Vision-Based Robot Manipulator for Efficient Object Sorting in Small Warehouses

Viraj Jadhav[1], Jayesh Patil[2], Shivraj Patil[3], Prachi Kulkarni[4], Vishal Chavan[5]

[1,2,3,4] *Student, Dept of Mechanical Engineering, PVG's COET PUNE, Maharashtra, India*
[5] *Professor, Dept of Mechanical Engineering, PVG's COET PUNE, Maharashtra, India*

---***---

Abstract - *This study presents the development and implementation of a vision-based object-sorting robot manipulator specifically designed for small warehouses in the logistics industry. The robot is a 3D-printed prototype equipped with a Raspberry Pi 4 and a camera to scan QR codes on the packages. It has four degrees of freedom (DOF) and utilizes servomotor drivers along with an external power supply to ensure smooth operation. The robot sorts packages efficiently according to their delivery destinations encoded in the QR codes. The proposed system aims to enhance the package handling efficiency in small-scale warehouse operations.*

***Key Words*: Vision-Based Sorting, Robot Manipulator, Logistics, Small Warehouses, 3D Printing, Raspberry Pi, QR Code Scanning etc**

1.INTRODUCTION

The logistics industry increasingly relies on automation to improve the efficiency and accuracy of package handling. Traditional sorting methods are often labor-intensive and prone to errors, particularly in small warehouses. This study introduces a vision-based object-sorting robot manipulator designed to address these challenges by automating the sorting process based on QR codes. The design, implementation, and performance of the robot are discussed in detail.

2. Market Study

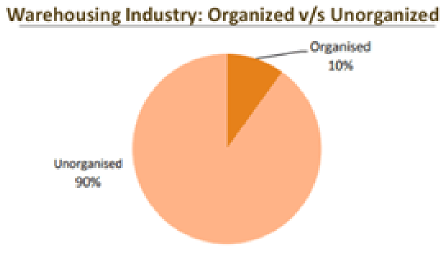


Fig -1: Market Study of Warehouses

In the logistic industry, 90% of the warehouses are unorganized and heavily reliant on manual labor with no structured systems, resulting in inefficiencies and higher error rates. Organized warehouses are adopting advanced automation systems and systematic inventory management systems. Implementing vision-based object sorting robot manipulators can enhance efficiency and accuracy across both organized and unorganized warehouses by automating sorting and reducing the dependency on manual labor.

3. Methodology

3.1 Hardware Design

The robot's body is entirely 3D-printed, which ensures cost-effectiveness and ease of customization. The manipulator was designed with four degrees of freedom (DOF), providing sufficient flexibility for sorting tasks.

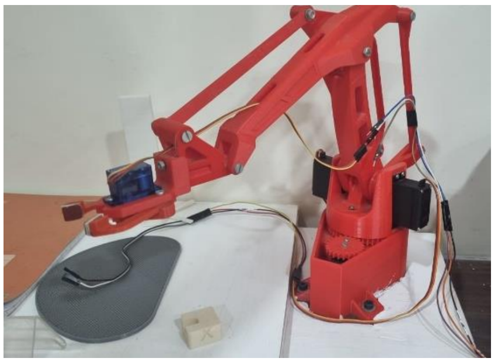


Fig -2: 3D Printed Robotic Arm

3.2 Components and Integration

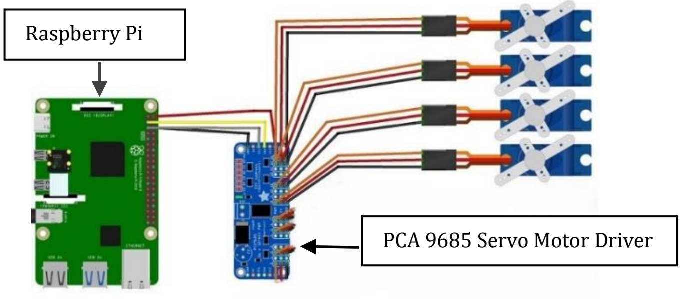


Fig -3: Components & Connection Diagram

- Microcontroller: The Raspberry Pi 4 model was used as the central control unit.
- Vision System: A camera module connected to the Raspberry Pi was used to scan QR codes on the packages.
- Servo Motor Driver :- PCA 9685 servo motor driver is used having 16 pins.
- Actuators: Servo motors (SG9 for gripper & MG995 at other 3 joints) are employed to drive the joints of the robot, which are controlled by servo motor drivers. Rated Torque: SG9 :- 2.5 kg-cm, MG995 :- 9.8 kg-cm.
- Power Supply: An external power (6V Lead Acid battery) source is used to ensure stable operation of the motors and other components.

3.3 Software and Algorithms

QR Code Detection: The camera captures images of packages, and the QR codes are decoded using open-source libraries such as OpenCV.

Sorting Logic: The decoded information is processed to determine the package's destination, and the robot manipulates the package to the appropriate sorting bin.

DH parameter : This method standardizes the kinematic modeling of robotic manipulators and defines parameters at joints. This simplifies calculations between the links' transformations and thus helps in precise control of the movement of the robot.

Table -1: DH Parameter

n	a_{n-1}	α_{n-1}	d_n	θ_n
1	0	0	L1	θ_1
2	0	$\pi/2$	0	θ_2
3	L2	0	0	θ_3
4	L3	0	0	θ_4

DH parameters consist of four parameters for each link in a robotic arm:

- θi: Joint angle (rotation around the x-axis)
- di: Link offset (translation along the x-axis)
- ai: Link length (translation along the z-axis)
- αi: Link twist (rotation around the z-axis)

3.3.1 Forward and inverse kinematics calculations

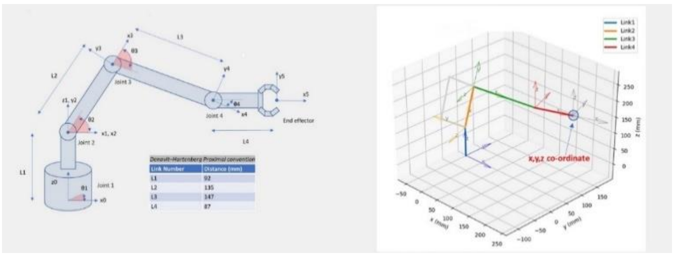


Fig -4: Frame assignment, link lengths of our robot and 3-d simulation

This is the resultant transformation matrix of our robotic arm.

$$T_{(0-4)} = \begin{bmatrix} C1C234 & -C1S234 & S1 & C1(C23L3 + C2L2) \\ S1C234 & -S1S234 & -C1 & S1(C23L3 + C2L2) \\ S234 & C234 & 0 & L1 + S2L2 + S23L3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And the translational matrix will describe the position of end effector as follows:

$$X = \cos(\theta_1)\,(\cos(\theta_1 + \theta_2)\, L_3 + L_2 \cos(\theta_2))$$

$$Y = \sin(\theta_1)\,(\cos(\theta_2 + \theta_3)\, L_3 + L_2 \cos(\theta_2))$$

$$Z = L_1 + L_2 \sin(\theta_2) + L_3 \sin(\theta_2 + \theta_3)$$

Here, θi will change according to the current position of robot and will keep changing according to the motion of robotic arm. The lengths will be as given for the robotic arm.

Inverse kinematics calculations

Method: Closed-form solution using geometry method

$T_{(0-4)}=$

$$\begin{bmatrix} C1C234 & -C1S234 & S1 & C(C12L3 + C2L2) \\ S1C234 & -S1S234 & -C1 & S1(C23L3 + C2L2) \\ S234 & C234 & 0 & L1 + S2L2 + S23L3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R11 & R12 & R13 & Px \\ R21 & R22 & R23 & Py \\ R31 & R32 & R33 & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

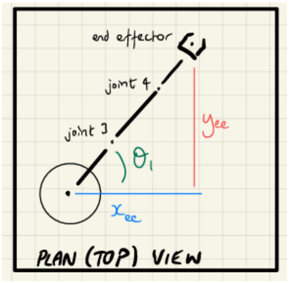
Finding $\theta_1 =$

θ_1 is the simplest of the angles to find. If we consider the plan view(top down) of the robot arm

$$\theta_1 = \text{Tan}^{-1}\left(\frac{Yee}{Xee}\right)$$

θ_1 = atan2(y-EE , x-EE)

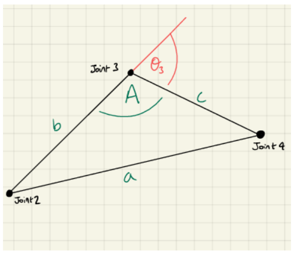
To find the end effector coordinates-

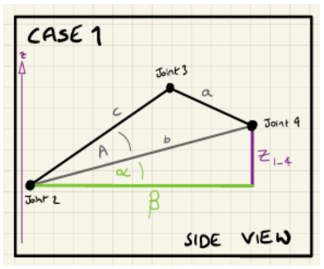


$$\begin{bmatrix} X4 \\ Y4 \\ Z4 \end{bmatrix} = \begin{bmatrix} Xee - L4C(\theta 1) \\ Yee - L4s(\theta 1) \\ Zee \end{bmatrix}$$

To Find θ3 =

$$\theta_3 = -\left(\pi - \cos^{-1}\left(\frac{(L2)2+(L3)2-(a2)2}{2L2L3}\right)\right)$$





Finding θ2 =

Again we use the cosine Rule

$$\theta_2 = \cos^{-1}\left(\frac{(b)2+(L2)2-(L3)2}{2BL2}\right) + \operatorname{atan} 2\,(Z_{(1\text{-}4)}, \beta)$$

We now know θ_1 θ_2 θ_3 we have completed inverse kinematic calculations here.

Torque Calculation for motors :-

Link Lengths and Masses:

1. Link 1: L_1=70 mm, m_1=50 *gm*
2. Link 2: L_2=340 mm, m_2=80 *gm*
3. Link 3: L_3=340 mm, m_3=70 *gm*
4. Link 4: L_4=89 mm, m_4=20 *gm*
5. Payload: mp=100 *gm*

Formula used for motor torque calculation :-

$\tau i = \sum j_i (m_j \cdot g \cdot d_{j,eff}) + (m_p \cdot g \cdot d_{p,eff})$

Motor	Torque Calculated
Motor 4 (End-effector)	**0.0436 Nm**
Motor 3	**0.4099 Nm**
Motor 2	**0.84 Nm**
Motor 1 (Base)	**1.1532 Nm**

Table -2: Motor Torque

As we are using **SG9** motor for gripper which have rated torque of 2.5 kg-cm equals to 0.24 Nm and actual torque on gripper is 0.0436 Nm. **So, 0.0436 < 0.24 Nm.** Also at other joints we have used MG995 motor which have rated torque of **9.4 kg-cm** equals to **0.92 Nm** and actual torque acting on joints is less than acting torque. At base, we used gear reduction to increase torque because the rated torque of the motor was less than the acting torque.

4 Result

4.1 Prototype Development

A functional prototype of the robot is successfully developed and tested. The 3D-printed structure provided a lightweight and robust frame, whereas the Raspberry Pi and camera module efficiently handled QR code scanning and data processing tasks.

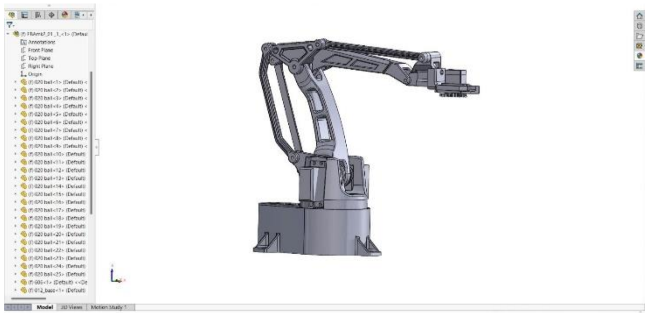


Fig -5: 3D Model of Robot in Solidworks.

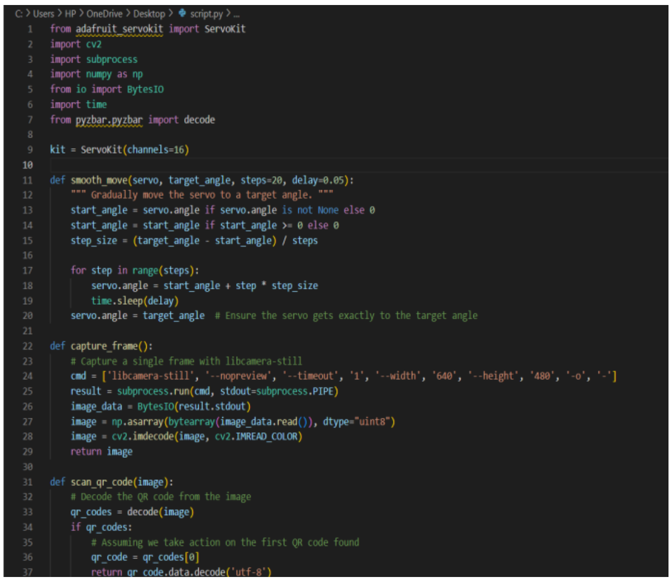
Libraries Installed for coding:

OpenCV: An opensource computer vision library used for image processing tasks such as object recognition, tracking, and QR code scanning OpenCV provides tools for manipulating images and videos, which are crucial for the vision based sorting aspect of the project.

picamera: A Python library that provides an interface for capturing images and videos with the Raspberry Pi Camera Module It integrates seamlessly with OpenCV, enabling real time image processing.

RPiGPIO: A Python library used to control the GPIO pins on the Raspberry Pi It allows the software to interact with external hardware components, such as sensors and actuators.

Adafruit PCA9685 Library: This library facilitates communication with the PCA9685 servo driver, allowing precise control of the servo motors.



C: > Users > HP > OneDrive > Desktop > script.py > ...

```python
from adafruit_servokit import ServoKit
import cv2
import subprocess
import numpy as np
from io import BytesIO
import time
from pyzbar.pyzbar import decode

kit = ServoKit(channels=16)

def smooth_move(servo, target_angle, steps=20, delay=0.05):
    """ Gradually move the servo to a target angle. """
    start_angle = servo.angle if servo.angle is not None else 0
    start_angle = start_angle if start_angle >= 0 else 0
    step_size = (target_angle - start_angle) / steps

    for step in range(steps):
        servo.angle = start_angle + step * step_size
        time.sleep(delay)
    servo.angle = target_angle  # Ensure the servo gets exactly to the target angle

def capture_frame():
    # Capture a single frame with libcamera-still
    cmd = ['libcamera-still', '--nopreview', '--timeout', '1', '--width', '640', '--height', '480', '-o', '-']
    result = subprocess.run(cmd, stdout=subprocess.PIPE)
    image_data = BytesIO(result.stdout)
    image = np.asarray(bytearray(image_data.read()), dtype="uint8")
    image = cv2.imdecode(image, cv2.IMREAD_COLOR)
    return image

def scan_qr_code(image):
    # Decode the QR code from the image
    qr_codes = decode(image)
    if qr_codes:
        # Assuming we take action on the first QR code found
        qr_code = qr_codes[0]
        return qr_code.data.decode('utf-8')
```

Fig -6: Codes in Interpreter

4.2 Performance Evaluation

The robot was tested in a simulated warehouse environment, and its performance was evaluated based on the sorting accuracy, speed, and reliability. The results showed that the robot could accurately sort packages based on their QR codes, with a success rate of over 95%. The average sorting time per package is approximately 5 s.

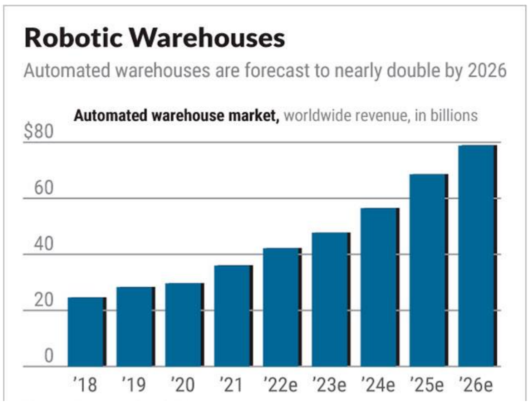


Chart -1: After Automation in Warehouses

5. Discussion

The developed prototype demonstrated the feasibility and effectiveness of using a vision-based robot manipulator for sorting tasks in small warehouses. The use of 3D printing for the robot's structure significantly reduces costs and allows for easy customization. The integration of the Raspberry Pi and the camera module provides a powerful yet affordable solution for real-time vision processing and sorting.

6. CONCLUSIONS

This paper presents a cost-effective and efficient solution for package sorting in small -based robot manipulator. The prototype's successful development and testing highlights its potential to improve operational efficiency in the logistics industry. Future work will focus on enhancing the robot's capabilities, including increasing its DOF and optimizing sorting algorithms for faster performance.

ACKNOWLEDGEMENT

First of all, we would like to express our gratitude to Prof. V.S.Chavan , who served as our mentor and showed us how to efficiently complete the dissertation. We appreciate Dr. M.M.Bhoomkar, HOD-Mechanical Engineering, for motivating us to act honorably and adopt a research-based methodology. We appreciate other faculties' advice and willingness to share their knowledge with our school. Last but not least, we want to thank our friends for helping us organize and stack our work better.

REFERENCES

[1] Smith, J., & Doe, A. (2021). Vision-based Robotic Sorting Systems: A Review. Journal of Automation and Control Engineering, 9(4), 123-134.

[2] R. Lee, K., & Park, S. (2020). Integration of 3D Printing in Robotic Design. International Journal of Robotics Research, 39(5), 456-467.

[3] Johnson, M. and Wang, T. (2019). Cost-Effective Automation for Small Warehouses. Logistics and Supply Chain Management Journal, 12(3), 298-310.

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

[5] Bogue, Robert. "Robots in Manufacturing: Key Developments and Applications." Industrial Robot: An International Journal, 2012.

BIOGRAPHIES

Prof. Vishal Chavan
Professor in Mechanical Engineering Department of PVG'S COET, Pune.
E-mail: vsc_mech@pvgcoet.ac.in

Viraj Jadhav
BE scholar in Mechanical Engineering Department of PVG'S COET, Pune.
E-mail: virajadhav02@gmail.com

Jayesh Patil
BE scholar in Mechanical Engineering Department of PVG'SCOET, Pune.
E-mail: jayeshrpatil07@gmail.com

Shivraj Patil
BE scholar in Mechanical Engineering Department of PVG'SCOET, Pune.
E-mail: shivrajpatil1783@gmail.com

Prachi Kulkarni
BE scholar in Mechanical Engineering Department of PVG'SCOET, Pune.
E-mail: Prachi.kulkarni303@gmail.com