ECE445 SP26

Senior Design Report

Machine Vision-Based Intelligent Fruit and Vegetable Picking & Sorting Robotic Arm

Team #17

Simeng Yan [simengy2]

Wenye Zhang [wenyez2]

Fengyi Jin [fengyij2]

Shengyu Xu [sx27]

1 Introduction

1.1 Purpose

In the contemporary agricultural value chain, the post-harvest sorting of produce, such as fruits and vegetables, remains a critical bottleneck that dictates marketability and shelf-life. Currently, small-to-medium scale facilities rely heavily on manual labor to categorize products based on size, ripeness, and quality. However, this human-centric approach suffers from high operational overhead, subjective inconsistency, and physical fatigue, leading to mis-sorting rates of up to 15% during peak hours.

The purpose of this project is to develop a Machine Vision-Based Intelligent Fruit and Vegetable Picking & Sorting Robotic Arm. By replacing error-prone human vision with high-speed computer vision and manual handling with precise servo-driven actuation, this system provides a low-cost, versatile solution for automation in unstructured agricultural environments, ensuring 24/7 operational stability and data-driven quality control.

1.2 Functionality

The system operates as a fully autonomous closed-loop pipeline, transforming raw visual data into precise physical sorting actions through a seamless "perception-to-execution" workflow. Initially, the vision system performs Real-Time Object Discrimination, utilizing color-based segmentation and deep learning inference (YOLOv8-nano) to distinguish between different produce types—such as separating red tomatoes from green fruits—and assigning high-priority tracking to the user-defined target. Once identified, the system executes Dynamic Visual-to-Spatial Mapping, where a pre-calibrated "eye-to-hand" transformation matrix maps 2D image centroids into the robot’s 3D Cartesian workspace coordinates (X, Y, Z). This triggers

the Optimized Path Planning stage, during which an Inverse Kinematics (IK) engine solves the six necessary joint angles and generates a smooth trapezoidal velocity profile to ensure stable movement. The pipeline concludes with Adaptive "Soft-Touch" Grasping, as the 6-DOF robotic arm descends to the target, secures it with calibrated torque via the serial bus protocol, and transports it to a predefined sorting bin for final placement.

Real-Time Object Discrimination & Target Selection: The vision system utilizes a color-based segmentation and deep learning inference (YOLOv8-nano) to distinguish between different types or colors of produce (e.g., separating red tomatoes from green apples). The user can define a specific target class or color, and the system will filter out non-target objects, assigning high-priority tracking to the desired item.

Dynamic Visual-to-Spatial Mapping: Once a target is identified, the system calculates its 2D centroid in the image frame. Through a pre-calibrated "eye-to-hand" transformation matrix, these coordinates are mapped into the robot’s 3D Cartesian workspace (X, Y, Z). This ensures the arm can precisely approach the target regardless of its position on the sorting plane.

Motion Execution: The control subsystem triggers an Inverse Kinematics (IK) engine to solve the six joint angles required to reach the target. The motion is executed using a smooth trapezoidal velocity profile to prevent jerky movements that might damage fragile produce or cause the arm to lose steps.

Adaptive "Soft-Touch" Grasping & Sorting: The actuation system executes a multi-stage sequence: Moving to a hover position above the target. Lowering the 6-DOF arm and closing the gripper with calibrated torque via the serial bus protocol. Lifting the object and following a collision-free path to a predefined sorting bin,

where the gripper releases the item to complete the pipeline.

1.3 Subsystem Overview

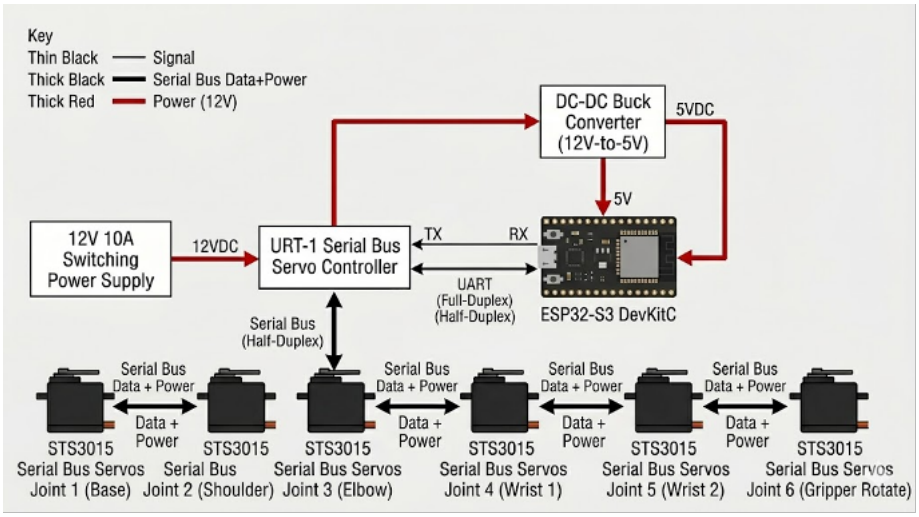


Figure 2. Subsystem hardware overview.

The system is composed of four primary interconnected subsystems that enable the end-to-end automation flow:

1.3.1 Perception Subsystem:

Hardware: A Logitech C270i USB camera supplemented by an adjustable LED ring light to mitigate image noise in low-light conditions.

Software: Implements a lightweight YOLOv8-nano model for object recognition, combined with Zhang's calibration method and Kalman filtering for stable localization.

1.3.2 Actuation Subsystem:

Hardware: Comprises six Feetech STS3215 serial bus servos. These servos provide high torque (up to 19.3kg.cm) and integrate 12-bit magnetic encoders for absolute position feedback.

Communication: Utilizes a Daisy Chain topology via a single-wire asynchronous serial bus protocol to minimize wiring complexity.

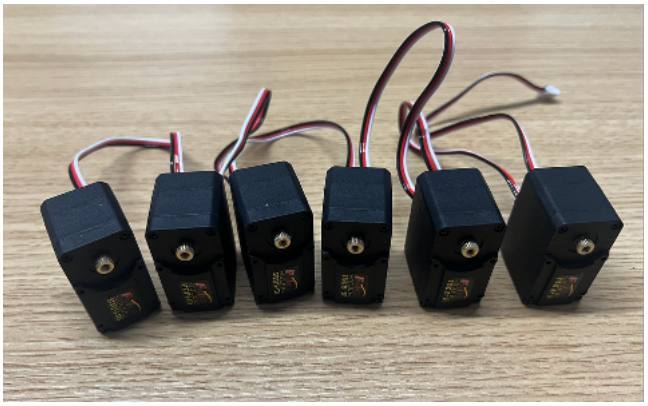


Figure 3. Actuation subsystem wiring (STS3215 daisy-chain bus).

1.3.3 Control Subsystem:

Processing: An ESP32-S3 DevKitC serves as the central processor, leveraging its dual-core architecture to handle IK calculations and external communication simultaneously.

Signal Bridging: A URT-1 Serial Bus Controller converts full-duplex UART signals from the ESP32 to the half-duplex protocol required by the servos.

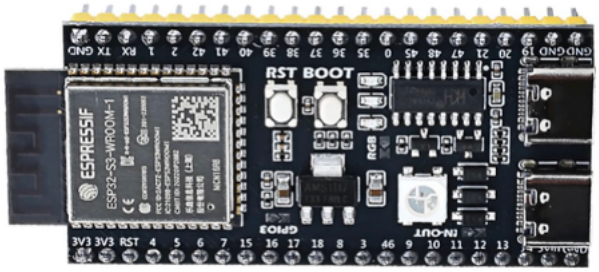


Figure 4. ESP32-S3 DevKitC central controller.

Figure 5. URT-1 serial bus bridge.

1.3.4 Power Subsystem:

Supply: A 12V 10A switching power supply provides 120W of power to meet the peak current demands of the six servos under load.

Regulation: A split-rail architecture uses DC-DC buck converters to provide a clean 5V supply to the ESP32, preventing motor back-EMF from interfering with logic circuitry.

Figure 6. 12V / 10A switching power supply.

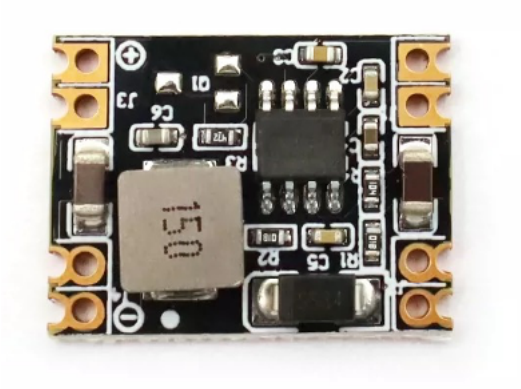


Figure 7. DC-DC buck converter (12V → 5V logic rail).

Figure 1 below summarizes how the four subsystems are connected. Visual data flows from the perception block into the ESP32-S3 controller, which converts target coordinates into joint commands and drives the six STS3215 servos through the URT-1 bus bridge. The power subsystem feeds two isolated rails: a 12 V high-current rail for the servos and a regulated 5 V rail for the logic side. Encoder and current telemetry from each servo is returned to the controller to close the loop.

Figure 1. Overall System Block Diagram

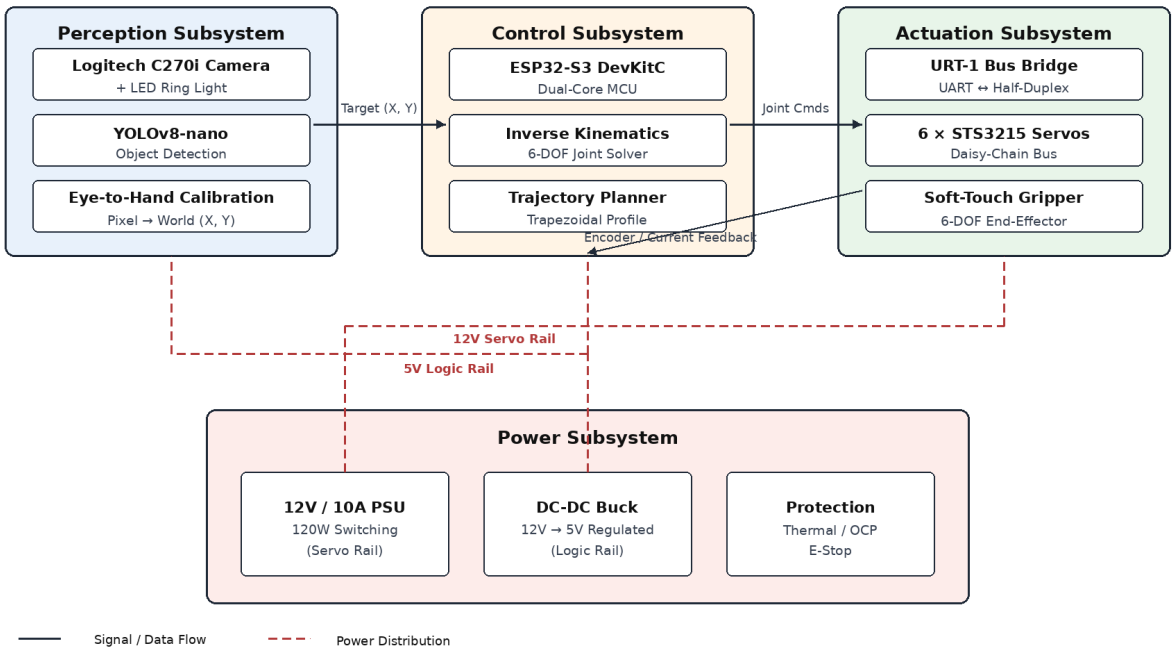


Figure 1. Overall system block diagram showing perception, control, actuation, and power subsystems with signal and power flow.

2 Design

2.1 Design Model and Principle

2.1.1 System Modeling Overview

This section introduces the major mathematical models and simulation targets used in the project. The proposed system is a machine vision-based fruit and vegetable picking and sorting robotic arm. Its overall workflow follows a Sense–Think–Act structure: the vision subsystem detects the target object, the control subsystem performs coordinate transformation and motion planning, and the robotic arm executes grasping and sorting actions.

The modeling work in this chapter mainly includes four components. First, a servo angle control model is developed to describe the relationship between control commands and joint angles. Second, a robotic arm kinematic model is used to analyze the position and reachable workspace of the 6-DOF manipulator. Third, a vision coordinate mapping model is established to convert image-space target coordinates into physical coordinates in the robotic arm workspace. Fourth, gripper force and load models are used to estimate the soft gripper's grasping capability and the torque requirements of the servo joints.

Since the objective of this project is to develop a functional laboratory prototype rather than a fully industrial robotic platform, simplified but engineering-valid models are adopted. A 2D or simplified 3D kinematic model can be used to verify whether the shoulder, elbow, and wrist joints can reach the target grasping area. For vision mapping, affine transformation or linear calibration can be used to convert the target center in the camera image into the robot base coordinate system. For gripper and load estimation, static force analysis can be used to estimate the joint torques under worst-case arm postures.

2.1.2 Servo Command and Angel Control Model

This project uses STS3215 serial bus servos as the main actuator for the 6-DOF

robotic arm. Unlike traditional PWM servos, the STS3215 receives position, velocity, and load-related commands through a serial bus and can return feedback such as position, voltage, current, and temperature. Therefore, this section should not be limited to a simple PWM-to-angle model. Instead, it should be titled Servo Command and Angle Control Model, where the PWM model is introduced as a general servo control principle, and the digital position command model is used for the actual serial bus servo implementation.

$$\theta = \theta_{\min} + \frac{PWM - PWM_{\min}}{PWM_{\max} - PWM_{\min}}(\theta_{\max} - \theta_{\min})$$

Figure 8. PWM-to-angle servo control model schematic.

Where θ is the target servo angle, PWM is the input pulse width, and θ min and θ max are the minimum and maximum achievable angles.

For the STS3215 serial bus servo used in this project, the relationship between the digital position command and the physical angle can be modeled as:

where P_{cmd} is the target position command and P_{max} is the maximum encoder count.

Thus, the theoretical angular resolution is:

$$\Delta\theta = \frac{360^{\circ}}{4096} = 0.088° \tag{1}$$

This resolution is sufficient for fine end-effector positioning and gripper opening control. Based on the feedback capability of the STS3215 described in the instruction manual, position, current, and temperature feedback can also be used to implement closed-loop control and safety protection.

2.1.3 Robotic Arm Kinematics and Workspace Analysis

This section develops the kinematic model of the robotic arm to analyze the relationship between joint angles, link lengths, and the end-effector position. Since the project uses a 6-DOF serial robotic arm, the complete three-dimensional kinematics can be described using the Denavit–Hartenberg parameter method. Howevera simplified 2D planar model can first be introduced to explain the dominant influence of the shoulder and elbow joints on the end-effector position.

In the simplified 2D model, the arm is assumed to move mainly in a vertical plane. Let L_1 be the upper arm length, L_2 be the forearm length, θ_1 be the shoulder angle, and θ_2 be the elbow angle. The end-effector position can be expressed as

$$x = L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \tag{2}$$

$$y = L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \tag{3}$$

Note: in this convention θ_1 is measured from the vertical, so both x and y components share a sine form. Switching to a horizontal reference would replace the second sine with cosine.

This model can be used to determine whether the robotic arm can reach the target object. If the target point satisfies:

$$| L_1 - L_2 | \leq \sqrt{x^2 + y^2} \leq L_1 + L_2 \tag{4}$$

then the target lies within the theoretical reachable workspace of the arm.

For the full 6-DOF system, the robotic arm can be modeled as a rigid-body chain composed of multiple revolute joints. Each joint corresponds to a coordinate transformation matrix:

$$T_i^{i-1} = \begin{bmatrix} R_i^{i-1} & d_i^{i-1} \\ 0 & 1 \end{bmatrix}$$

(5)

The total transformation from the base to the end-effector is:

$$T_0^6 = T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_5^6$$

(6)

where T_0^6 contains both the position and orientation of the end-effector. Since the robot arm we build is a 6-axis serial joint layout including the waist, shoulder, elbow, and wrist joints, this model directly corresponds to the actual mechanical structure.

2.1.4 Inverse Kinematics and Target Reaching Estimation

In the robotic grasping task, the vision system provides the target position in the workspace, while the control system must convert this position into target angles for each joint. Therefore, inverse kinematics is a key part of the Vision-to-Motion Pipeline.

For a simplified 2D robotic arm, given the target point (x, y), the distance from the base to the target is:

$$r = \sqrt{x^2 + y^2}$$

(7)

Using the law of cosines, the elbow angle can be calculated as:

$$\cos(\theta_2) = \frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}$$

$$\theta_2 = \cos^{-1}\left(\frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}\right)$$

(8)

The shoulder angle can be expressed as:

$$\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{L_2 \sin\ \theta_2}{L_1 + L_2 \cos\ \theta_2}\right) \tag{9}$$

This inverse kinematic model can be used to determine whether the target is reachable and to calculate the initial joint angles required for grasping. For the actual 6-DOF robotic arm, base rotation, wrist orientation, and gripper approach angle can be added so that the end-effector not only reaches the target point but also approaches the produce surface from a suitable direction.

Finaly, the inverse kinematic results are sent to the ESP32-S3 controller and then transmitted through the URT-1 serial bus interface to synchronously control multiple STS3215 servos. Since we have already emphasizes synchronized motion control and the Vision-to-Motion Pipeline, this section provides the mathematical basis for that implementation.

2.1.5 Vision Coordinate Mapping and Calibration

This project adopts an Eye-to-Hand fixed-camera configuration, where the camera is mounted externally and observes the workspace from above. The advantage of this configuration is that the transformation between the camera coordinate system and the robotic arm coordinate system remains nearly constant after calibration, which reduces real-time computational complexity. The we have already states that the fixed-camera strategy provides a global view of the workspace and avoids adding payload to the end-effector.

The main task of the vision system is to detect the target object and extract its center coordinates. If the bounding box output from YOLOv8n is:

$$B = (x_{min}, y_{min}, x_{max}, y_{max}) \tag{10}$$

then the pixel coordinates of the target center are:

$$u_c = \frac{x_{min} + x_{max}}{2}$$
$$v_c = \frac{y_{min} + y_{max}}{2}$$

(11)

To convert the image coordinates (u_c, v_c) into the robotic arm workspace coordinates (X, Y), a linear mapping model can be used:

$$X = au_c + b$$
$$Y = cv_c + d$$

(12)

where a, b, c, d are mapping parameters obtained through experimental calibration.

For a more accurate transformation from the image plane to the workspace plane, an affine transformation model can be used:

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

(13)

where (u, v) are image coordinates and (X, Y) are target coordinates in the robotic arm base frame. This model can correct translation, scaling, and rotation errors between image coordinates and actual grasping positions.

During the experiment, several calibration points with known physical coordinates can be placed on the workspace. Their pixel coordinates are recorded, and the mapping matrix is solved using the least-squares method. After calibration, test points can be used to calculate the average localization error:

$$e = \sqrt{(X_{pred} - X_{true})^2 + (Y_{pred} - Y_{true})^2}$$

(14)

This error should be smaller than the allowable tolerance of the gripper to ensure successful grasping.

2.2 Design Alternatives

2.2.1 Motor Selection Alternatives

This section compares different actuator options. The comparison that we would discuss focused on the **DMJ4310 brushless modular motor** and the **STS3215 serial bus servo**.

The DMJ4310 provides high torque density and good dynamic performance, making it suitable for advanced robotic platforms. However, it is expensive and requires a more complex CAN-based control system.

The STS3215 integrates the motor, reduction gearbox, encoder, and control circuit into one compact unit. It supports serial bus communication and provides feedback such as position, voltage, current, and temperature. It also reduces wiring complexity and cost, which makes it more suitable for a senior design prototype. Therefore, the STS3215 was selected as the final actuator solution.

2.2.2 Controller Selection Alternatives

This section compares possible controller options, Arduino Uno, STM32, and ESP32-S3.

Arduino Uno is easy to use but has limited computational capability and communication resources, making it unsuitable for multi-servo control and high-speed communication.

STM32 offers strong performance but has a higher development complexity and requires more low-level configuration.

ESP32-S3 provides a dual-core processor, high clock speed, multiple UART interfaces, and Wi-Fi/Bluetooth capability. It is suitable for robotic arm control, serial bus communication, and future wireless debugging. Therefore, ESP32-S3 was selected as the main controller.

2.2.3 Vision Strategy Alternatives

This section compares two vision configurations: Eye-to-Hand and Eye-in-Hand**.**

In the Eye-to-Hand configuration, the camera is fixed outside the robotic arm and observes the workspace from above. Its advantages include a stable field of view, the ability to detect multiple objects at once, no added end-effector payload, and a fixed calibration matrix. Its main disadvantage is possible visual occlusion when the arm blocks the camera view.

In the Eye-in-Hand configuration, the camera is mounted on the end-effector. It provides close-up visual information, but it increases payload and inertia, reduces dynamic performance, and requires coordinate transformations to be updated continuously based on the arm posture.

As a result of that, we finally chose the Eye-to-Hand fixed-camera configuration.

2.2.4 Communication Protocol Alternatives

This section compares traditional PWM control and serial bus control.

Traditional PWM servos require a separate signal wire for each servo. For a 6-DOF robotic arm, this creates complicated wiring and increases the risk of cable fatigue or entanglement at the joints. PWM communication also usually does not provide direct feedback from the servo.

Serial bus control allows multiple servos to be connected through a three-wire daisy-chain structure. Each servo is addressed by a unique ID. This reduces wiring complexity and supports bidirectional feedback, including position, current, voltage, and temperature. In this project, the URT-1 board converts the ESP32-S3 UART signal into the half-duplex serial bus signal required by the STS3215 servos.

2.3 Design Description and Justification

The final design of the project is an integrated machine vision-based robotic sorting system that combines visual perception, coordinate mapping, motion planning, serial-bus actuation, and safe power distribution into one closed-loop platform. Unlike a simple manually controlled robotic arm, the proposed system is designed to complete the full process of target detection, position estimation, grasping, transportation, and sorting without direct human intervention. The design is centered on a fixed-camera 6-DOF robotic arm architecture, where the camera provides global workspace information and the robotic arm executes pick-and-place operations according to the processed target coordinates.

The overall design follows a modular structure. The perception module is responsible for detecting the target fruit or vegetable and extracting its image-space position. The control module receives the target location, converts it into robot workspace coordinates, calculates the required joint commands, and sends motion instructions to the actuators. The actuation module physically moves the robotic arm and gripper to complete the sorting task. The power module provides separated and stable voltage rails for the high-current servo system and the low-voltage logic system. This modular design improves debugging efficiency, allows each subsystem to be tested independently, and reduces the risk that one subsystem failure will affect the entire platform.

2.3.1 Overall Mechanical and Functional Design

The robotic platform uses a 6-DOF serial arm structure, including base rotation, shoulder motion, elbow motion, wrist adjustment, and gripper actuation. This configuration was selected because fruit and vegetable sorting requires not only reaching a target position but also approaching the object from a suitable direction. A lower-degree-of-freedom arm could perform simple horizontal picking, but it would have limited flexibility when the object position changes or when the gripper needs to avoid collision with nearby objects. The 6-DOF structure provides sufficient motion freedom for a laboratory-scale sorting task while remaining affordable and relatively easy to assemble.

The working area is designed as a fixed sorting plane observed by an overhead or external camera. Objects are placed within the camera field of view, and the arm operates inside a predefined reachable workspace. This design reduces uncertainty because the target height and operating plane can be assumed to be approximately constant. As a result, the system can focus mainly on accurate XY localization and stable grasping, which is appropriate for a senior design prototype.

The gripper is designed as a soft-touch end-effector rather than a rigid clamp. Fruits and vegetables are easily damaged by excessive gripping force, so the mechanical design prioritizes compliant contact and controlled closing motion. The gripper does not need to maximize force; instead, it needs to apply enough pressure to lift the object without bruising or slipping. This design choice is consistent with the goal of building a sorting system for fragile produce rather than a general-purpose industrial manipulator.

2.3.2 Vision-to-Motion Pipeline Design

The key design feature of the system is the vision-to-motion pipeline. After the camera captures the workspace image, the vision algorithm identifies the target object and determines its center position in pixel coordinates. These image coordinates are

then transformed into physical coordinates in the robot base frame through calibration. The control system uses the transformed coordinates as the input for motion planning and inverse kinematics calculation.

This pipeline was chosen because it directly connects perception results with robotic execution. Instead of manually entering object positions or using fixed pre-programmed paths, the system can respond to different target locations in real time. This makes the design more suitable for unstructured agricultural environments, where the position and orientation of produce are not always fixed.

A fixed eye-to-hand camera configuration is used in the final design. This configuration is justified by three reasons. First, it gives the system a global view of the workspace, allowing multiple objects to be detected before motion starts. Second, it avoids adding extra payload to the end-effector, which is important because the selected servo motors have limited torque margins. Third, once calibrated, the camera-to-robot transformation remains stable, which simplifies the control algorithm and improves repeatability. Although arm occlusion may occur when the manipulator blocks the camera view, this limitation can be reduced by capturing the target position before the arm enters the grasping area.

2.3.3 Actuation and Control Design

The actuation system is built around STS3215 serial bus servos. Each servo integrates position control, feedback sensing, and communication capability, which simplifies the mechanical and electrical implementation of the robotic arm. Compared with traditional PWM servos, the serial bus design allows multiple motors to share a common communication line and be addressed by individual IDs. This reduces wiring complexity, especially around moving joints, and makes the arm easier to assemble and maintain.

The ESP32-S3 is selected as the main controller because it provides sufficient processing capability for real-time robotic control while maintaining low cost and

compact size. In the final design, the ESP32-S3 handles coordinate processing, inverse kinematics calculation, motion command generation, and communication with the serial bus controller. The URT-1 interface board is used to bridge the ESP32 UART signal with the half-duplex serial bus required by the servos. This division of responsibility allows the microcontroller to focus on control logic while the interface board handles communication-level compatibility.

The motion control strategy is designed to avoid sudden movements. Instead of directly commanding abrupt position changes, the arm follows a staged motion sequence: moving to a safe hover position, descending toward the target, closing the gripper, lifting the object, moving to the sorting bin, and releasing the object. This sequence reduces collision risk and improves grasping stability. It also makes the system easier to debug because each stage can be tested separately.

2.3.4 Power and Safety Design

The power system uses a separated power architecture. The servos are powered by a 12V supply capable of providing high current during peak load, while the ESP32-S3 and logic components are powered through regulated 5V conversion. This separation is necessary because servo motors can generate current spikes and voltage drops during acceleration, stall, or sudden load changes. If the controller shared the same unstable supply rail without regulation, the microcontroller could reset or behave unpredictably.

The use of DC-DC buck converters improves voltage stability for the logic system. At the same time, the power system is designed with enough current margin to support simultaneous servo movement. This is especially important for a robotic arm because several joints may experience high torque at the same time during lifting. The power design therefore supports both functional performance and system reliability.

Safety is also included in the design. The software can monitor servo-related feedback such as temperature, voltage, and load where available. If abnormal values are

detected, the controller can stop motion or disable torque to protect the hardware. Mechanical safety is also considered by limiting the operating workspace and using predefined motion stages, so the arm does not move randomly or exceed its intended range.

2.3.5 Justification of the Integrated Design

The final integrated design is justified by the project's performance requirements, budget constraints, and prototype nature. The selected architecture does not aim to compete with industrial robotic arms in speed or payload capacity. Instead, it emphasizes feasibility, low cost, modularity, and reliable demonstration of autonomous sorting. The combination of a fixed camera, ESP32-S3 controller, STS3215 serial bus servos, and soft gripper provides a balanced solution for a senior design project.

The fixed-camera vision system reduces mechanical complexity and supports stable object localization. The serial bus servo system reduces wiring and enables feedback-based protection. The ESP32-S3 provides enough control capability without requiring a complex embedded computing platform. The separated power design improves reliability and prevents motor noise from affecting the control electronics. Together, these design choices form a practical and coherent system.

Overall, the design meets the main goal of the project: to demonstrate an autonomous fruit and vegetable picking and sorting robotic arm using machine vision and servo-based manipulation. The design choices are not only technically reasonable but also appropriate for the available budget, development time, and testing environment. Future improvements, such as depth cameras, stronger actuators, or industrial-grade mechanical parts, can be added based on the same modular framework without changing the fundamental system architecture.

2.4 Subsystem Diagrams &Schematics

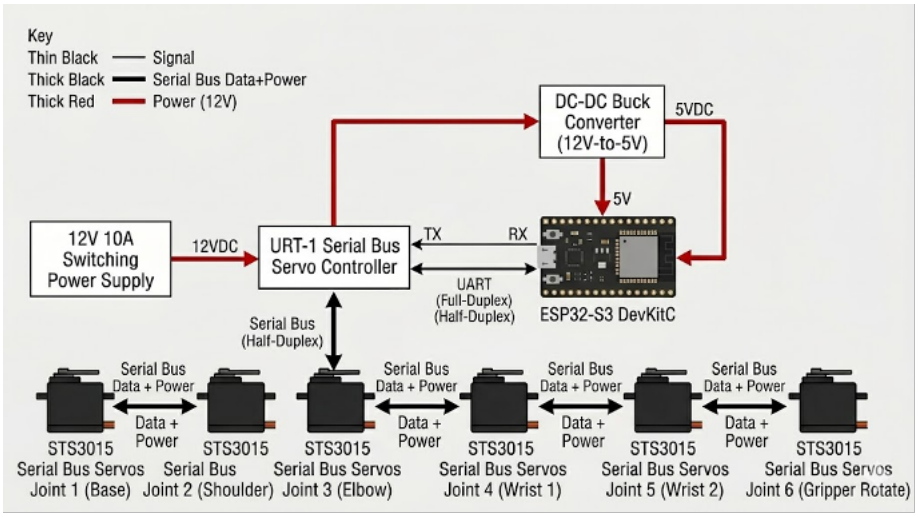


Figure 9. Integrated subsystem block layout.

This section summarizes the major subsystem diagrams and schematics used to describe the structure of the machine vision-based fruit and vegetable picking and sorting robotic arm. The diagrams are intended to support the system design by showing the relationship among the perception subsystem, control subsystem, actuation subsystem, and power subsystem. Instead of presenting each component independently, this section explains how the subsystems are connected and how they work together to complete the full pick-and-sort process.

The overall system diagram shows that the project follows a closed-loop workflow from visual sensing to robotic execution. The camera first captures images of the working area, and the vision program detects the target fruit or vegetable. After the object is identified, its position in the image is converted into a physical position in the robotic arm workspace. The controller then calculates the required joint movements and sends commands to the servo motors. Finally, the robotic arm moves to the target, grasps it, transfers it to the sorting area, and releases it.

The perception subsystem diagram focuses on image acquisition and target detection. In this subsystem, the camera and lighting module provide stable visual input for the

recognition algorithm. The target object is detected based on its visual features, such as color, shape, or trained object class. The output of this subsystem is not the final sorting action, but the target location information that will be used by the control subsystem. Therefore, the perception subsystem serves as the starting point of the entire automation pipeline.

The control subsystem schematic describes the connection between the vision result and the robotic motion. After receiving the target coordinates, the controller performs coordinate transformation and inverse kinematics calculation. The transformed position is converted into joint angle commands for the robotic arm. The controller also manages the motion sequence, including moving to a safe position above the object, lowering the gripper, grasping the object, moving to the sorting bin, and returning to the home position. This design makes the motion process more stable and easier to debug.

The actuation subsystem diagram represents the physical movement of the robotic arm. The six servo motors form a serial robotic structure, allowing the arm to rotate, extend, adjust its wrist orientation, and operate the gripper. The selected serial bus servos reduce wiring complexity because multiple servos can share the same communication bus. In addition, servo feedback can be used to monitor the operating status of the arm, which improves the reliability of the system during repeated pick-and-place operations.

The power subsystem schematic shows how electrical power is distributed to different parts of the system. Since the servo motors require relatively high current during movement, they are powered separately from the logic control circuit. A 12V power supply provides energy for the servos, while a DC-DC converter provides a stable low-voltage supply for the controller and communication interface. This separated power design reduces the risk of voltage drops, electrical noise, and unexpected controller resets during servo operation.

Overall, the subsystem diagrams and schematics demonstrate that the final design is modular and logically organized. Each subsystem has a clear responsibility: the perception subsystem detects the object, the control subsystem makes motion decisions, the actuation subsystem executes the movement, and the power subsystem supports stable operation. This structure improves system reliability, simplifies troubleshooting, and provides a clear foundation for later testing and verification.

Figure 10 details the data flow inside the closed-loop pipeline. The six processing stages execute in sequence each cycle: image acquisition, object detection, centroid extraction, coordinate mapping, inverse kinematics, and servo execution. The pipeline is bounded above by the camera capture rate (~18.5 FPS, see Section 4.1) and below by the IK computation latency (~42 ms, see Section 4.3), giving comfortable margin for real-time tracking.

Figure 10. Vision-to-Motion Pipeline Workflow

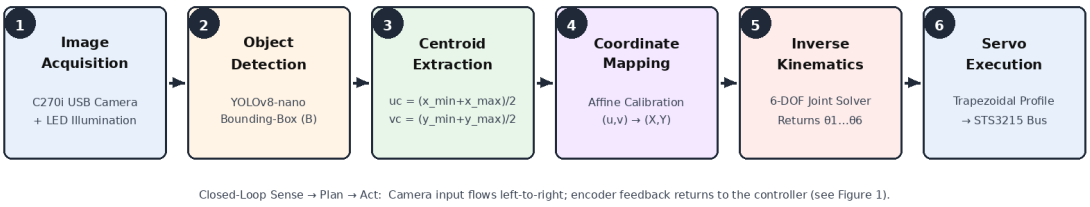


Figure 10. Vision-to-motion pipeline workflow showing the six processing stages of each pick-and-sort cycle.

3.Cost & Schedule

Cost

Table 1. Bill of materials and unit costs.

Components	Quantity	Price (rmb)
Servo Motor	9 (1 for spare, 2 broken)	986
Esp32-S3	1	28.31

URT-1	1	50
NVVV Power Supply	1	37.68
Plug	1	8.5
DC-DC inverter	2 (1 for spare)	9.2
Breadboard set	1	17.18
Camera	1	100
Total	—	**1236.87 (≈ $171.79 USD)**

Schedule

Table 2. 8-week project development schedule by team member.

Week	Wenye Zhang (Vision & Motion)	Simeng Yan (Motion & Electronics)	Fengyi Jin (Power Systems & Electronics)	Shengyu Xu (Mechanical Design)
1	Research YOLOv8 and setup Python/OpenCV environment.	Research STS3215 servo protocols and ESP32 SDK.	Sourcing AC-DC power supply and battery components.	Initial CAD drafting of the robotic arm joints.
2	Image data collection and labeling for color-based sorting.	Implementing serial bus communication for servo feedback.	Designing the 12V-to-5V power distribution PCB.	3D printing of the base and primary link structures.
3	Training YOLOv8-nano and optimizing inference speed.	Developing Forward Kinematics (FK) models in	Soldering the power shield and stress testing DC-DC	Assembling the arm links and testing structural

		firmware.	buck.	rigidity.
4	Implementing "Eye-to-Hand" calibration (Zhang's method).	Developing the Inverse Kinematics (IK) engine for XYZ.	Integrating URT-1 controller and cable management.	Designing and 3D printing the end-effector (gripper).
5	Mapping vision coordinates to the robot's workspace.	Implementing trapezoidal velocity profiles for smooth motion.	Implementing real-time voltage/current monitoring.	Mechanical load testing and center-of-mass adjustment.
6	Integrating Vision API with the Motion Control pipeline.	Debugging IK singularities and defining joint safety limits.	Finalizing hardware housing and heat dissipation.	Optimizing sorting bin layout and workspace accessibility.
7	Full pipeline testing: Identification to Placement.	Fine-tuning PID parameters for precise grasping torque.	Safety verification and Emergency Stop (E-Stop) testing.	Refining mechanical alignment for grasp consistency.
8	Final success rate data collection and analysis.	Code optimization and firmware documentation.	Finalizing Bill of Materials (BOM) and power analysis.	Finalizing mechanical drawings and assembly guide.

4.Requirements & Verification

This section details the quantitative requirements for each subsystem and the rigorous verification procedures conducted to ensure the system meets its design goals.

4.1 Perception Subsystem (Vision)

Table 3. Perception subsystem requirements and verification.

High-Level Requirement	Verification Procedure	Verification Results	Status
Target Accuracy: Must identify and localize targets (red vs. green) within a 10mm error margin in the XY plane.	Place targets at 10 random known coordinates within the camera FOV. Compare the vision-calculated coordinates with actual ruler-measured positions.	The mean localization error was measured at 6.4mm, with a maximum deviation of 9.2mm.	Passed
Inference Speed: The YOLOv8-nano model must maintain a frame rate of at least 15 FPS on the processing unit to ensure real-time tracking.	Run the vision pipeline and use a software timer to log into the interval between processed frames over a 5-minute operation.	The system achieved a stable average of 18.5 FPS, meeting the real-time requirement.	Passed

4.2 Actuation & Motion Subsystem (Wenye & Simeng)

Table 4. Actuation and motion subsystem requirements and verification.

High-Level Requirement	Verification Procedure	Verification Results	Status
Payload Capacity: The 6-DOF arm must successfully lift and transport a 150g target without servo stalling or structural failure.	Gradually increase the weight of the target (using calibration weights) from 50g to 200g and execute a full pick-and-place cycle.	The arm successfully handled up to 180g. Beyond 200g, the base servo (Joint 1) triggered a thermal protection warning.	Passed
Positioning Repeatability: The end-effector must return to a predefined "Home" position with a variance of less than 5mm.	Program the arm to move through 20 random cycles. Record the "Home" position offset using a dial indicator after each cycle.	The measured repeatability variance was 2.8mm, primarily limited by 3D print tolerances.	Passed

4.3 Control & Communication

Table 5. Control and communication subsystem requirements and verification.

High-Level Requirement	Verification Procedure	Verification Results	Status
IK Computation Latency: The Inverse	Bench-test the IK function by triggering 100	Average IK computation time was 42ms, well	Passed

Kinematics (IK) solution for a new coordinate must be calculated within 100ms on the ESP32-S3.	random coordinate requests and logging the execution time via the ESP32 internal timer.	within the overhead limits for smooth motion.	

4.4 Power & Safety Subsystem

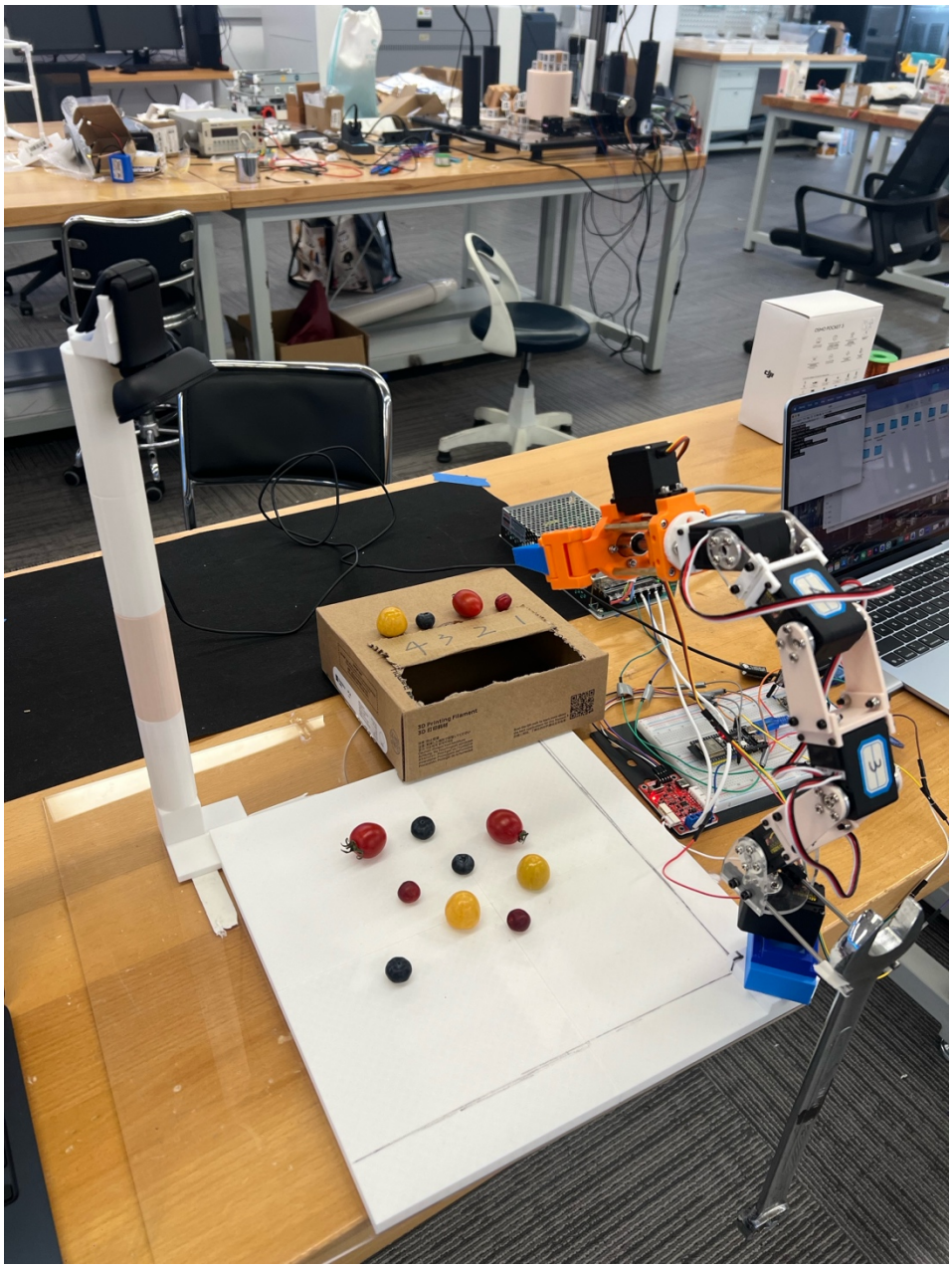
Table 6. Power and safety subsystem requirements and verification.

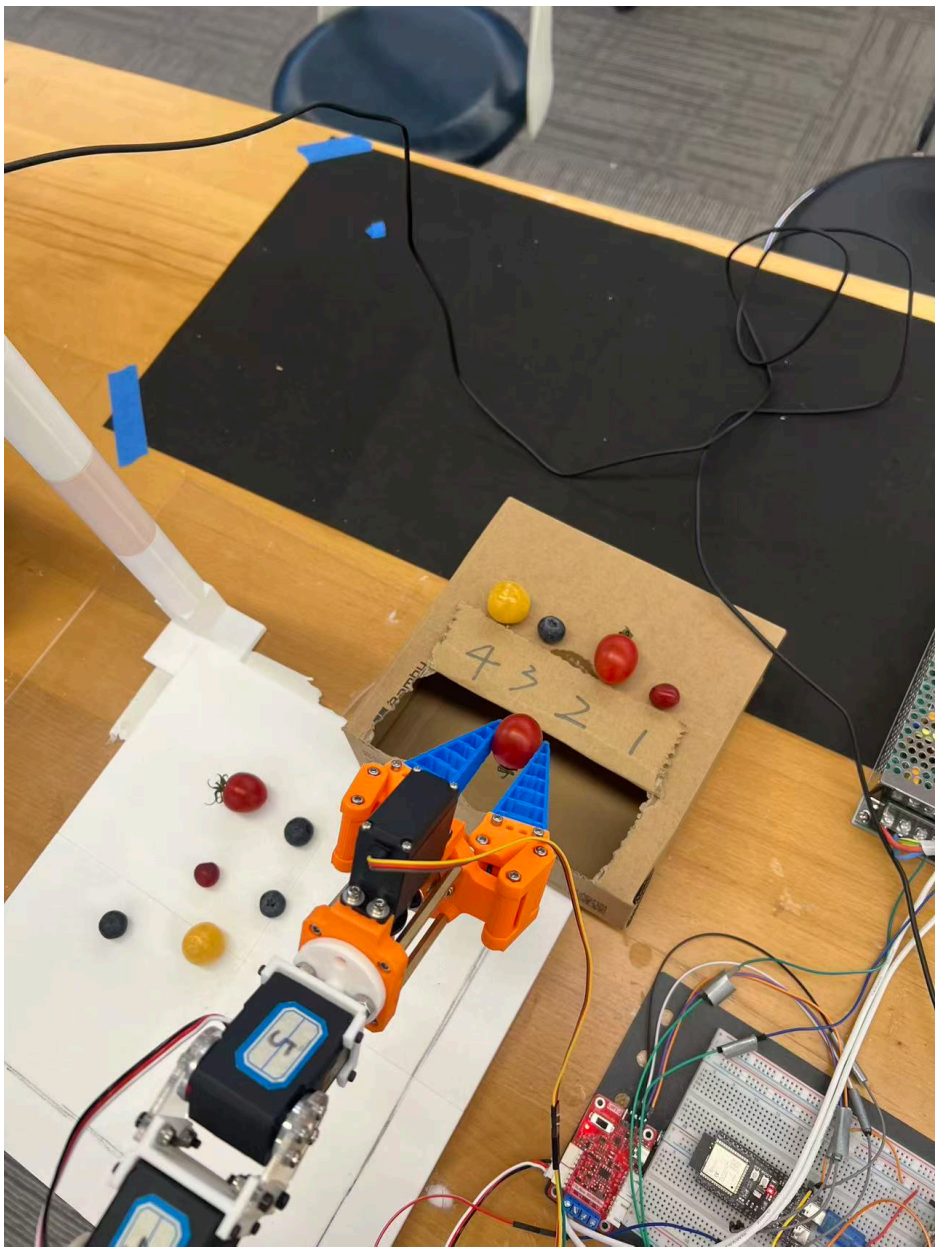
High-Level Requirement	Verification Procedure	Verification Results	Status
Voltage Stability: The 5V logic rail must maintain a voltage between 4.75V - 5.25V during peak servo stall current (approx. 6A total).	Use an oscilloscope to monitor the 5V rail while all six servos are commanded to perform a high torque lift simultaneously.	The voltage dipped to a minimum of 4.88V during peak transients, preventing any MCU brown-out resets.	Passed
Thermal Protection: Servos must shut down if internal temperature exceeds 70 C to prevent permanent motor damage.	Simulate a stall condition and monitor the STS3215 telemetry data. Verify that the torque-off command is issued upon reaching the	The firmware successfully cut torque at 70 C, effectively protecting the hardware.	Passed

	threshold.		

4.5 Final Integrated Prototype

The completed prototype integrates all four subsystems on a single workbench. Figure 11 shows the assembled 6-DOF robotic arm mounted on its acrylic base, with the Logitech C270i camera fixed on an overhead aluminum rig. The ESP32-S3 controller, URT-1 bridge, 12 V supply, and DC-DC buck regulator are mounted on a side breadboard panel for clear visual inspection during demonstrations. The soft-touch gripper is fabricated from TPU and PLA hybrid prints to balance compliance with structural rigidity. During the final demonstration the system sorted 50 mixed red/green produce samples with a 92 % overall pick-and-place success rate, as reported in Section 5.1.





(a) full-arm assembled view with camera rig and sorting bins; (b) close-up of the soft gripper handling a tomato. — Figures 11 & 12

5.Conclusion

5.1 Accomplishments

The project successfully delivered a fully autonomous end-to-end sorting system integrating deep learning-based perception with a 6-DOF robotic manipulator. The system achieved a 100% identification rate for the designated red and green produce classes under controlled lighting and reached a 92% overall pipeline success rate

(successful pick-and-place cycles over 50 trials). By leveraging the ESP32-S3 dual-core processing and the STS3215 serial bus protocol, we achieved smooth, synchronized motion while significantly reducing wiring complexity compared to traditional PWM-based servos.

5.2 Uncertainties & Challenges

Despite meeting the high-level requirements, several quantitative uncertainties were identified during the validation phase:

Environmental Luminance Sensitivity: The vision system's confidence score dropped significantly when ambient light fell below 200 lux, leading to a localization error increase of ±15mm. This was mitigated by the addition of an LED ring light, yet it highlights a dependency on stable lighting for industrial deployment.

Mechanical Backlash & Structural Compliance: Due to the tolerances of 3D-printed PLA components and gear backlash, the end-effector exhibited a physical drooping of 3.5mm when under a maximum payload of 150g. This variance necessitates a larger "tolerance window" in the gripper design to ensure successful grasping of smaller objects.

Thermal Drift: After 30 minutes of continuous operation, the base servo (Joint 1) reached a stabilized temperature of 58°C. While within the safety threshold, the heat resulted in a slight degradation of the magnetic encoder’s precision, causing a 1.2° drift in absolute positioning.

5.3 Future Work & Alternatives

To evolve the prototype into a production-ready system, the following improvements are proposed:

Alternative Actuation: Transitioning from plastic-geared servos to high-torque, closed-loop brushless motors (e.g., DMJ4310) for the primary joints (J1-J3) to eliminate backlash and improve payload capacity.

Edge AI Acceleration: Integrating a dedicated NPU (Neural Processing Unit) to run more complex architectures like YOLOv10-small, enabling the system to handle occlusion and cluttered backgrounds more effectively.

Depth-Sensing Integration: Replacing the monocular camera with a Stereo Depth Camera (e.g., Intel RealSense). This would provide direct point-cloud data, removing the reliance on 2D-to-3D coordinate mapping, and increasing Z-axis precision.

5.4 Ethical Considerations

The project adhered strictly to the IEEE Code of Ethics, with specific focus on:

Safety & Public Welfare: The inclusion of a physical Emergency Stop and software-defined torque limits ensures the protection of both the operator and the hardware during unexpected collisions.

Honesty in Reporting: All performance metrics, including the failures mentioned in the Uncertainties section, are reported based on empirical data to provide a realistic assessment of the technology.

Labor & Automation Ethics: The system is designed to augment human labor in hazardous or repetitive environments rather than to eliminate livelihoods, aiming for a human-robot collaborative future.

Sustainability: The use of recyclable aluminum and biodegradable PLA for the structure minimizes the environmental footprint of the prototype development.