implemented a button on the Raspberry Pi 4 to control the camera. Using soldering, we connected the button and a 1kΩ resistor, wiring them to GPIO pin 17 and Ground. This setup was thoroughly tested to ensure both the hardware's stability and the responsiveness of the button control.

Robot Design

In our robot design, we utilize a Raspberry Pi 4 to control the robot's movements and arm. The robot is equipped with two servos for controlling the arm and two servos for robot movement. The servos are connected to specific GPIO pins on the Raspberry Pi, which are used to send control signals for precise movement. The Raspberry Pi acts as the central control unit, coordinating the servos' actions based on the input from the camera-equipped glove. Our GPIO pin mappings are shown in the table below.

Table 1: RaspberryPi GPIO Pin to Signal Mapping

Pi GPIO Pin	Signal Name
GPIO 4	Robot wrist rotation control
GPIO 5	Direction control for left motor
GPIO 6	Direction control for left motor
GPIO 12	Robot arm base rotation control
GPIO 13	Robot shoulder elevation control
GPIO 16	PWM signal for right motor
GPIO 19	PWM signal for left motor
GPIO 20	Direction control for right motor
GPIO 21	Direction control for right motor
GPIO 26	Robot elbow joint control

Software Design

Glove Design

The software design for the device is centered around detecting and identifying color-coded objects using computer vision techniques and sending the detected color to the robot. The program utilizes the OpenCV library (cv2) for real-time image processing, where the camera feed is continuously captured and analyzed to detect specific color cubes.

1. The Color Detection: The color detection is based on the HSV color space, with predefined ranges for different colors (Green, Blue, Yellow). The software first captures a frame from the camera, converts it to the HSV color space, and then applies a mask to isolate regions of the image matching the color ranges. Contours of these regions are found and used to calculate the center of the object.

2. System Integration: A button is used to trigger the color detection process, allowing the user to initiate the system when desired. The system utilizes Wi-Fi communication to transmit the detected color information. Once the color is detected, the software sends the color information via a socket connection over Wi-Fi to a remote server. This ensures the robot is informed of the object's color for further action in real-time.

Robot Design

1. Server: The robot's communication system uses Wi-Fi for seamless data exchange with the camera-equipped glove. The robot acts as a server, listening for connections on a specific IP and port, while the glove connects as a client. Upon receiving signals from the glove, the server decodes the data and triggers actions based on the detected color. This setup ensures real-time, efficient coordination between the glove and the robot.

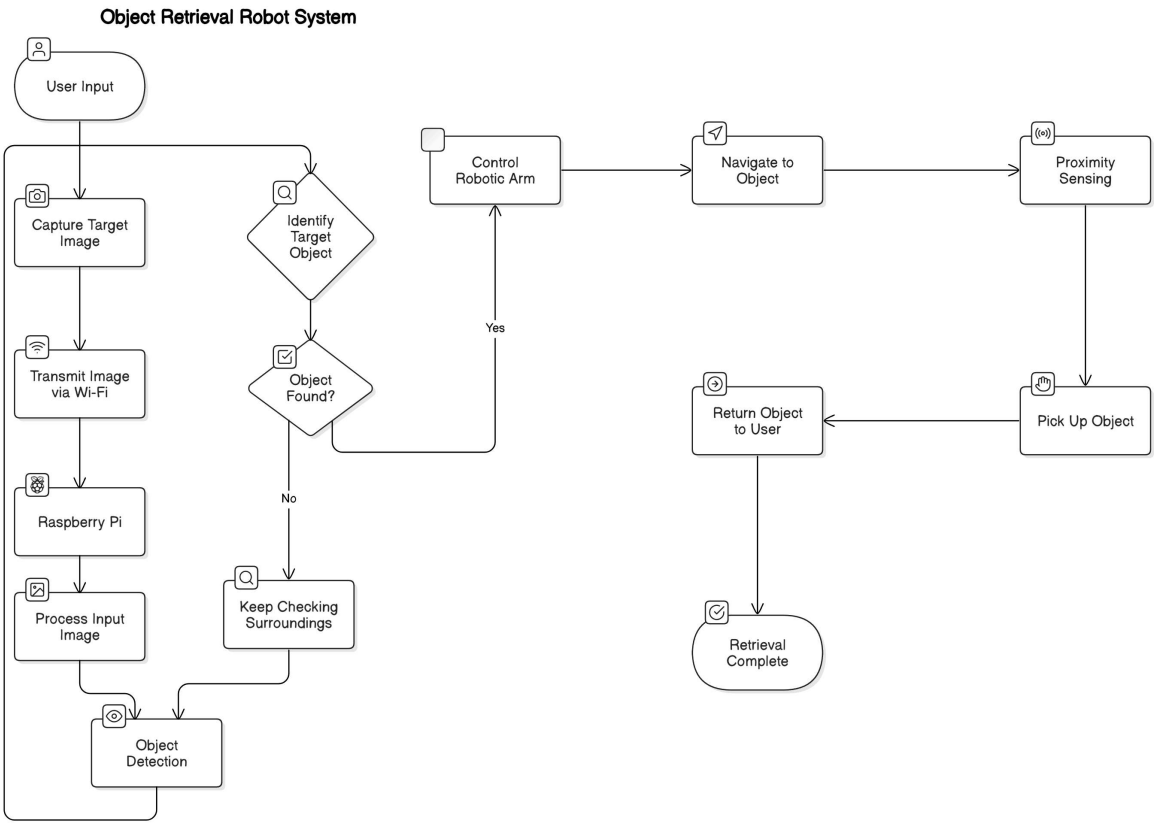
2. Arm control: This code implements a robotic arm control system using the pigpio library on a Raspberry Pi, with functions to smoothly operate servo motors. Each servo (base, shoulder, elbow, and wrist) is configured with specific GPIO pins and defined pulse width ranges for movement. A helper function ensures gradual transitions between positions to prevent jerky motions. High-level functions like default, pick, and release coordinate multiple servo movements to perform actions such as returning to a home position, picking up objects at specified coordinates, and placing them down. The system also includes a cleanup function to safely stop the servos and disconnect from the pigpio daemon. This structured approach provides precise and smooth control over the robotic arm's movements.

3. Color detect: This code implements real-time color-based object detection using OpenCV. It uses a webcam feed to identify and track colored cubes within a specified color range (e.g., Green, Blue, Yellow). The HSV color space is used to define thresholds for six colors, and contours are extracted from binary masks to locate the objects. The centroid and approximate shape of each detected object are calculated and displayed on the video feed, along with HSV values at the object's center. A threshold filter can optionally be enabled to ignore objects below a minimum Y-coordinate. The program uses threading to handle shared variables like the detected color and object position, ensuring thread-safe updates. If a detected object matches the target color and crosses a set Y-coordinate threshold, the system stops further detection and releases resources, such as the webcam feed and OpenCV windows. This program is useful for applications like object tracking and robotic control based on visual inputs.

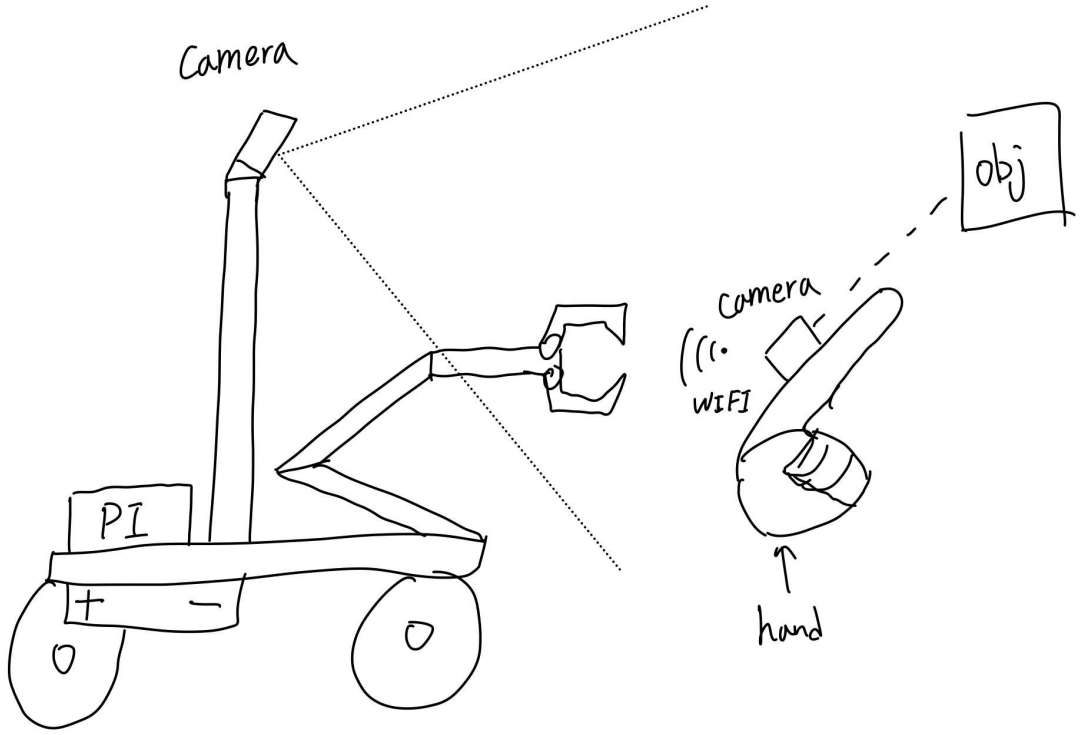
4. Motor control: This code integrates robotic motor control, computer vision, and server communication to detect a target object by its color, track it using motors, and manipulate it with a robotic arm. It uses the pygame library for GUI initialization, RPi.GPIO for motor control, and a custom module for color detection. The system fetches the target color from a server, calibrates motor control for specific positions, and uses threading to run motor control and color detection simultaneously. The motor control logic employs proportional control to align the motors with the target's position. Once the object is detected, the robotic arm picks it up and can later release it at a specified location. Robust exception handling ensures a safe cleanup of resources, stopping motors and color detection on exit. This framework provides a synchronized and automated robotic system for object detection and manipulation.

5. run (Integration): This bash script sets up a continuous loop to manage the execution of two Python scripts (motor_control.py and go_back.py) for controlling a robotic system. It starts by launching the pigpiod daemon, which is essential for managing GPIO operations on a Raspberry Pi. The loop ensures that the motor control script runs first, followed by a script to reset or reposition the robotic components (go-back script). An optional delay can be added between iterations for system stability. This script provides a simple and automated way to repeatedly execute the required control routines for the robotic system

Drawings



Flow chart



hand drawing

Testing

In our testing, we conducted separate tests for the glove and the robot. We tested the glove by pointing it at a specific color and printing the color. For the robot, we tested it by placing a ball in front of it, and the robot was able to pick up the ball. With the basic tests passed, we also implemented additional tests, such as ensuring the robot can ignore balls of different colors and pick up only the assigned color. Additionally, we tested its ability to detect the blue bar and return it to the designated location.

After thoroughly testing these two devices, we performed system integration. In this test, the glove points to a color, and the glove sends the color to the robot. The robot should follow the color contained in the message, go pick up the ball, and return it to its designated location.

In our testing, sometimes the robot can pick up the ball correctly, but there are instances where it does not work as expected. In some cases, we need to perform calibration. Additionally, due to some multithreading issues, the camera did not close correctly, causing the video to reopen unexpectedly. In general, we need to recalibrate the robot after approximately 10 pickups. Also, when the robot continuously picks up and returns balls without restarting the program, the camera may encounter issues opening properly. However, once the calibration is complete and the program is restarted, the robot can perform the task of picking up different balls in most situations.

Result

In conclusion, we have the glove point to a specific color and use Wi-Fi transmission to send the assigned color to the robot. The robot can detect the specific color ball, move towards it, pick it up, and return the ball to the designated position. We have our demonstration in the video above. In this project, we still need to improve reliability, whether by using more decision-making processes to determine whether the ball has been picked up, performing calibration automatically, adding PID control to regulate the signal, or improving multithreading for more stable performance.

All in all, we consider this five-week project to be a significant achievement and a great learning experience.

Parts List

- Raspberry Pi 4 $35.00
- Raspberry Pi Camera V2 $20.00
- Raspberry Pi Camera V2 $20.00
- Robot Arm - $22.13 (https://www.amazon.com/dp/B081YH77C3?ref=ppx%20yo2ov%20dt%20b%20fed%20asin%20title)
- Resistors, Wires and Buttons- Provided in lab

Total: $97.13

References

PiCamera Document (https://picamera.readthedocs.io/)
Tower Pro Servo Datasheet (http://www.micropik.com/PDF/SG90Servo.pdf)
Pigpio Library (http://abyz.co.uk/rpi/pigpio/)
R-Pi GPIO Document (https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/)
Open a Socket (https://projects.raspberrypi.org/en/projects/get-started-pico-w/3)
Object detecion (https://github.com/automaticdai/rpi-object-detection)