Automated Shopping Cart Using OpenCV

Dr. Ushadevi M B[1], Gowri R[2], Yashaskara T G[3], Vinaya Kumara S[4], Srujan D[5]
[1]*Professor, Dept. of Electronics & Telecommunication Engineering, JNNCE*
[2345]*UG student, Dept. of Electronic & Telecommunication Engineering, JNNCE*

Abstract—**This paper presents the design and implementation of an autonomous smart moving cart tailored for retail environments, utilizing the OpenCV algorithm for precise object detection, real-time updates, and secure navigation. The system architecture integrates key hardware components including a camera module, Arduino/Raspberry Pi microcontroller, RFID reader, LCD display, Wi-Fi module, ultrasonic sensors, and motor drivers to enable mobility and intelligent control. The smart cart leverages OpenCV to detect obstacles and track motion, ensuring smooth aisle navigation while minimizing collisions. The RFID reader facilitates product identification and inventory management, transmitting real-time updates to both the user and a centralized system via Wi-Fi. An LCD display provides a user-friendly interface, showcasing product details, cart status, and navigation cues. Ultrasonic sensors enhance navigational safety by enabling dynamic obstacle avoidance. This real-time data processing capability allows the cart to adapt to its environment, reducing human error and improving the overall shopping experience. By integrating advanced technologies and real-time communication, the proposed system streamlines retail operations and delivers a seamless, efficient shopping experience.**

Index Terms—**Autonomous Cart, OpenCV, RFID, Raspberry Pi, Retail Automation, Real-Time Navigation, Ultrasonic Sensors, Smart Shopping**

I. INTRODUCTION

The Smart Shopping Cart project revolutionizes the traditional shopping experience by integrating modern technologies to create a hands-free, autonomous retail solution. This innovative cart is designed to follow customers throughout the store, eliminating the need to manually push or manage a cart. By enabling shoppers to focus solely on product selection, the system significantly enhances customer convenience and streamlines the overall shopping journey. In addition to mobility, the Smart Shopping Cart incorporates automated checkout functionality. As customers add items to the cart, the system automatically calculates the total cost, allowing for quick and efficient purchase completion without waiting in long queues. This automation reduces dependency on traditional checkout counters and improves operational efficiency. The system also promotes sustainability by replacing paper receipts with digital alternatives, thereby minimizing paper waste and contributing to an eco-friendly retail environment. Through the use of RFID technology, real-time data processing, and user-friendly interfaces, the Smart Shopping Cart offers a seamless and modernized shopping experience. Overall, this project represents a significant advancement in retail automation. By combining autonomous navigation, real-time billing, and environmentally conscious practices, the Smart Shopping Cart transforms conventional shopping into a more efficient, intelligent, and customer-centric process.

II. LITERATURE SURVEY

A. "Smart Shopping Trolley for Automated Billing Process Using Image Processing" Proposed by Dr. Nagendra Kumar M., Nandini S., Priya C., Supriya N., and Varun Kumar K. (2021), this paper presents a smart shopping trolley that automates billing using image processing techniques. The system employs Raspberry Pi and related technologies to reduce checkout time and improve customer satisfaction in supermarkets and hypermarkets.

B. "Automated Shopping Trolley Using Raspberry Pi Device" Authored by Mohd. Ashif Khan, Nikunj Bharadwaj, Nihal Kotwal, Yogesh Yadav, and Pragati Shrivastava Deb (2021), this work integrates Raspberry Pi, barcode scanner, and LCD display to enable real-time billing and self-scanning. The system enhances retail efficiency and reduces waiting times at checkout counters.

C. "Smart Car Parking System Using Raspberry Pi" Published by IJARCCE (2021), this paper explores an intelligent parking system using Raspberry Pi, RFID tags, and IR sensors. The system identifies vehicles, detects available slots, and manages data via a central controller, aiming to reduce parking time and fuel consumption through IoT integration.

D. "A Review on Automated Billing for Smart Shopping System Using IoT" Presented by Priyanka S. Sahare, Anup Gade, and Jayant Rohankar (2019), this review discusses IoT-based smart shopping systems. It highlights the use of RFID, Raspberry Pi, and LCD displays to automate billing, reduce human error, and improve inventory management.

E. "A Survey on Smart Trolley System Based on Android Application" Published on Academia.edu (2019), this study investigates a smart trolley system using an Android app. Customers scan product barcodes via smartphones, which are automatically added to the cart and billed through the app, reducing checkout queues and enhancing retail convenience.

III. PROBLEM STATEMENT

The traditional shopping experience presents several challenges from both customer and retailer perspectives. Customers often face long checkout lines, leading to frustration and wasted time, while manual billing errors can result in incorrect charges. The lack of real-time updates on total spending and difficulties in identifying or scanning products further diminish the shopping experience. Additionally, the physical effort required to push a cart can be inconvenient, especially in crowded retail environments. From the retailer's standpoint, managing queues and maintaining customer satisfaction is demanding. Inventory tracking is often slow and inaccurate, and labor costs rise due to the need for multiple checkout operators. Manual processes are prone to errors, causing financial discrepancies and inefficiencies. Moreover, competitive pressure drives retailers to adopt automated solutions to remain relevant in a rapidly evolving market. The proposed automated smart cart system addresses these issues by integrating OpenCV and Raspberry Pi-based technologies, offering a hands-free, intelligent, and efficient solution. This innovation not only enhances customer convenience but also improves operational efficiency, positioning it as a transformative approach to modern retail.

IV. OBJECTIVES

The primary objective of this project is to develop an intelligent smart cart system that integrates OpenCV with Raspberry Pi microcontrollers to enable autonomous navigation and object detection. The system aims to implement RFID technology for automatic product identification, thereby streamlining the billing process. Real-time billing updates will be displayed on an LCD screen, enhancing user interaction and transparency. Wireless communication between the smart cart and the central billing system will be established to ensure seamless data transfer and synchronization. By minimizing checkout times and reducing human errors in billing, the system enhances operational efficiency. Additionally, the cart is designed to move autonomously, eliminating the need for manual effort and offering a hands-free shopping experience.

V. METHODOLOGY

The methodology for this project involves the integration of multiple hardware and software components to develop an efficient and autonomous smart cart system. The cart's movement is controlled by motor drivers such as the *L298N*, which receive directional signals from the *Raspberry Pi* based on the customer's position. The Raspberry Pi dynamically adjusts the cart's speed and trajectory to maintain proximity to the user, enabling a hands-free shopping experience.

Serving as the central controller, the Raspberry Pi manages data processing, decision-making, and communication across all modules. It processes camera inputs using *OpenCV* algorithms for object detection and tracking, handles *RFID* data for product identification, calculates the total bill, and communicates with the *Wi-Fi module* to send SMS notifications.

RFID tags affixed to products are scanned by the RFID module when items are placed in the cart. The Raspberry Pi processes this data and updates the *LCD display* with product details and pricing. Real-time billing is maintained, with continuous updates to the total bill amount as items are added or removed.

The system supports wireless communication, transmitting billing data to a centralized system and to the user's smartphone for remote monitoring and payment. Additional features enhance both user experience and security. If a product is removed from the cart, the system allows for bill adjustment, and unscanned items trigger an alarm to prevent billing discrepancies.

OpenCV, a powerful tool for computer vision tasks, plays a critical role in enabling autonomous navigation by facilitating object detection, motion tracking, and image processing. This integrated methodology ensures that the smart cart follows the customer throughout the store, reduces human error in billing, and provides valuable data analytics for retailers—thereby modernizing and optimizing the retail shopping experience.

VI. SYSTEM DESIGN

The smart shopping cart system is an innovative solution designed to automate and streamline the shopping experience by combining *robotics*, *computer vision*, and *IoT* technologies. At its core, a *Raspberry Pi* acts as the central processing unit, managing all operations. The cart uses a *camera module* with *OpenCV* to track and follow a customer wearing a designated red belt. The camera continuously captures video, and *OpenCV* processes the feed to detect the red color. Based on the belt's position, the *Raspberry Pi* generates commands to control the cart's movement, which are executed by an *L298N motor driver* connected to two *12V 555 motors*. To ensure safety and accuracy, an *ultrasonic sensor* measures the distance between the cart and the customer, maintaining a fixed following distance and avoiding obstacles.

The cart is equipped with an *RFID scanner* that allows customers to scan the *RFID tags* of products they wish to add to their cart. Once scanned, the product details are sent to a cloud-connected *website* linked to the cart. The website serves as an interface for customers to view the products in their virtual cart, add or remove items, and monitor their ongoing shopping session. At the end of the shopping process, customers can click the *"End Shopping"* button on the website, which calculates the total bill, provides a breakdown of purchased items, and offers *payment options*.

The system's motorized design enables *autonomous navigation*, while the combination of *computer vision* and *sensor data* ensures smooth and reliable tracking of the customer. The integration of *RFID technology* and a user-friendly *web interface* simplifies the process of adding and managing items, eliminating the need for manual intervention at checkout. Overall, this smart shopping cart system delivers a futuristic, user-centric approach to shopping, enhancing *convenience*, *efficiency*, and *customer satisfaction*.

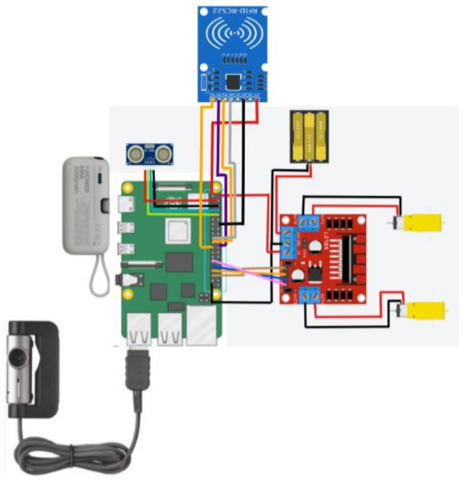


Fig. Circuit diagram of system

Technical Specifications

The smart shopping cart system is powered by a *Raspberry Pi 4 Model B* microcontroller, which serves as the central processing unit. It operates at an *input voltage of 5V*, supplied either via USB-C or through the GPIO header. The recommended voltage range is *4.75V to 5.25V* for *USB-C* and *3.3V to 5V* for GPIO input. The board features *40 GPIO pins* for digital input/output operations, although it does not support analog input natively—requiring an external ADC for such functionality. Each GPIO pin can handle a maximum current of *16mA*, with a total current limit of *50mA* across all pins. Storage is managed via a *MicroSD card*, typically supporting capacities of *32GB* or higher. The system is equipped with *4GB LPDDR4-3200 SDRAM* for memory and operates at a clock speed of *1.5 GHz*, powered by a *quad-core ARM Cortex-A72* (64-bit) processor. These specifications ensure robust performance and efficient multitasking for real-time data processing and control within the smart cart system.

General Specifications

The Raspberry Pi 4 Model B exhibits variable power consumption depending on usage and connected peripherals, typically drawing around *600–700 mA* during normal operation. It features a versatile set of USB connections, including two *USB 3.0 ports, two USB 2.0 ports, a USB-C power port*, and an *Ethernet port* for high-speed networking. The device supports multiple programming environments, with Thonny (Python IDE) being a preferred choice for development due to its simplicity and effectiveness. Additionally, the Raspberry Pi is compatible with various operating systems, most notably Raspberry Pi OS (formerly Raspbian), which offers a stable and robust platform for software deployment and system control. These specifications make the Raspberry Pi an ideal choice for embedded applications such as the smart shopping cart system.

A. Pictorial Representation

The figure below illustrates the conceptual architecture of the smart shopping cart system. The cart integrates multiple technological components including a camera module, microcontroller, and wireless communication interface. A laptop is shown interacting with the cart, representing remote monitoring and data exchange. The camera mounted on the cart enables object detection and tracking, while the embedded electronics support autonomous navigation and real-time billing.

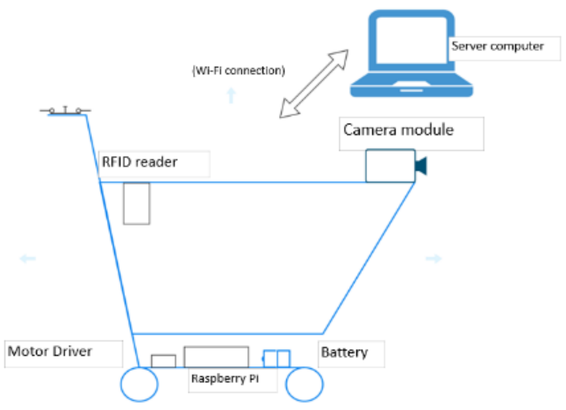


Fig. Pictorial representation of smart shopping cart system architecture

The block diagram represents the core architecture of the self-balancing autonomous bicycle system, with the *Raspberry Pi* serving as the *central processing unit.*

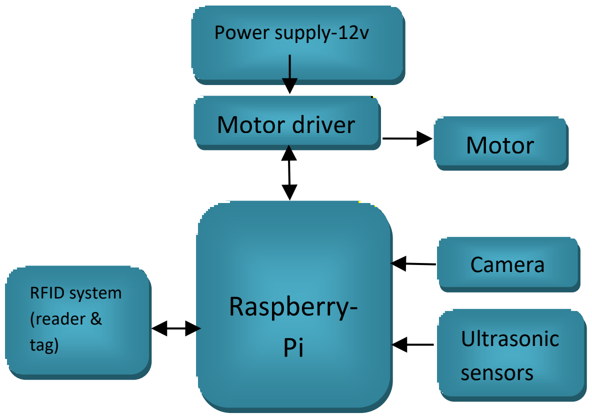


Fig. Block Diagram representation

It interfaces with a range of sensors and actuators to facilitate both dynamic stabilization an autonomous navigation. The block diagram below illustrates the overall system architecture of the smart cart, centred around the *Raspberry Pi* microcontroller. Each component plays a critical role in enabling autonomous navigation, product identification, and real-time billing. The *power supply* delivers *12V* to the *motor driver*, which controls the *motor* responsible for cart movement. The *Raspberry Pi* processes inputs from the *RFID system*, *camera*, and *ultrasonic sensors*, and coordinates all decision-making and communication tasks. It also interfaces with the *LCD display* and *Wi-Fi module* to provide real-time updates and remote monitoring capabilities.

B. Working Principle

The working procedure of the smart shopping cart system involves multiple stages, seamlessly integrating various components to deliver a user-friendly and automated shopping experience. Upon startup, the *Raspberry Pi* initializes all connected devices including the *camera module*, *ultrasonic sensors*, *RFID scanner*, and *motor driver*. The *OpenCV* software begins processing the camera feed to detect the red belt worn by the customer. The analysed by the *Raspberry Pi* using color segmentation techniques to identify the red belt based on its unique colour signature. Once detected, the system determines the customer's position and sends commands to the *L298N motor driver* to adjust the cart's movement—forward, backward, left, or right—ensuring it follows the customer while maintaining the belt in view. The *ultrasonic sensor* monitors the distance between the cart and the customer, slowing or stopping the cart if the distance falls below a safe threshold.

For product scanning, the cart features an accessible *RFID scanner*. When a product is scanned, the *Raspberry Pi* reads the tag data and sends it to a cloud-connected server, which updates the linked website with product details such as name, price, and quantity. The website serves as a virtual interface, displaying a real-time list of items added to the cart. Customers can access it via smartphone or other devices to view, add, or remove items, with all changes synchronized in real time. As the customer navigates the store, the cart uses tracking data from the *camera* and *ultrasonic sensor* to follow smoothly and avoid obstacles. If an object or person is detected in its path, the cart adjusts its movement to prevent collisions.

At the end of the shopping session, the customer clicks the *"End Shopping"* button on the website. The system calculates the total cost and presents payment options such as card, *UPI*, or mobile wallets. Upon successful payment, the cart resets for the next user. Power management is handled by the *Raspberry Pi*, which monitors battery levels and issues alerts when charging is needed. Safety protocols ensure the cart halts if the red belt is not detected or if the connection to the website is lost. By integrating *computer vision*, *IoT*, and *robotics*, the system automates the shopping process, reduces manual effort, and enhances overall customer convenience.

C. Hardware and Software requirements

Hardware-The core of the smart shopping cart system is the *Raspberry Pi*, which acts as the central controller. It coordinates all system operations and manages communication between various components. Powered by a standard *5V* supply via *micro-USB* or *USB-C*, the *Raspberry Pi* uses its *GPIO pins* to control the *motor driver*, process inputs from the *ultrasonic sensor*, and interact with the *RFID scanner*. It also connects to a local *Wi-Fi* network, enabling real-time data exchange with the cart's linked website, which displays the cart's contents and allows user interaction.

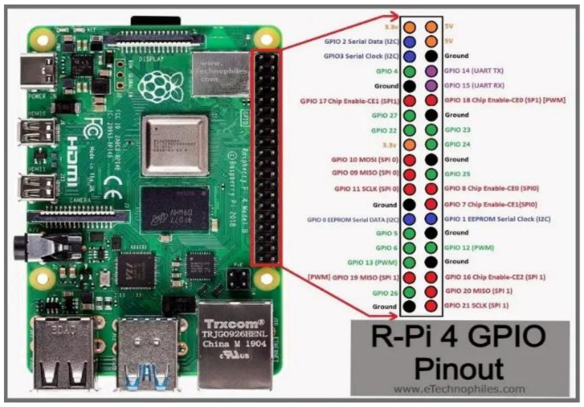
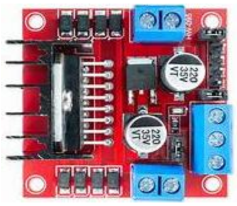


Fig. 3.6.1 – Raspberry Pi 4 Model B

The *L298N motor driver* is responsible for controlling the *12V 555 motors* that drive the cart's wheels. It enables precise control of motor speed and direction. Powered by a *12V battery*, the driver uses control pins (*IN1, IN2, IN3, IN4*) connected to the *Raspberry Pi* to manage directional movement. The enable pins (*ENA, ENB*) are connected to a *5V* supply to activate the motors, while output pins (*OUT1–OUT4*) connect directly to the motors. This setup ensures smooth navigation and obstacle avoidance based on



instructions from the *Raspberry Pi*.

Fig. 3.6.2 – L298N Motor Driver

The *12V 555 motors* provide the mechanical movement for the cart. Controlled by the *L298N driver*, these motors adjust speed and direction based on data processed by the *Raspberry Pi*. Each motor is connected to two output pins of the driver, enabling directional control such as forward, backward, left, and right. These motors drive the cart's wheels and respond to real-time commands for dynamic navigation.

Fig. 3.6.3 – 12V 555 Motor

The *ultrasonic sensor* (typically *HC-SR04*) measures the distance between the cart and the customer, ensuring a safe following distance. Powered by the *5V* pin of the *Raspberry Pi*, it uses four connections: *VCC* (5V), *GND* (ground), *Trig* (trigger pulse), and *Echo* (reflected pulse). These are connected to *GPIO pins* on the *Raspberry Pi*, which calculates the distance and adjusts motor commands accordingly. If the customer

is too close, the cart slows or stops; if too far, it accelerates to catch up.

Fig. 3.6.4 – Ultrasonic Sensor

The *RFID scanner* (e.g., *MFRC-522*) scans RFID tags attached to products. When a product is scanned, the scanner sends its unique ID to the *Raspberry Pi*, which updates the virtual cart on the website. Powered by the *3.3V* pin, the scanner uses connections such as *SDA*, *SCK*, *MOSI*, *MISO*, *IRQ*, and *RST*, all linked to *GPIO pins* for data transfer and control. This enables seamless product identification and real-time cart updates.

Fig. 3.6.5 – RFID Scanner

The *camera module* is used to track the customer wearing a red belt. Connected via *USB* and powered through the *3.3V* pin, it provides real-time video to the *Raspberry Pi*. Using *OpenCV*, the system detects the red belt and sends movement commands to the motor driver. The cart adjusts its position to maintain a consistent distance, using data from both the camera and the *ultrasonic sensor*.

Fig. 3.6.6 – Camera Module

Power management is handled through a dual supply system. The *Raspberry Pi* is powered by a *5V adapter* or portable power bank, while the motors and motor driver are powered by a separate *12V lithium-ion battery pack* configured as *3S5P* (3 series, 5 parallel). A *DC-DC step-down converter* ensures appropriate

voltage levels for each component, preventing power interference and ensuring stable operation.

Fig. 3.6.7 – Power Supply & Battery Setup

Software- To operate and manage the smart cart system, two key software tools were used: *RealVNC* and *Thonny (Python)*. *RealVNC* is a remote access software that allows developers to control the *Raspberry Pi* from any location. This facilitates convenient software deployment and system monitoring.

Thonny is a beginner-friendly *Python IDE* used to write and debug the code running on the *Raspberry Pi*. This code integrates various components such as *RFID technology* for product identification, *OpenCV* for visual recognition, and *ultrasonic sensors* for obstacle detection and navigation. The combination of these tools enabled seamless integration of hardware and ensured accurate, efficient performance of the smart cart system.

VII. IMPLEMENTATION

The implementation of the smart shopping cart system integrates both hardware and software components to deliver a seamless, automated shopping experience. The system overview remains consistent, with the *Mu Python Emulator* used for coding and debugging *Python* scripts, and *System Studios* employed for designing and managing the cart's *web interface*. The hardware setup includes the *Raspberry Pi*, *camera module*, *ultrasonic sensor*, *RFID scanner*, *L298N motor driver*, *12V motors*, and a dedicated *power supply*, all connected through proper wiring as previously described.

For software implementation, the *Mu Editor* serves as a lightweight and beginner-friendly *Python IDE*, ideal for real-time development and debugging. It supports execution of scripts using *OpenCV* for visual tracking and *GPIO* libraries for hardware control. Developers can write and test Python code for key functionalities such as *red belt detection*, *distance measurement*, and *RFID-based product scanning*, ensuring smooth interaction between software and hardware components. This integrated approach enables the smart cart to operate efficiently, combining computer vision, sensor data, and web-based control to enhance the overall shopping experience.

Steps for Using Mu:

I. Install Mu:

```
sudo apt install python3-mu-editor
```

Alternatively, you can download Mu from its official website if working on a desktop.

II. Developing Python Scripts:

Write and test Python scripts for:

- OpenCV-based red belt detection.
- Ultrasonic distance maintenance.
- RFID scanner operations

Example code snippet for red belt detection using Mu:

```
import cv2
import numpy as np
def track_red_belt():
  cap = cv2.VideoCapture(0)  # Initialize the camera
  while True:
    _, frame = cap.read()
    hsv = cv2.cvtColor(frame,
cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 120, 70])
    upper_red = np.array([10, 255, 255])
    mask = cv2.inRange(hsv, lower_red, upper_red)
    # Find contours for tracking
    contours, _ = cv2.findContours(mask,
cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    for contour in contours:
      if cv2.contourArea(contour) > 500:
        x, y, w, h = cv2.boundingRect(contour)
        cv2.rectangle(frame, (x, y), (x + w, y + h),
(255, 0, 0), 2)

    cv2.imshow('Red Belt Tracking', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
      break
  cap.release()
  cv2.destroyAllWindows()
if _name_ == "_main_":
  track_red_belt()
```

#Run this script in Mu to test the camera's ability to detect the red belt.

Web Design Using System Studios:

System Studios is used to design and implement the web interface for the cart.

Steps for Building the Web Interface:

A. Frontend Design

Use System Studios to create an intuitive and responsive interface. The web interface of the smart shopping cart system includes several essential components designed to enhance user interaction and streamline the shopping process. These components include a *product list display* that shows all items currently added to the cart, *"Add"* and *"Remove"* buttons that allow users to manage cart contents dynamically, and an *"End Shopping"* button that finalizes the purchase and initiates the billing and payment process. Together, these features provide a responsive and intuitive interface that supports real-time updates and simplifies the overall shopping experience.

Sample HTML Layout:

```
<div class="cart-container">
  <h1>Shopping Cart</h1>
  <div id="product-list"></div>
  <button onclick="endShopping()">End
Shopping</button>
</div>
```

B. Backend Development

Use Flask (or Django) as the backend framework. APIs for adding/removing products, retrieving product lists, and handling checkout.

Sample Flask API:

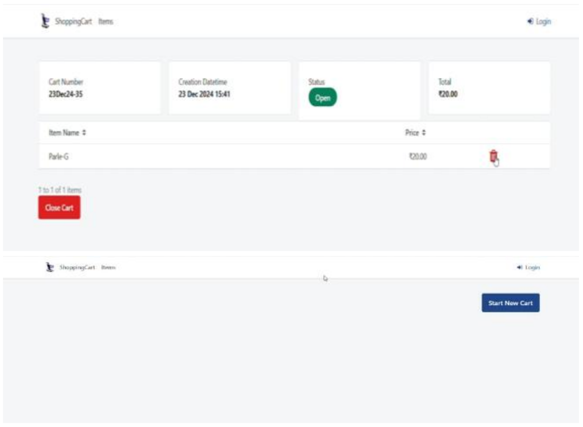
```
from flask import Flask, request, jsonify
app = Flask(_name_)
cart = []
@app.route('/add_product', methods=['POST'])
def add_product():
  product = request.json
  cart.append(product)
  return jsonify(cart)
@app.route('/remove_product', methods=['POST'])
def remove_product():

  product = request.json
  cart.remove(product)
  return jsonify(cart)
@app.route('/checkout', methods=['GET'])
def checkout():
  total = sum(item['price'] for item in cart)
  return jsonify({"total": total})
```

```
if _name_ == "_main_":
    app.run(debug=True)
```

C. Integration with Cart

To ensure seamless functionality, the *RFID scanner* is integrated with the cart's *web interface* using *APIs* that enable real-time updates. When a customer scans a product using the *RFID scanner*, the system sends a *POST request* to the */add product* API, which updates the virtual cart stored on the server. The updated product list is then dynamically displayed on the *web interface*, allowing the customer to view newly added items instantly. This integration ensures that every scanned product is accurately reflected in the cart, enhancing transparency and user control throughout the shopping process.



Cart Number	Creation Datetime	Status	Total
23Dec24-35	23 Dec 2024 15:41	Open	₹20.00

Item Name	Price
Parle-G	₹20.00

1 to 1 of 1 items

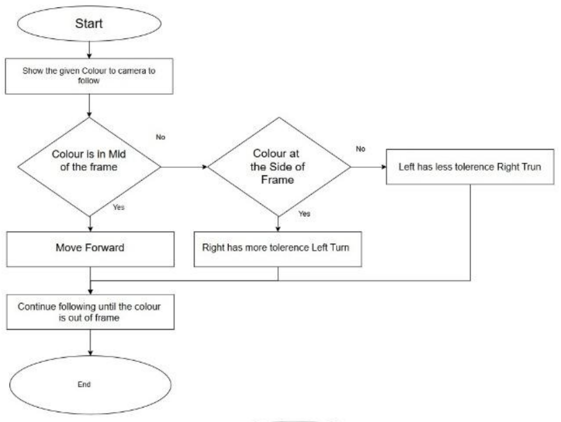
Fig. Developed Web page picture

D. Workflow

The smart shopping cart system follows a structured workflow that integrates tracking, product management, web interaction, and safety protocols. Initially, the *camera module* tracks the red belt worn by the customer, and the cart follows using *OpenCV* for visual recognition and *motor control* via the *L298N driver*. As the customer shops, products are scanned using the *RFID scanner*, and the scanned data is updated on the *website* in real time. The user can manage the cart's contents and initiate checkout directly through the *web interface*. To maintain a safe following distance, the *ultrasonic sensor* continuously monitors the space between the cart and the customer, adjusting movement as needed. During development, *Python scripts* are tested and debugged using the *Mu Emulator* to ensure proper hardware interaction, while the website is tested in *System Studios* for responsiveness and *API functionality*. This workflow ensures a smooth, autonomous, and user-friendly shopping experience.

VIII. FLOWCHART

The smart shopping cart system incorporates two primary flowcharts to illustrate its operational logic. The first flowchart outlines the *camera-based colour-following mechanism*, where the system detects a specific colour—such as a red belt—in the camera frame. Based on its position, the cart decides whether to *move forward*, *turn left*, or *turn right*, ensuring accurate tracking of the customer. The second flowchart represents the *product scanning and checkout process*. It begins with scanning an item, checks if any item needs to be deleted, and then proceeds to *payment* and *cart closure*. These flowcharts visually convey the system's decision-making steps, enhancing clarity in both navigation and transaction workflows.


Start
Show the given Colour to camera to follow
Colour is in Mid of the frame
No
Yes
Colour at the Side of Frame
No
Yes
Left has less tolerence Right Trun
Move Forward
Right has more tolerence Left Turn
Continue following until the colour is out of frame
End

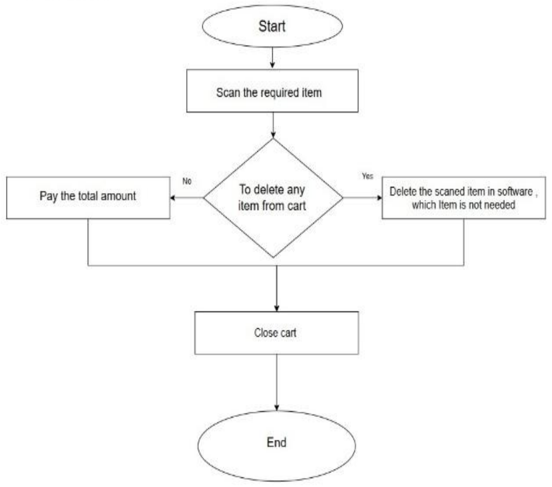

Start
Scan the required item
To delete any item from cart
No
Yes
Pay the total amount
Delete the scaned item in software , which Item is not needed
Close cart
End

Fig. Flow chart

IX. RESULTS OBTAINED

The smart shopping cart system successfully demonstrates its core functionalities across multiple domains. *Customer tracking* is achieved through *OpenCV's colour segmentation techniques*, allowing the cart to accurately detect and follow a customer wearing a red belt. The system processes video input in real time, ensuring smooth tracking as the customer moves throughout the store. To maintain safety, the *ultrasonic sensor* continuously measures the distance between the cart and the customer, adjusting speed or stopping as needed to prevent collisions. For *product scanning*, the *RFID scanner* efficiently reads item tags, enabling customers to add products to their virtual cart with ease. The system also supports item removal, offering flexibility in cart management. The *linked website* provides *real-time updates*, instantly reflecting additions or deletions and allowing users to review and manage their cart through an intuitive interface. Upon clicking *"End Shopping,"* the website calculates the total cost and initiates a secure transaction via an integrated *payment gateway*. The cart's movement is precisely controlled by the *L298N motor driver* and *12V 555 motors*, enabling responsive navigation and effective obstacle avoidance. Overall, the system enhances *automation* and *convenience*, minimizing manual effort and delivering a seamless shopping experience. Additionally, robust *error handling* mechanisms ensure stability in scenarios such as temporary loss of visual tracking or incorrect RFID scans, maintaining uninterrupted operation and user confidence.

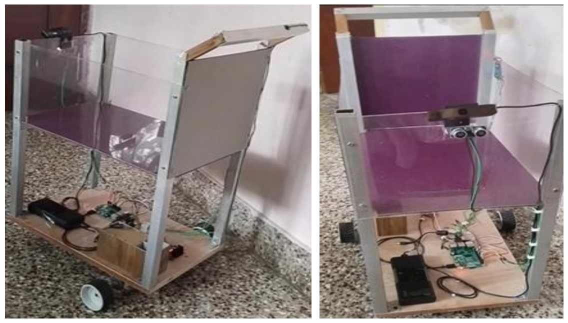


Fig. Pictorial of Smart shopping cart using OpenCV

X. BENEFITS TO SOCIETY

The smart shopping cart system offers significant *societal benefits* by enhancing accessibility, efficiency, and convenience in retail environments. By automating tasks such as *product scanning*, *customer tracking*, and *digital checkout*, the system reduces manual effort and minimizes wait times, especially benefiting *elderly* or *physically challenged* individuals. The integration of *IoT* and *computer vision* promotes *technological literacy* and encourages the adoption of *smart retail solutions*. Additionally, the system supports *contactless shopping*, contributing to *public health safety* in post-pandemic contexts. Its real-time data processing and web-based interface foster *transparency* and *user empowerment*, allowing customers to manage their purchases independently. Overall, the smart cart represents a step toward *inclusive*, *automated*, and *digitally connected* shopping experiences that align with the evolving needs of modern society.

XI. CONCLUSION

The smart shopping cart system presents a transformative approach to modern retail by integrating *computer vision*, *IoT*, and *embedded systems* into a single, user-friendly platform. Through the use of a *Raspberry Pi* as the central controller, combined with components such as the *camera module*, *ultrasonic sensor*, *RFID scanner*, and *motor driver*, the system achieves autonomous navigation, real-time product tracking, and seamless checkout functionality. The implementation of *OpenCV* for customer tracking and a responsive *web interface* for cart management ensures a smooth and efficient shopping experience. By automating key processes and minimizing manual intervention, the system enhances *convenience*, *accuracy*, and *safety* for users. Its potential to improve accessibility and reduce operational overhead makes it a valuable innovation in the retail sector, paving the way for smarter, more inclusive shopping environments.

XII. APPENDIX

APPENDIX A – API Endpoints

Details of the backend API used for cart operations:

End point	Method	Description
/add_product	POST	Adds a product to the cart
/remove_product	POST	Removes a product from the cart

/checkout	GET	Calculates total and initiates payment

APPENDIX B – Flowchart Logic Summary

This appendix summarizes the decision-making logic used in the system:
Camera-Based Tracking:
If color is centered → Move forward
If color is left/right → Turn accordingly
If no color detected → Stop
RFID-Based Cart Management:
Scan product → Add to cart
Delete product → Remove from cart
End shopping → Trigger checkout

APPENDIX C – Testing Protocols

Outlines the testing procedures followed:
Hardware Testing:
Verified GPIO pin outputs using LED indicators
Checked motor response to directional commands
Software Testing:
Debugged Python scripts in Mu Emulator
Validated OpenCV color detection accuracy
Web Testing:
API response validation using Postman
UI responsiveness tested across devices

ACKNOWLEDGEMENT

Firstly, We would like to express our profound gratitude to our project guide, Dr. Ushadevi M. B., Professor, Department of Electronics and Telecommunication Engineering, for her insightful suggestions, technical expertise, and continuous encouragement. Her guidance was instrumental in the successful execution of our project.
Finally, we express our gratitude to Jawaharlal Nehru New College of Engineering for providing us with all the essential infrastructure and academic support, without which this project would not have been possible.

REFERENCES

[1] Dr. Nagendra Kumar M., Nandini S., Priya C., Supriya N., and Varun Kumar K., "Smart Shopping Trolley for Automated Billing Process Using Image Processing," Proc. Int. Conf. on Smart Systems and Technologies, 2021.
[2] Mohd. Ashif Khan, Nikunj Bharadwaj, Nihal Kotwal, Yogesh Yadav, and Pragati Shrivastava Deb, "Automated Shopping Trolley Using Raspberry Pi Device," Int. J. Sci. Res. Eng. Trends, vol. 7, no. 4, pp. 112–117, 2021.
[3] IJARCCE Editorial Board, "Smart Car Parking System Using Raspberry Pi," Int. J. Adv. Res. Comput. Commun. Eng. (IJARCCE), vol. 10, no. 2, pp. 88–92, 2021.
[4] Priyanka S. Sahare, Anup Gade, and Jayant Rohankar, "A Review on Automated Billing for Smart Shopping System Using IoT," Int. J. Eng. Sci. Comput., vol. 9, no. 3, pp. 45–49, 2019.
[5] Academia.edu Contributors, "A Survey on Smart Trolley System Based on Android Application," [Online]. Available: https://www.academia.edu
[6] Raspberry Pi Foundation, "Raspberry Pi 4 Model B specifications," [Online]. Available: https:// www.raspberrypi.com/products/raspberry-pi-4-model-b/
[7] OpenCV Developers, "OpenCV: Open Source Computer Vision Library," [Online]. Available: https://opencv.org
[8] MFRC522 RFID Reader Datasheet, NXP Semiconductors, [Online]. Available: https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf
[9] L298N Motor Driver Module Datasheet, STMicroelectronics, [Online]. Available: https://www.st.com/en/motor-drivers/l298n.html
[10] HC-SR04 Ultrasonic Sensor Technical Manual, [Online]. Available: https://components101.com/sensors/hc-sr04-ultrasonic-sensor
[11] RealVNC Ltd., "Remote access software for Raspberry Pi," [Online]. Available: https:// www.realvnc.com/en/connect/download/vnc/
[12] Mu Code Editor, "A simple Python IDE for beginners," [Online]. Available: https://codewith.mu
[13] System Studios, "Web development platform for responsive design," [Online]. Available: https://systemstudios.com
[14] Flask Documentation, "Flask: Web development, one drop at a time," [Online]. Available: https://flask.palletsprojects.com
[15] Python Software Foundation, "Python 3 Programming Language," [Online]. Available: https://www.python.org.