Robotic Arm on Object Sorting with Bluetooth Command

Mrs. Lubna Afza[1], Mrs. Tahsina Kauser[2], Mrs. Jabeena Banu[3], Ms. Syeda Nabeeha Tabassum[4]
[1,2,3]*Assistant Professor, Dept. of ECE, KNS Institute of Technology, Bengaluru*
[4]*UG Student of, II-Semester, Dept. of CSE-DS, KNS Institute of Technology, Bengaluru*

***Abstract*—This paper presents a color-based object sorting robot system utilizing a Raspberry Pi, camera module, and motor driver circuits to identify and sort colored boxes—specifically red, green, and blue. The robot leverages computer vision techniques, implemented using OpenCV, to detect specific color regions in real- time video frames. Based on the identified color, the robot executes pre-defined motor sequences to pick up the detected object and place it in a designated location using a gripper mechanism and mobile platform.The system integrates serial Bluetooth communication to receive commands remotely, allowing for color-specific or manual control modes. The robotic arm's movement— controlled through GPIO pins—is capable of lifting, dropping, and opening/closing the gripper based on the object color. Each color triggers a unique movement sequence to ensure correct sorting. The robot is capable of distinguishing between different color objects by processing their HSV values and tracking them using contour detection.This automated approach can be applied to warehouse automation, recycling systems, or educational robotics, demonstrating efficient integration of hardware control with real-time image processing**

***Index Terms*—Robotic Arm, DC Motor, Python, Rasberry Pi 3, Motor Drivers**

I. INTRODUCTION

In India robotic arms have become a cornerstone of modern automation, offering precision, efficiency, and consistency in various industrial and domestic applications. One of the critical implementations.

One of the critical implementations of robotic arms is in automated object sorting systems, where objects are identified, picked, and placed at designated locations.

This project explores the development of a robotic arm designed specifically for object sorting, with advanced features that enable smooth movement from a pick-up point to a designated destination.

The proposed robotic arm utilizes a combination of actuators, sensors, and a Bluetooth communication module to achieve seamless operation.Through Bluetooth connectivity, commands can be wirelessly transmitted from a smartphone or any Bluetooth-enabled device, allowing for remote control and flexibilty in operation.

II. RELATED WORK

[1] "Color based Product Sorting Machine using Raspberry Pi" Authors: Dr. P. Perumal, Dr. B. Mathivanan, Dr. K. Deepa Published: 2024 5th International Conference on Smart Electronics and Communication (ICOSEC) | 979-8-3315-0440- 3/24/$31.00 ©2024 IEEE DOI: 10.1109/ICOSEC61587.2024.10722515 4 IEEE | DOI: 10.1109/ICOSEC61587.2024.10722515

Bottle can be arranged in different groups based on their qualities, such as size, color, form, etc. This work primarily focuses on designing and developing a bottle-sorting device powered by Arduino. In this instance, the glass's height and color have been used to identify bottles from different companies. Most of the time, this is done by hand. Color and contact sensors are employed to determine the size and color of the bottles. The conveyer belt-transported bottles are sorted by turning on the appropriate product gateway. A rotary actuator with an arm removes the bottles off the belt conveyor for sorting and processing. One keeps track of the overall number of bottles sorted in LCD. This prototype sorting device conserves the time and resources that a worker would have used to hand-sort the objects. The suggested system classifies objects

using various image processing methods based on factors including color and form. The framework's input will be an image, which will then be processed to identify the variety or form. The pieces will also be organized using embedded mathematics.

[2] Design and Development of a Color Sorting Robotic ARM" Authors: Dr. Sanjay Kumar1, Shiv Sahay Yadav2, Anirudha Kalaskar Pradyumna3, Awate Maitriy Shripal4, Nerlekar Mandar Dattaprasad5.Published: International Journal of Enhanced Research in Management & Computer Applications ISSN: 2319-7471, Vol. 13, Issue 4, April-2024, Impact Factor:8.285

The integration of robotic systems in industrial applications has evolved significantly over the decades, driven by advancements in sensor technology, microcontrollers, and automation techniques. This literature review explores the various aspects of robotic arms, particularly focusing on color sorting applications, and highlights the contributions of various researchers in this field. Robotic manipulators have long been employed in industrial settings for tasks that require precision and repeatability. Angelo (2007) discusses the early development and applications of robotic manipulators, emphasizing their importance in automating repetitive tasks with high accuracy (Angelo, 2007). The International Federation of Robotics (2003) provides a comprehensive classification of robots, detailing their applications across different industries, which has laid the groundwork for understanding the diverse functionalities of robotic systems (International Federation of Robotics, 2003) The specific use of color sensors in robotic systems for sorting tasks has garnered significant attention in recent years. Zhang and Wang (2019) conducted a review on color-based sorting systems, highlighting the various techniques and technologies employed to achieve high accuracy in color detection and object sorting (Zhang & Wang, 2019)

[3] "Simulation of Pick and Place Robotic Arm using Coppeliasim" Authors: S. Rooban, R Manimegalai,, Published: 2022 6th International Conference on Computing Methodologies and Communication (ICCMC) | 978-1-6654-1028-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCMC53470.2022.9754013

To assemble the elements of the robotic system it's essential to undergo diverse structure and methodologies to achieve perfect momentum of designed robotic structure. The design flow of robotic system includes programming the task and the working environment. Robot design includes mechanical units, sensor units, actuation units and supervision units. Program sends commands to robotic system which determines actions on the environment. Mechanical units of robot include end effectors, wrist and supporting structure.

III. BLOCK DIAGRAM

As we follow along with the block diagram. This system is a Raspberry Pi–based robotic control circuit that integrates a camera, an HC-05 Bluetooth module, motor drivers, and multiple high-power DC motors, all powered by a central 5V power supply. The camera sends visual data to the Raspberry Pi, which serves as

The main controller, while the HC-05. Bluetooth module allows remote commands to be sent wireless from a smartphone or computer. The Raspberry Pi interprets the command and uses its GPIO pins to control various motors through the motor driver. Specifically, GPIO pins 26, 19, 13, and 6 control the robot's movement (forward, backward, left, and right), pins 2 and 3 operate the gripper for opening and closing, pins 21 and 20 control the lifting arm's up and down motions, and pins 17 and 27 handle additional actuator actions like moving or dropping items. Functions defined in the Python code activate specific GPIO combinations to execute tasks such as picking up colored boxes, transporting them, and dropping them at predefined locations, with each box type (green, red, or blue) having a distinct set of motor commands and timing sequences for coordinated movement. The HC-05 communicates with the Raspberry Pi through its UART pins, typically GPIO14 and GPIO15, enabling command reception and initiating the appropriate motor-driven routines coded in Python.

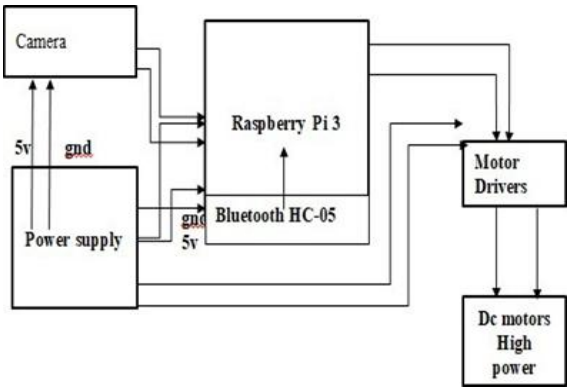


Fig.:1 Hardware Components

1. Raspberry Pi

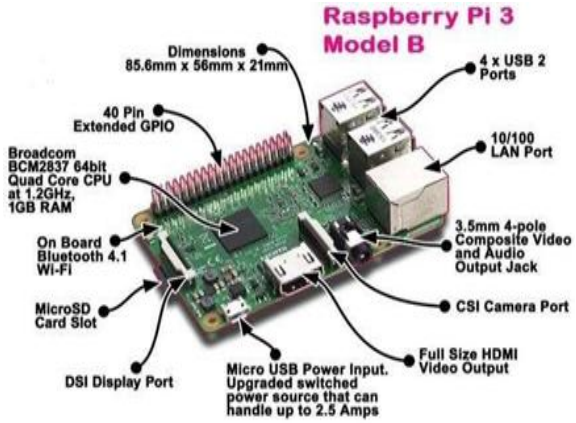


Fig.:2

Use: Acts as the central controller that processes input from the camera and Bluetooth, and controls the motors using GPIO pins.

Pin Format: Uses a 40-pin GPIO header (2x20). Key pins used in your code:

1) GPIO2 (Pin 3) and GPIO3 (Pin 5) → Gripper motor control
2) GPIO6 (Pin 31), 13 (Pin 33), 19 (Pin 35), 26 (Pin 37) → Main movement motors
3) GPIO20 (Pin 38), 21 (Pin 40) → Lifting arm
4) GPIO17 (Pin 11), 27 (Pin 13) → Additional actuator
5) GPIO14 (TXD, Pin 8), GPIO15 (RXD, Pin 10) →UART communication with HC-05
6) 5V (Pin 2 or 4) and GND (multiple pins) → Power and ground connections

2. HC-05 Bluetooth Module

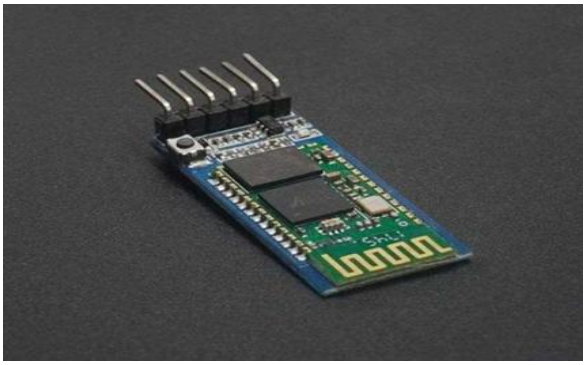


Fig.:3

Use: Enables wireless communication between Raspberry Pi and a mobile device or PC.

Pin Format:

1) VCC → Connects to 5V on Raspberry Pi (Pin 2 or 4)
2) GND → Connects to GND on Raspberry Pi
3) TXD → Connects to Raspberry Pi RXD (GPIO15, Pin 10)
4) RXD → Connects to Raspberry Pi TXD (GPIO14, Pin 8) via voltage divider (3.3V level)
5) EN (optional) → Enables the module when pulled HIGH (not always connected)

3. Motor Driver (e.g., L298N or L293D)

Fig.:4

Use: Acts as an interface between Raspberry Pi and high-power DC motors, allowing direction and speed control.

Pin Format:

1) IN1–IN4 → Logic control inputs from Raspberry Pi (for two motors)
2) IN5–IN8 → Additional inputs for other

motors like lift or gripper
3) ENA/ENB → Enable pins (sometimes wired directly to 5V or controlled by PWM)
4) VCC → Motor power (usually 12V)
5) GND → Common ground with Pi
6) OUT1–OUT4 → Connected to DC motors

4. DC Motors (High Power)

Fig.:5

Use: Used to move the robot, lift the arm, open/close the gripper, or slide an object.

Pin Format: Two wires per motor — connected to OUTx pins on motor driver:
1) Motor A → OUT1 and OUT2
2) Motor B → OUT3 and OUT4
Direction depends on input signals from Raspberry Pi via motor driver.

5. Camera Module (e.g., Pi Camera)

Fig.:6

Use: Captures real-time video or images for image processing (e.g., object recognition).

Pin Format:
1) Connects to Raspberry Pi's CSI camera port
2) Powered via 5V and GND
3) Uses MIPI-CSI interface, not GPIO pins

6. Power Supply
Use: Provides regulated 5V and GND to all components.

Pin Format:
1) 5V output → Connected to Raspberry Pi, Camera, Bluetooth, Motor Driver logic
2) GND → Common ground to all components
3) May also provide separate 12V line to motor driver for DC motors if needed

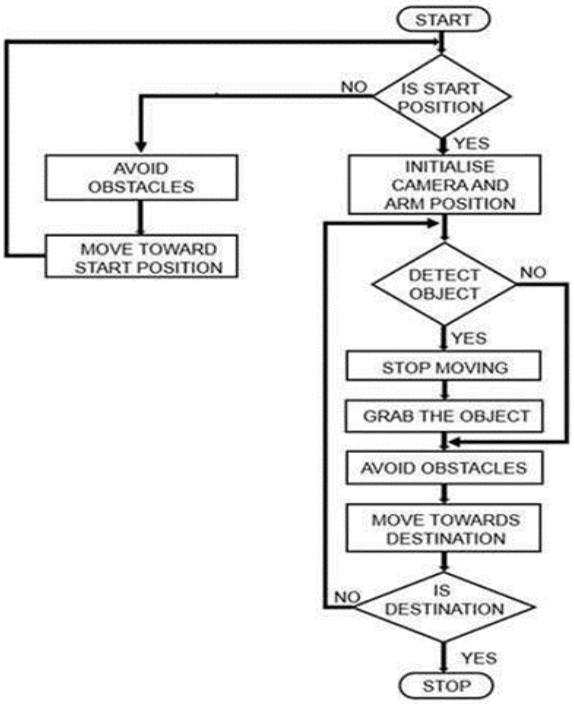
Software Components
Python
The system uses Python as the programming language because it is well-supported on the Raspberry Pi and provides easy-to-use libraries for hardware control. Specifically, the RPi.GPIO library is used to configure the Raspberry Pi's GPIO pins as outputs, allowing the script to send electrical signals that control various components such as DC motors, servos, and actuators through a motor driver. The time library is used to introduce delays between actions, ensuring motors have enough time to complete their movements before the next command begins. Python's simplicity makes it ideal for real-time robotic tasks like moving the robot forward, gripping an object, lifting it, and dropping it at a target location—all of which are defined in modular functions that can be reused for different colored boxes or movement sequences.

IV. METHODOLOGY

The methodology used in the given Python program is a modular, function-based control system for operating a robotic arm or vehicle using a Raspberry Pi. The logic is structured to handle physical movement, object handling, and automated task execution through well-defined steps. Here's a sentence-based explanation of how the methodology works:
The program begins by importing necessary libraries, setting up the GPIO pin mode to BCM, and configuring specific GPIO pins as outputs to control motors and actuators. It then defines a

series of modular functions, each responsible for a specific action such as moving forward, turning, stopping, raising or lowering a lift, opening or closing a gripper, or shifting an actuator. These low-level functions are grouped into higher- level task functions like green_box(), red_box(), and blue_box() which represent complete object-handling routines for sorting different colored boxes. Each of these task functions combines motor control, timing delays, and sequential operations to pick up a box, move it to a location, and return to the original position. This modular and sequential methodology allows the robot to perform complex tasks reliably by combining simple, reusable functions in a structured flow.

V. FLOW CHART



VI. IMPLEMENTATION

The project is a Raspberry Pi-based autonomous sorting robot designed to detect the color of an object and place it into the correct bin. The robot uses several DC motors, controlled through the Raspberry Pi's GPIO pins, to move around, pick up items using a gripper, lift and lower its arm, and rotate or shift its base to align with different bins.

The robot begins by initializing all the necessary GPIO pins, setting them as outputs, and ensuring all motors are initially off.

There are different motors connected to the Raspberry Pi: two motors for driving the wheels (controlled via IN1 to IN4), one motor for opening and closing the gripper (IN5 and IN6), another motor to lift and lower the arm (IN7 and IN8), and finally, one more motor to shift or rotate the robot's base left or right (IN9 and IN10). The wheel motors allow the robot to move forward, backward, left, or right. The arm motor allows the robot to pick up and place objects vertically, while the gripper motor controls the grip on objects. The final motor is used to slightly shift the robot's position or orientation before or after placing an object into a bin.

The robot receives commands via Bluetooth using a serial interface. Based on the received command—for example, 'rg' which might mean "red item to green bin"—the Raspberry Pi triggers the appropriate color detection script, such as red_detection.py. This script uses OpenCV and a webcam to detect specific colors in front of the robot. It converts the video feed into HSV color space and applies masking to isolate the desired color. When the color is detected consistently over several frames, the script exits and the robot proceeds to the sorting action.

Each color command triggers a predefined routine. For example, if a red object is detected and it must go to the green bin, the robot first lowers the arm, opens the gripper, then closes it to grab the object. Next, it lifts the object using the arm, possibly rotates to align with the correct bin, moves forward for a set duration, and drops the object. After placing the object, the robot resets its gripper and returns to its original position, ready for the next command.

All of these actions are automated and rely on precise timing and motor control. The gripper and arm movements are controlled using timed delays to ensure the motors run for just enough time to complete their action. Similarly, forward or backward movement is handled using timed runs, which assume that the robot will reach the correct location in that fixed amount of time.

Overall, this system combines computer vision, motor control, and Bluetooth communication to

create a basic but functional color-sorting robot. It demonstrates how a Raspberry Pi can serve as the central controller for coordinating sensors (the camera), actuators (the motors), and communication (Bluetooth), allowing the robot to perform tasks like object detection and sorting autonomously.

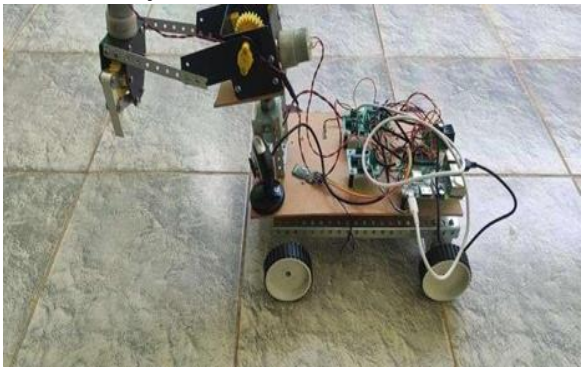
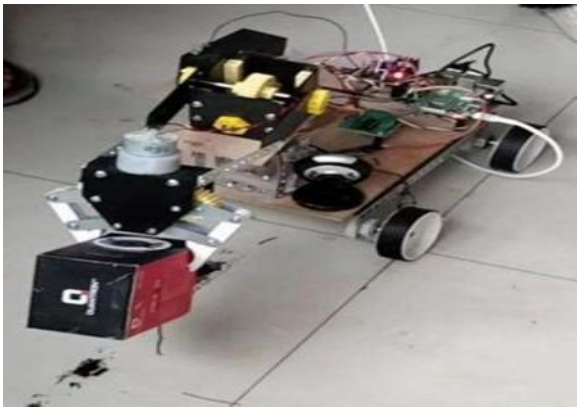


Fig-7 Implementation

VII. RESULT AND FUTURE SCOPE



The implemented color-sorting robot functions effectively by detecting red, green, and blue colored objects using real-time camera input and sorting them into respective bins through coordinated motor movements. The system successfully interprets Bluetooth commands, processes live video through
OpenCV for color recognition, and controls various DC motors via GPIO pins on the Raspberry Pi to carry out gripping, lifting, movement, and dropping operations. Testing has shown that the robot can consistently recognize colors, pick up objects, and deposit them accurately into predefined locations, validating the system's design and functionality.

FUTURE SCOPE

This paper has significant potential for expansion and improvement. In the future, the system can be enhanced with ultrasonic or infrared sensors to allow for obstacle detection and autonomous path planning, making the robot more adaptable in dynamic environments. Machine learning algorithms could be integrated to improve color recognition accuracy under varying lighting conditions or to recognize more complex object features like shape or texture. The gripper mechanism could also be upgraded with servos for more precise control, and the entire robot can be placed on a mobile chassis to perform sorting in larger or distributed workspaces. Furthermore, IoT integration could enable remote monitoring and control of the sorting process, making it suitable for industrial or smart warehouse applications.

VIII. CONCLUSION

In conclusion, this paper successfully demonstrates an intelligent robotic system capable of detecting object colors and sorting them into designated bins using a Raspberry Pi. By integrating computer vision through OpenCV, precise motor control using GPIO outputs, and wireless communication via Bluetooth, the robot can autonomously identify red, green, or blue objects and perform the required physical movements to sort them correctly. The system showcases how embedded hardware and software can work together to solve real-world automation tasks, laying a strong foundation for further development in robotic sorting, smart automation, and AI-assisted control systems.

ACKNOWLEDGMENT

We are grateful to the Founder & Late Chairman of our college, Mr. C. K. Jaffer Sharief, for having provided us with excellent facilities in the college.
We are indebted to the Chairman of our college, Mr.Abdul Rahman Sharief, for his constant support, motivation and encouragement
We thank our Principal, D. S. M. Prakash, for faciltating a congenial academic environment in the college.
We are thankful to our HOD. Dr. Aijaz Ali Khan, for his kind support, guidance and motivation.

REFERENCES

[1] S. Rajarajan, T. Kowsalya, N. S. Gupta, P. M. Suresh, P. Ilampiray, and S. Murugan, "IoT in brain-computer interfaces for enabling communication and control for the disabled," in Proc. 10th Int. Conf. Communication and Signal Processing, 2024, pp. 502–507.
[2] R. Raman, M. Kaul, R. Meenakshi, S. Jayaprakash, R. Ramya, and C. Srinivasan, "IoT-based magnetic field strength monitoring for industrial applications," in Proc. 2nd Int. Conf. Smart Technologies for Smart Nation, 2023, pp. 132–136.
[3] B. M. Kannan, P. Solainayagi, H. Azath, S. Murugan, and C. Srinivasan, "Secure communication in IoT-enabled embedded systems for military applications using encryption," in Proc. 2nd Int. Conf. Edge Computing and Applications, 2023, pp. 1385–1389.
[4] C. Jehan, P. S. Kumaresh, M. Raja Suguna, R. Anto Arockia Rosaline, S. M, and S. Murugan, "Adaptive silo networks with cloud computing and reinforcement learning for responsive grain storage," in Proc. Int. Conf. Inventive Computation Technologies, 2024, pp. 1611–1616.
[5] N. Vijayalakshmi, S. Yuvaraj, V. V. Baskar, R. Krishnaswamy, K. Sasikala, and C. Srinivasan, "Optimized control of IoT-monitored microgrid systems using genetic algorithm," in Proc. Int. Conf. Power, Energy, Environment & Intelligent Control, 2023, pp. 1014–1018.
[6] A. G. Deshpande, C. Srinivasan, R. Raman, S. Rajarajan, and R. Adhvaryu, "Enhancing cloud security: A deep cryptographic analysis," in Proc. Int. Conf. Advances in Computation, Communication and Information Technology, 2023, pp. 1118–1123.
[7] P. A. Priya M, P. Karthikeyani, N. Arunfred, M. Hariharan, K. Ramyadevi, and S. Murugan, "IoT and hydrogen transport: Revolutionizing fuel cell vehicle infrastructure," in Proc. 4th Int. Conf. Innovative Practices in Technology and Management, 2024, pp. 1–6.
[8] P. Srinivas, M. Arulprakash, M. Vadivel, N. Anusha, G. Rajasekar, and C. Srinivasan, "Support vector machines-based predictive seizure care using IoT-wearable EEG devices for proactive intervention in epilepsy," in Proc. 2nd Int. Conf. Computer, Communication and Control, 2024, pp. 1–5.
[9] M. Mubarakali and H. Azath, "Healthcare services monitoring in cloud using secure and robust healthcare-based blockchain (SRHB) approach," Mobile Netw. Appl., vol. 25, pp. 1330–1337, 2020.
[10] S. P. Vimal, M. Vadivel, V. V. Baskar, V. G. Sivakumar, and C. Srinivasan, "Integrating IoT and machine learning for real-time patient health monitoring with sensor networks," in Proc. 4th Int. Conf. Smart Electronics and Communication, 2023, pp. 574–578.