Tracking Recyclables Automated Sorting Hand T.R.A.S.H.

Group 7:

- Jayden Lumas
- Tristan Koopman
- Sayed Almousawi
- Dylan Bowers

Engineers

Jayden Lumas

Computer Engineer

Dylan Bowers

Photonics Engineer

Tristan Koopman

Mechanical Engineer

Sayed Almousawi

Electrical Engineer

Background & Motivation

The T.R.A.S.H. (Tracking Recyclables Automated Sorting Hand) project was developed in response to the growing need for efficient, scalable solutions in the waste management industry. As the U.S. population faces long-term labor shortages driven by declining fertility rates, industries reliant on manual labor, such as recycling, must adapt. While automation in domestic environments still faces safety and complexity challenges, industrial settings offer ideal conditions for deploying robotics due to their structured workflows and predictable environments. The TR.A.S.H. system is designed to bridge this gap by introducing an intelligent, robotic sorting solution that can reduce labor dependency and improve sustainability practices in line with national green initiatives.

- **Labor shortages** caused by declining fertility rates highlight the need for automation.
- **Industrial environments** are ideal for robotics due to their predictability and low safety complexity.
- **Manual recycling tasks** are repetitive and low-skill, making them strong candidates for automation.
- **Automation supports sustainability**, helping meet growing demand for efficient green waste solutions.
- **Robotic sorting systems** free human labor for more complex, high-value tasks.

Goals & Objectives

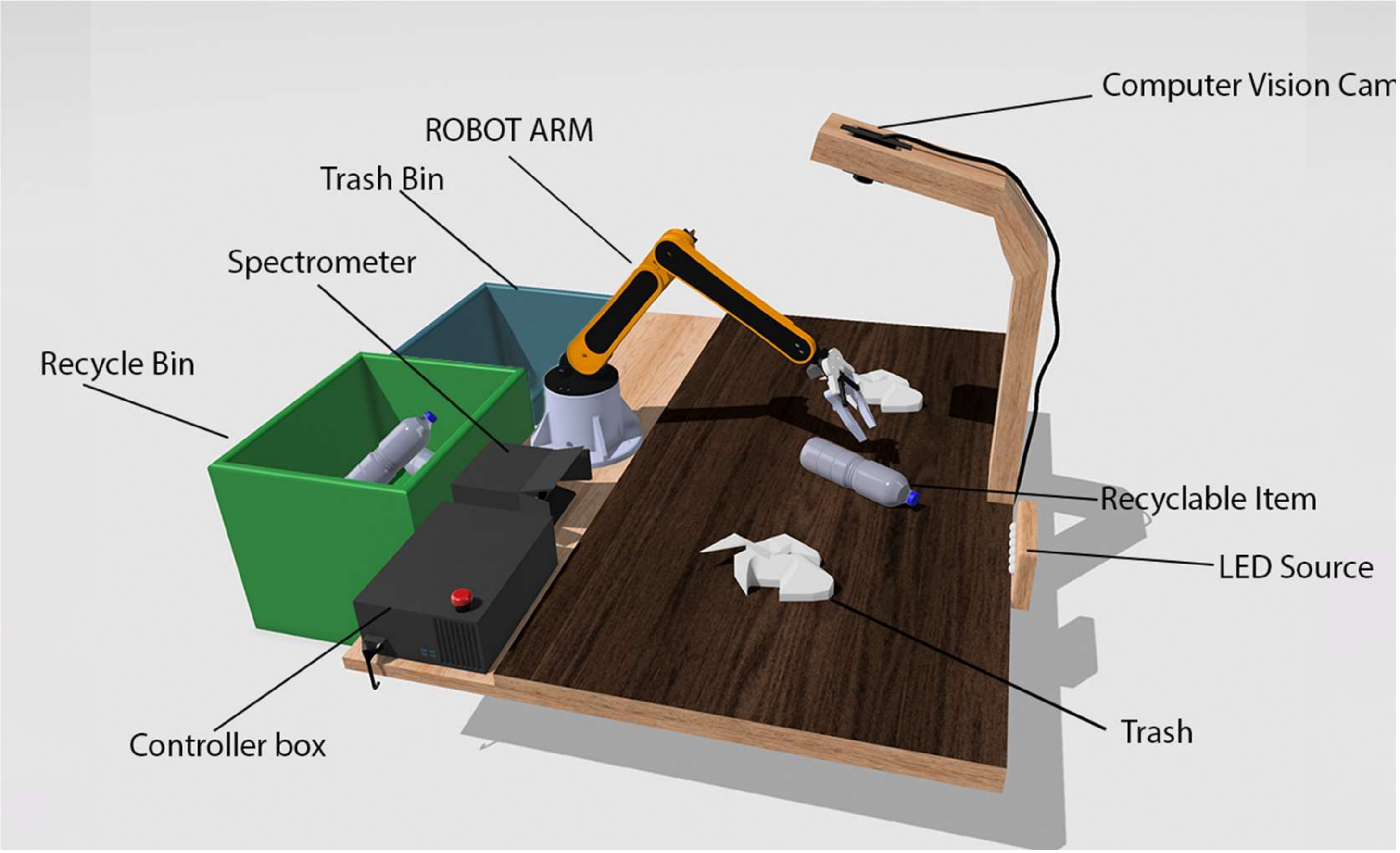
Pick & Sort	Develop an Automated Robotic Sorting System • Build a robotic arm that can pick up and sort recyclable materials like plastic, aluminum, and glass.
Identification	Enable Material Detection and Classification • Use cameras and sensors to identify the type of material without human input.
Automation	Improve Recycling Efficiency • Automate the sorting process to reduce human labor and increase speed and accuracy.
Affordability	Keep the System Lightweight and Affordable • Design with low-cost components and compact parts to make it accessible for small recycling centers.
Scalability	Make the System Easy to Use and Expandable • Create a simple interface and design the system to be easily integrated into existing setups.

Engineering Specifications

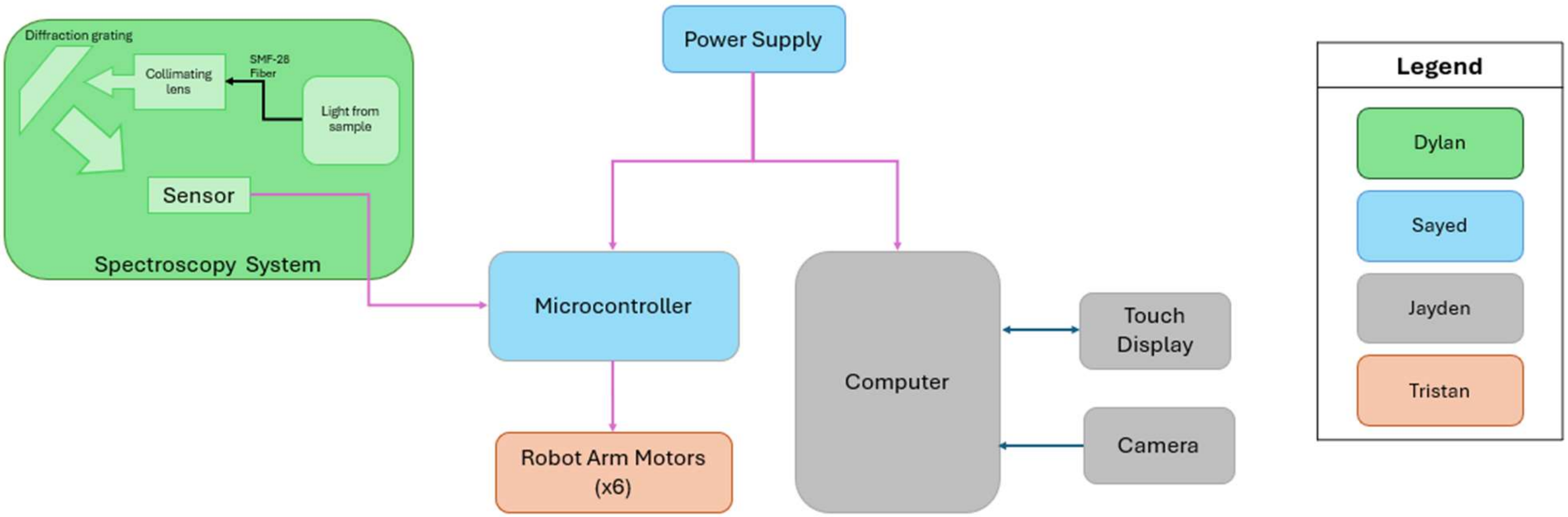
	Specification	Target Value	Units
General Specifications	Object Detection Time	20	sec
	Total Pickup Time	30	sec
	Pickup Accuracy	± 5	mm
	Min. Payload	100	g
	Arm Degrees of Freedom	6	DOF
	Power Consumption	≤ 350	W
	Power Supply	120	VAC
	Computer Vision	1	Camera
	Material Identification	At least 2	Materials

	Specification	Target Value	Units
Optical Specifications	Slit Width	$8 \leq x \leq 40$	μm
	Spectral Resolution	< 20	nm
	Lens Focal Distance	100	mm

Illustration



Simplified Hardware Block Diagram



Microcontrollers

- Required Capabilities For Optical Sensor
  - Fast and accurate ADC
  - Multiple precise timers with advanced controls and triggering functionalities.
  - 32-bit timers
  - DMA support
  - Fast communication protocols

- Required Capabilities For Motor Control
  - Multiple UART interfaces
  - I2C interface
  - Easy programming and debugging

Microcontroller Selection

- Since the CCD sensor is sensitive and has complex requirements, it is on a separate PCB with its own MCU.
- Timer capabilities and ADC accuracy led us to choosing the STM32 for driving the CCD Sensor.
- Availability of CCD sensor project examples using the STM32 online was a key factor.
- The 3xUART, I2C, and ease of programming capabilities of the ESP32-S3 are very useful for our motor control system which is on the main PCB.
- Price was not a main issue since the capabilities were most important for our project to function.

MCU	USB	UART	I2C	Timers	ADC	DMA	Cost
ATmega328P	NA	1	1	2xTimer 8-bit 1xTimer 16-bit	8 channels 10-bit	NA	$1.46
ESP32-S3 WROOM-1	OTG FS	3	2	4xTimer 54-bit (16-bit prescaler)	2xSAR ADC 10 channels 12-bit	YES	$5.71
STM32F401	OTG FS	3	3	2xTimer 32-bit 6xTimer 16-bit	10 channels 12-bit	YES	$5.30

Motor Technology

- Required Capabilities:
  - High torque to handle the payload specifications
  - Closed Loop System for precise control
  - Range of Motion
  - Low Cost

- For the main joints, high torque, precise control, and full range of motion is required.
- Stepper motors meet those requirements.
- End Effector doesn't require such capabilities, so servo motors were chosen to reduce cost.
- BLDC Motors are the perfect choice but very expensive!!

Motor	Position Control	Range of Motion	Holding Torque	Driver	Cost
Servo	Closed Loop (internally)	180°-270°	~0.5 - 3 Nm	PWM Signal	~\$2 - \$10
BLDC	Open Loop	360°	~0.1 - 100 Nm	ESC (Electronic Speed Controller)	~\$10 - \$50
Stepper	Open Loop	360°	~0.1 - 8 Nm	Stepper Driver (e.g., A4988)	~\$80 - \$200

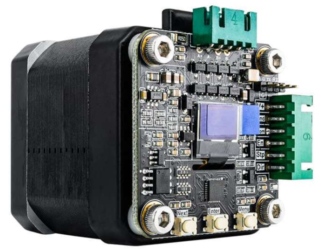
Motor Selection

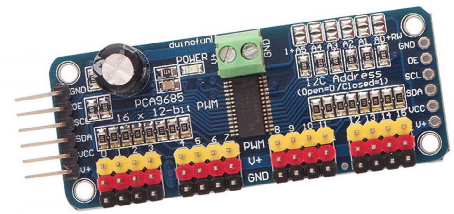
- MG996R were chosen since they provide better durability and highest torque.
- Nema 17s provide enough torque while drawing less current and have a smaller overall dimensions tan Nema 23s.

	Motor	Durability	Presic ion	Holding Torque	Cost
Servo	**SG90**	~100,000 cycles	±10°	1.8 kg·cm @ 4.8V	$2.38
	MG90S	~150,000 cycles	±8°	~2.2 kg·cm @ 4.8V	$3.5
	MG996R	~200,000 cycles (metal gear)	±5°	~9–11 kg·cm @ 6V	$4.25

	Motor	Step Count	Current	Holding Torque	Cost
Stepper	**Nema 14**	200 steps/rev (1.8°/step)	~0.4 - 1.0 A/phase	0.08 - 0.12 N·m	$9.07
	Nema 17	200 steps/rev (1.8°/step)	~1.2 - 2.0 A/phase	0.12 - 0.8 N·m	$13.99
	Nema 23	200 steps/rev (1.8°/step)	~2.0 - 3.5 A/phase	1.2 - 4.0 N·m	$25.99

Driver Selection





- Evaluated three stepper drivers: A4988, TMC2209, and MKS Servo42C
- A4988: basic, cheap open-loop driver
- TMC2209: adds microstepping and quiet operation, still open-loop
- MKS Servo42C: closed-loop with built-in encoder, mounted directly on motor
- Chosen: MKS Servo42C
  - Simplifies wiring and mounting
  - Enables feedback and tuning using PID
  - Supports UART control for precise, reliable performance
- For Servo Motors, we chose to go with PCA9685 I2C PWM Driver instead of GPIOs to reduce CPU load.

Driver	Control	Interface	Mounting	Size	Features	Cost
A4988	Open-Loop	Step/Dir	External	0.6” x 0.8”	Basic stepper control	$1.74
TMC2209	Open-Loop	Step/Dir	External	0.6” x 0.8”	Quiet stepping, stealthChop	$2.79
MKS Servo42C	Closed-Loop	UART, FOC, Step/Dir	On Motor	1.65” x 1.65”	PID control, auto tuning, feedback	$15.22

Single Board Computer

What we need:
- Capable of performing Object Detection within 1 seconds of a new object being on screen
- GPIO pins to communicate with our MCU
- Able to handle Machine Learning and ROS2 Nodes

Orange Pi

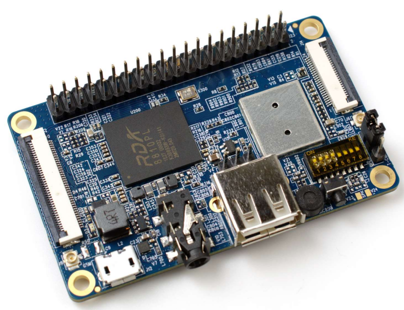
- Powerful octa-core CPU and integrated 6 TOPS Neural Processing Unit (NPU) for edge AI
- Offers high multi-threaded performance and dual-camera support at a lower cost
- Less software maturity and weaker ecosystem support compared to Raspberry Pi or Jetson

Nvidia Jetson Nano

- Built-in CUDA GPU acceleration for efficient AI inference and deep learning models
- Optimized software stack with TensorRT and NVIDIA Vision SDK for machine vision workflows
- Best for AI-focused tasks, but limited RAM (4GB) and storage flexibility may restrict scaling

Raspberry Pi 5 + Coral Accelerator

- Strong overall performance with fast CPU, high RAM (8GB), and excellent ROS2 compatibility
- Dual-camera CSI support and robust I/O make it ideal for real-time vision and control tasks
- Affordable, well-documented, and backed by a large support community for reliable development



Camera

Raspberry Pi HQ

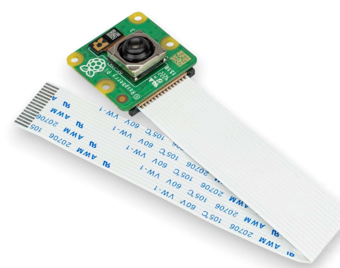
- Offers custom lenses for adjustable focus and zoom
- No HDR or autofocus; works best in controlled lighting
- Great for high-quality images where precision optics are needed

ArduCam 16MP

- High resolution with autofocus for detailed images
- No HDR, and not fully plug-and-play with Raspberry Pi
- Best for well-lit environments where resolution matters most

Raspberry Pi v3/ArduCam 12MP

- Has HDR and autofocus for better image clarity and lighting flexibility
- Fully compatible with Raspberry Pi OS and libraries like OpenCV and libcamera
- Affordable ($25) and reliable, making it ideal for real-time object detection



Linear CCD Sensor Selection

- The Spectral response is identical for both products
- Leaving Sensitivity as the next most important factor
- The TCD1304DG's much larger pixel area (i.e., 8 × 200 µm vs 7 × 7 µm) translates to significantly higher photon collection per pixel, yielding a dramatically greater sensitivity (~160 V/(lx ·s) for TCD1304DG

Specification	TCD1501D	TCD1304DG
Number of Elements	5000 elements	3648 elements
Pixel Size	Low: 7 µm by 7 µm	High: 8 µm by 200 µm
Sensitivity	Low: 10.4 to 15.6 V/lx·s	High: 110 to 160 V/lx·s
Dark Signal Voltage	2 mV	Up to 5 mV
Dynamic Range	3000	Up to 300 (no unit specified)
Saturation Output Voltage	3 V	450 to 600 mV
Output Impedance	0.5 to 1 kΩ	0.5 to 1 kΩ
Output Signal Voltage	Not specified	1.5 to 3.5 V
Operating Temperature	-20° C to +60° C	-25° C to +60° C
Package Type	22 pin CERDIP	22 pin CERDIP

Comparison of Illimitation Sources

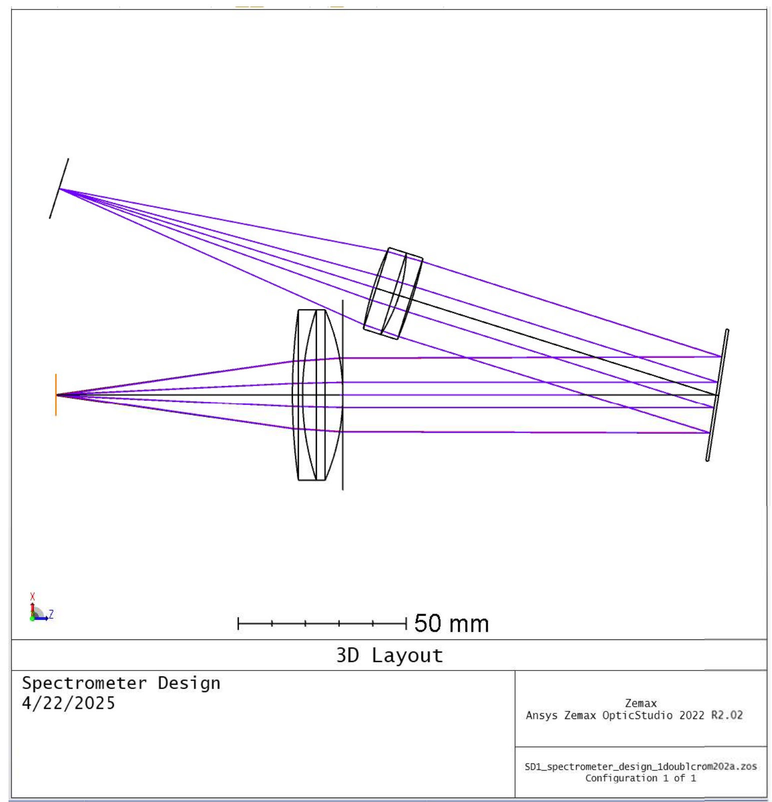
- Incandescent Source
  - Pros
    - Continues and inform illumination for 400 nm to 100 nm
  - Cons
    - High Power Consumption
    - High Operating temperatures that may damage 3D printed parts
- LED sources
  - Pros
    - Low power consumption
    - More flexible with system integration
  - Cons
    - Non-uniform illumination across the spectrum

Name	Central lambda	Lambda Range (FWHM)	Package	Quantity
LEDW7E	N/A	430 to 660	T-1 3/4	2
LED680L	660	670 to 690	TO-18	2
LED760L	760	748 to 772	TO-18	2
LED800L	800	785 to 815	TO-18	2
LED890L	890	868 to 912	TO-18	2
LED930L	930	900 to 960	TO-18	2
LED970L	970	947 to 993	TO-18	2
LED1050E	1050	995 to 1105	T-1 3/4	2

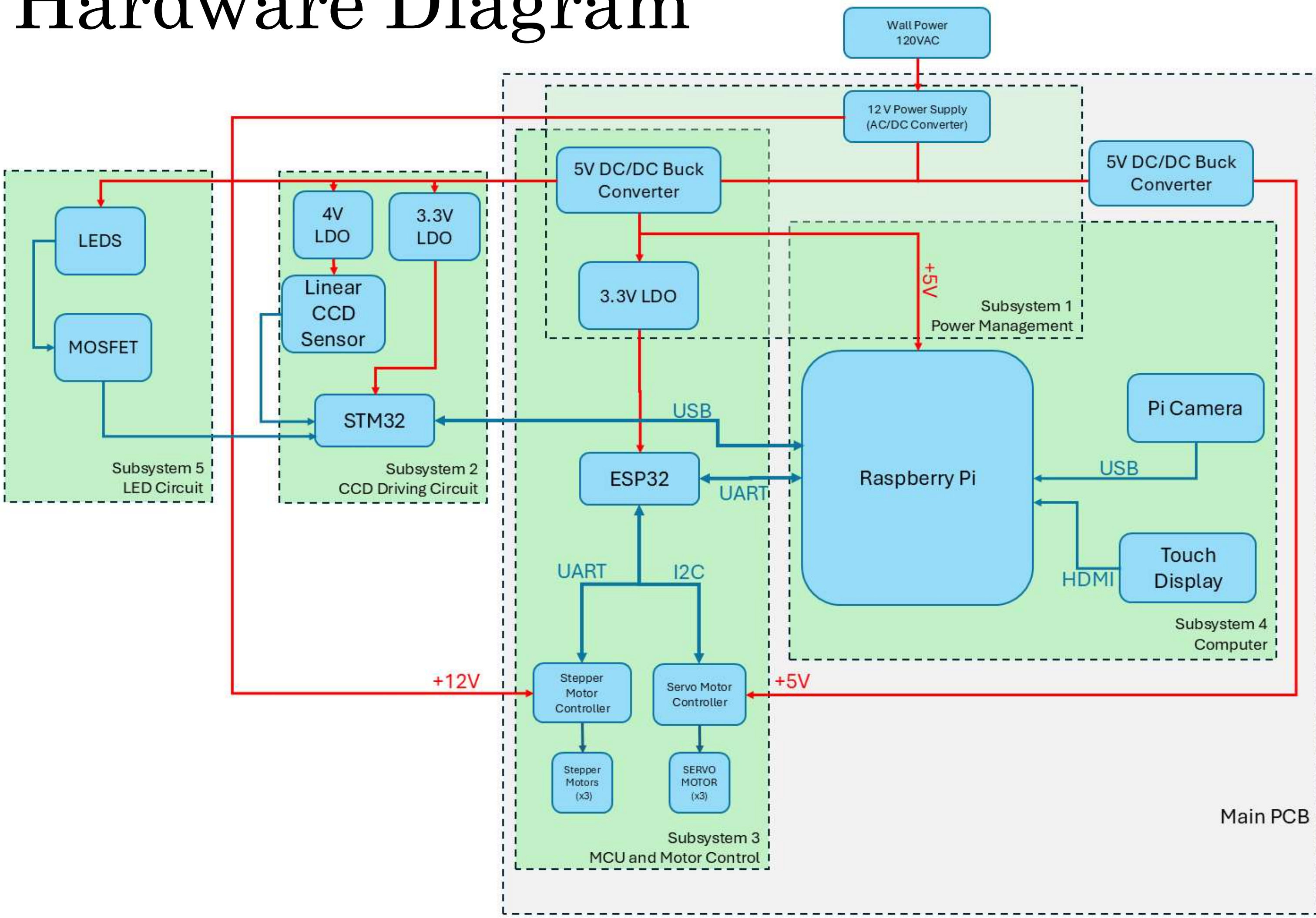
List of LEDs required to illuminate from 400 nm to 1100 nm

Hardware Design for Spectrometer

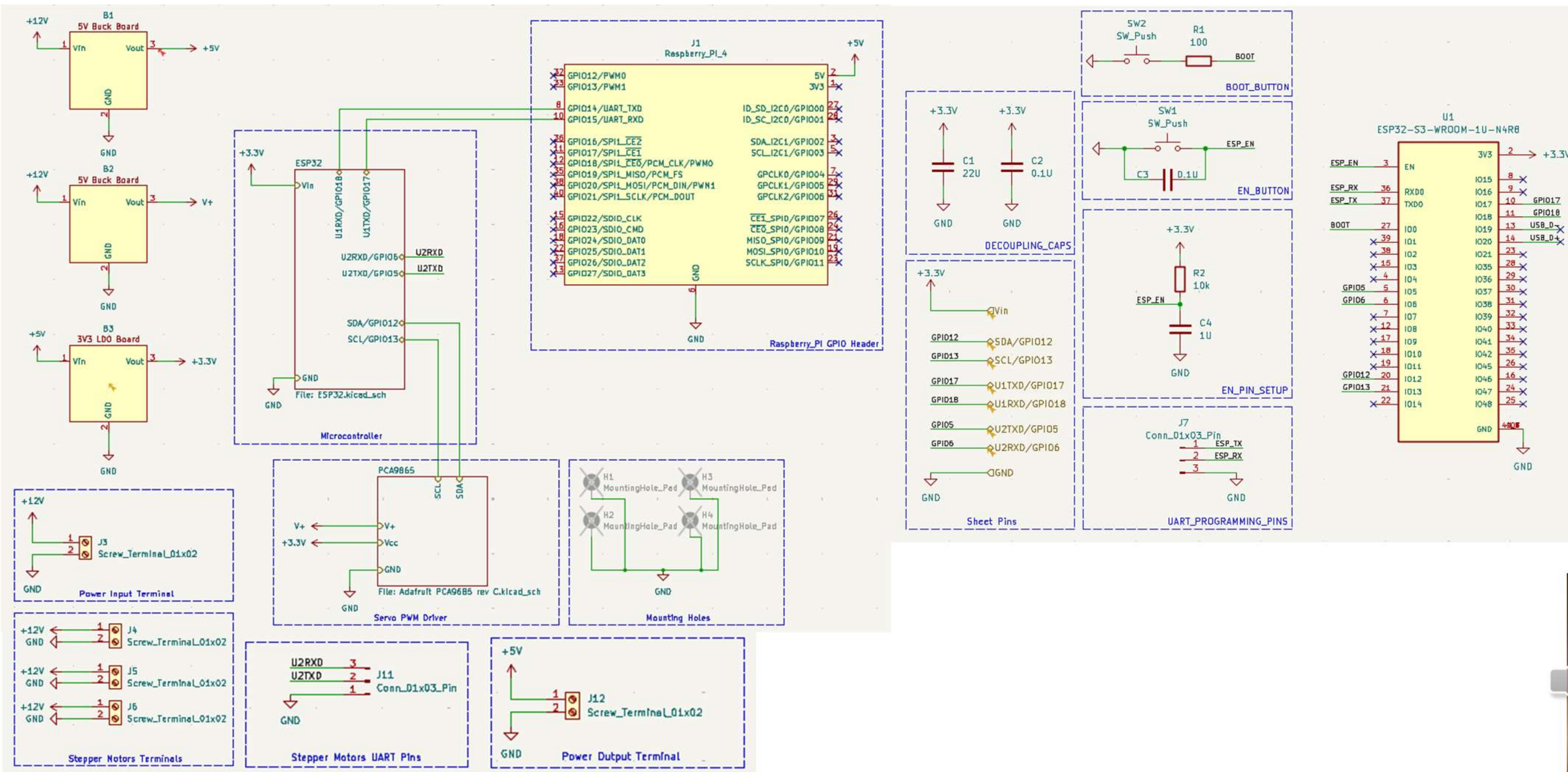
- The goal of this spectrometer design is to image the slit onto a single detector pixel by matching the slit width to the pixel pitch, optimizing spatial-spectral resolution in an aberration-limited regime (8 μm).
- The slit width is 10 μm to image into a single pixel the optical system has a 0.8 magnification.
- The final optimization that need to be done is insuring small spot size for all wavelengths



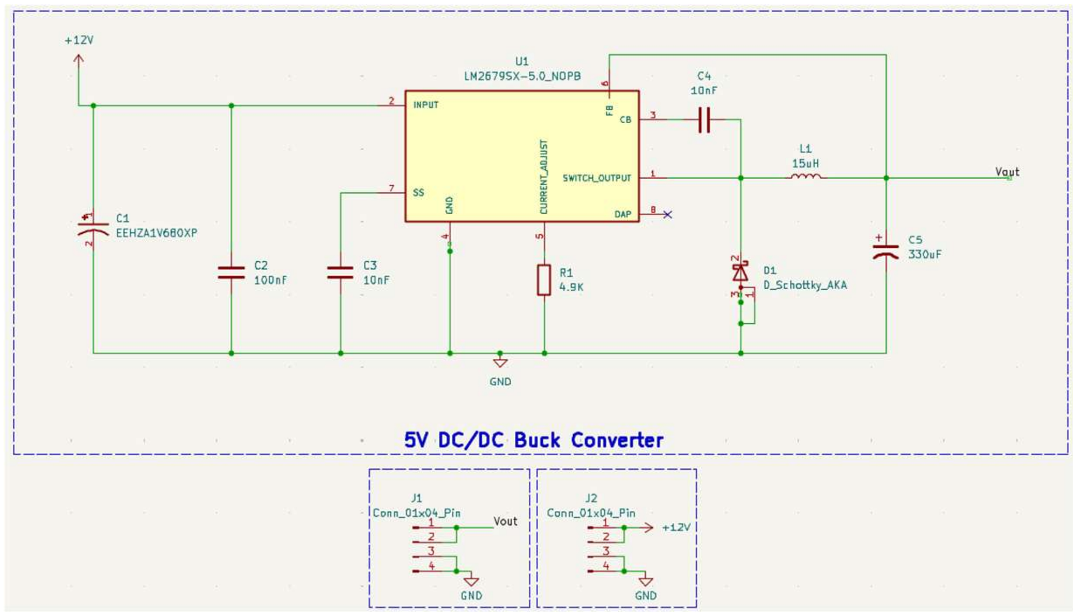
Hardware Diagram

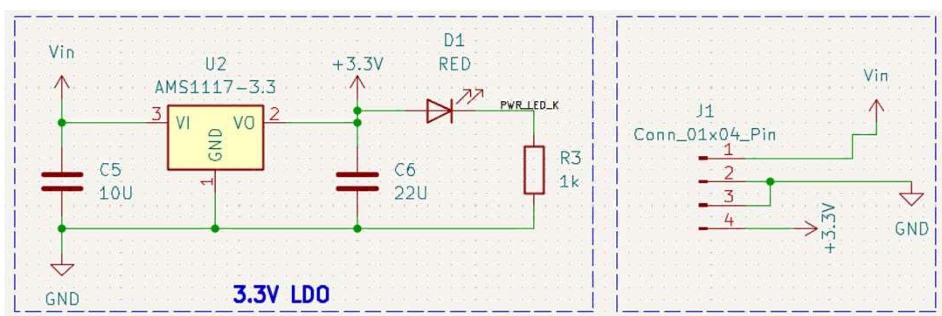


Main PCB Schematic

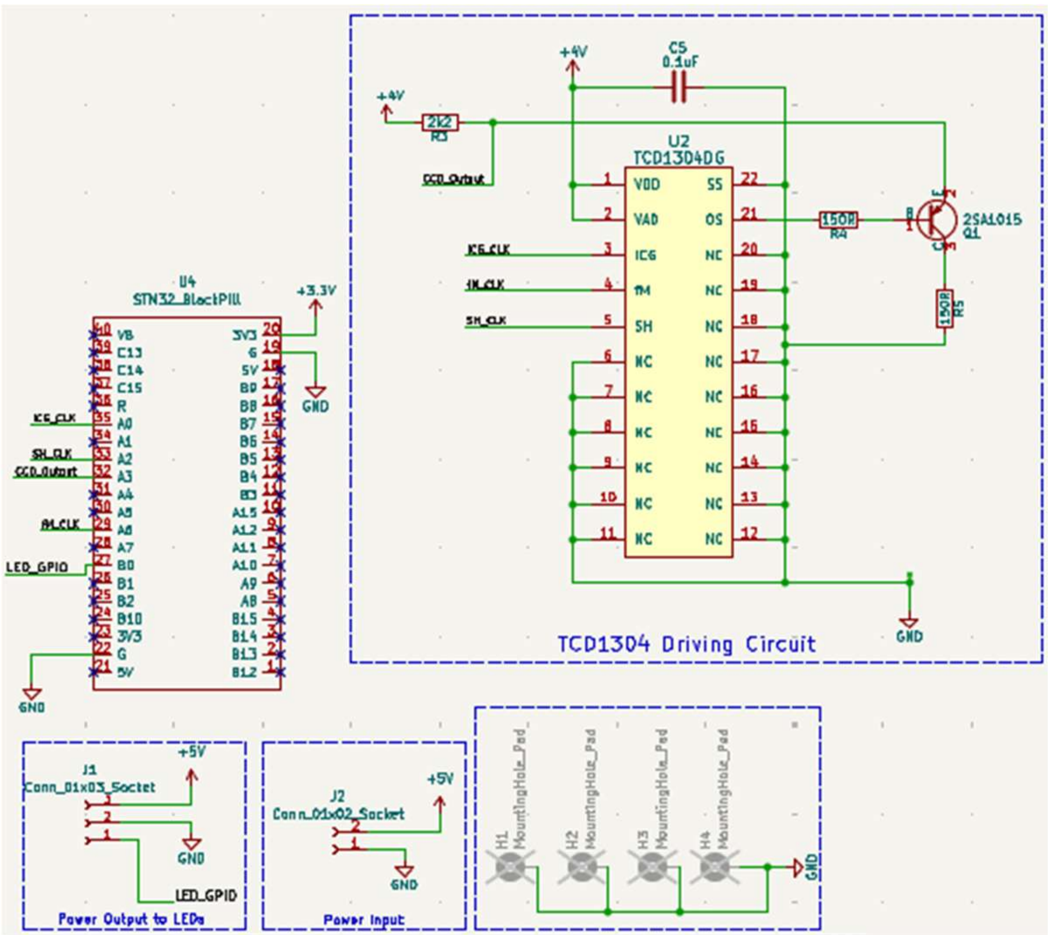


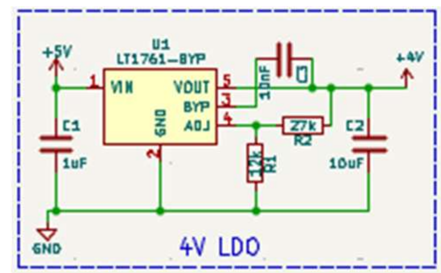
Power Regulators Schematic

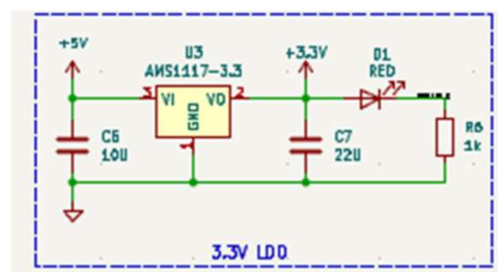




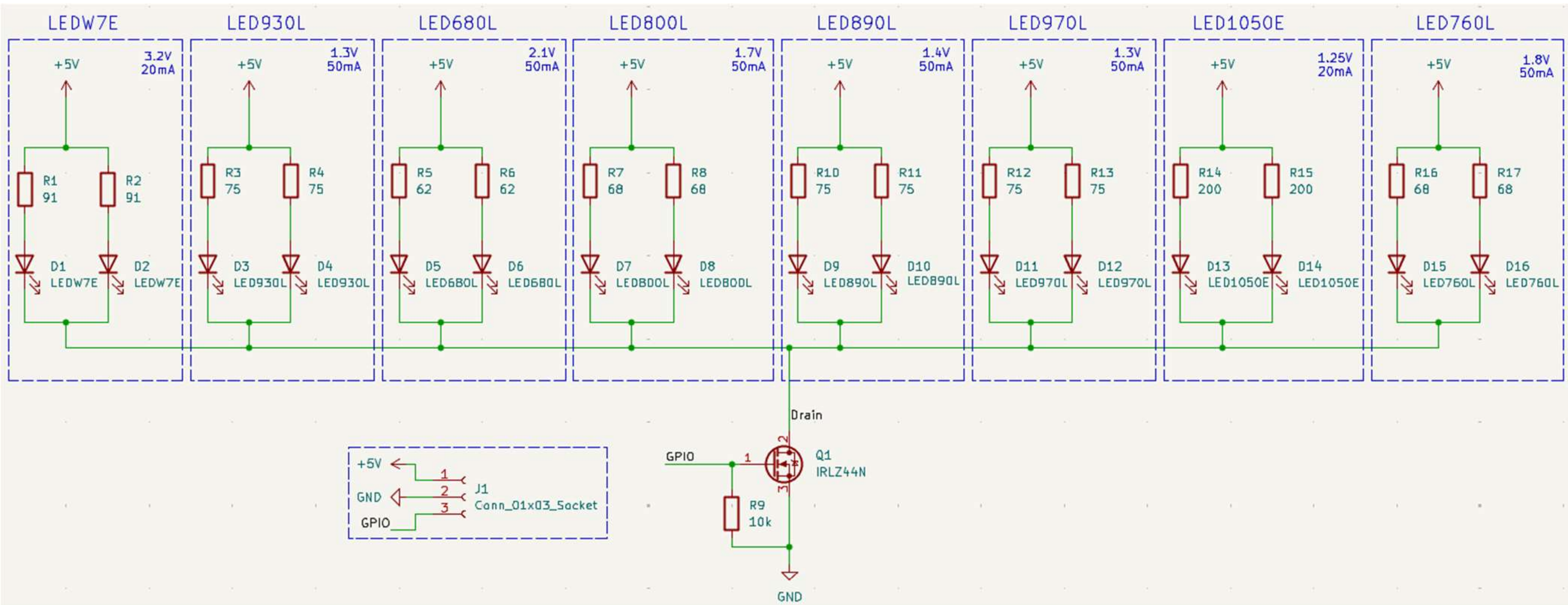
CCD Sensor Board Schematic

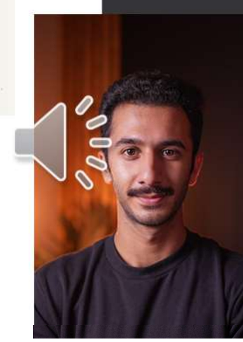






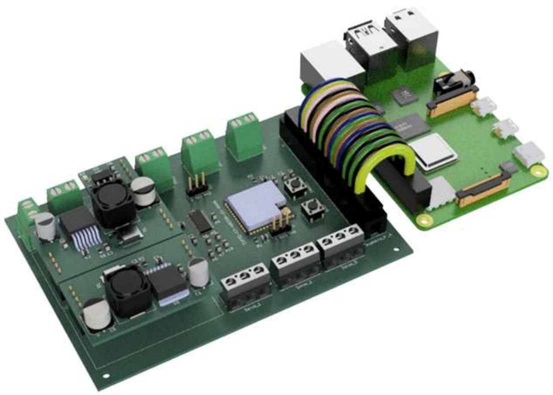
LED Ring Schematic



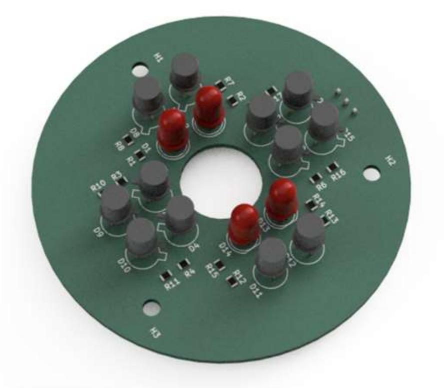


PCB Design

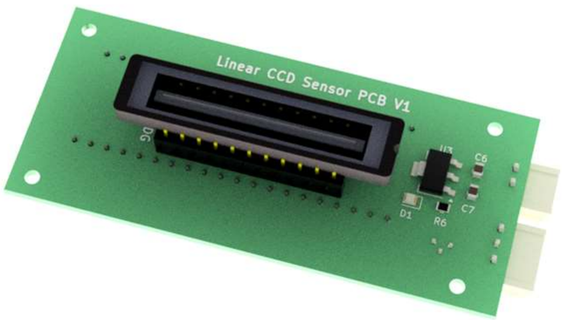
- Main Board
  - Includes all power and regulators circuitry
  - Regulators on Daughter Boards
  - ESP32 for motor control
  - Motor connections
  - Power output to other PCBS
  - Raspberry Pi Connection
  - Servo Driver
- Sensor Board
  - Has its own power regulators required for MCU and sensor
  - 5V power input connector
  - 5V and GPIO connector for LED PCB
  - STM32 DEV Board on the back side
- LED Board
  - Has 16 specific LEDs needed for spectrometer system
  - MOSFET Controlled by the STM32 GPIO pins
  - Circle cutout in the center to fit around spectrometer slit
  - 3-Pin connector for power and control from the sensor board.



Main PCB

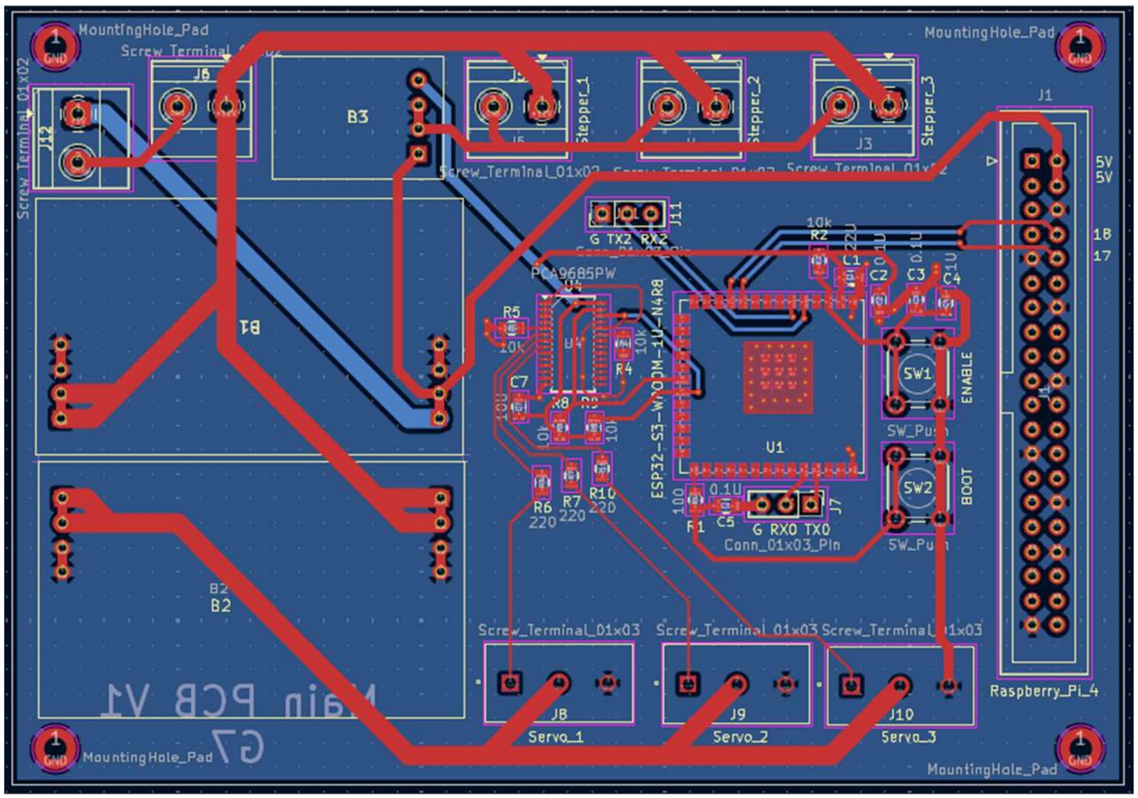


LED Ring PCB

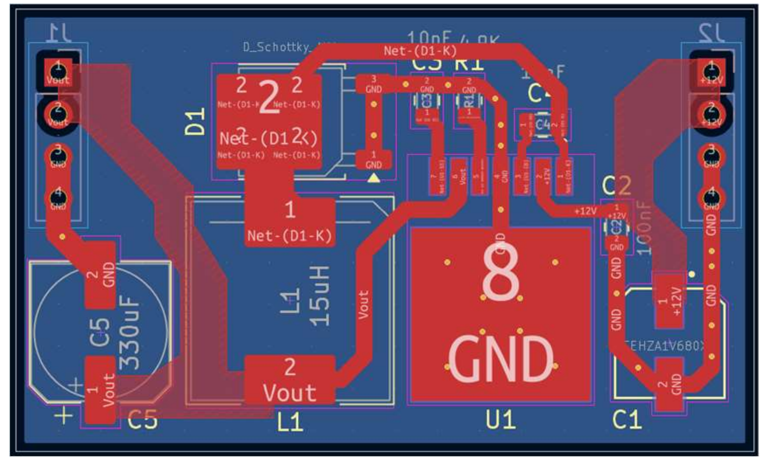


CCD Sensor PCB

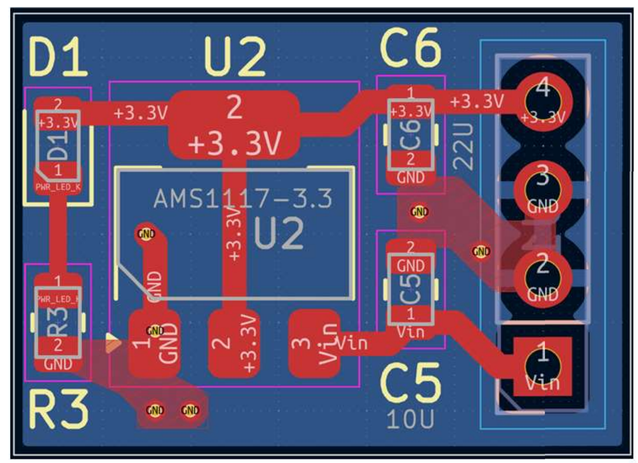
Main PCB Layout



Main PCB

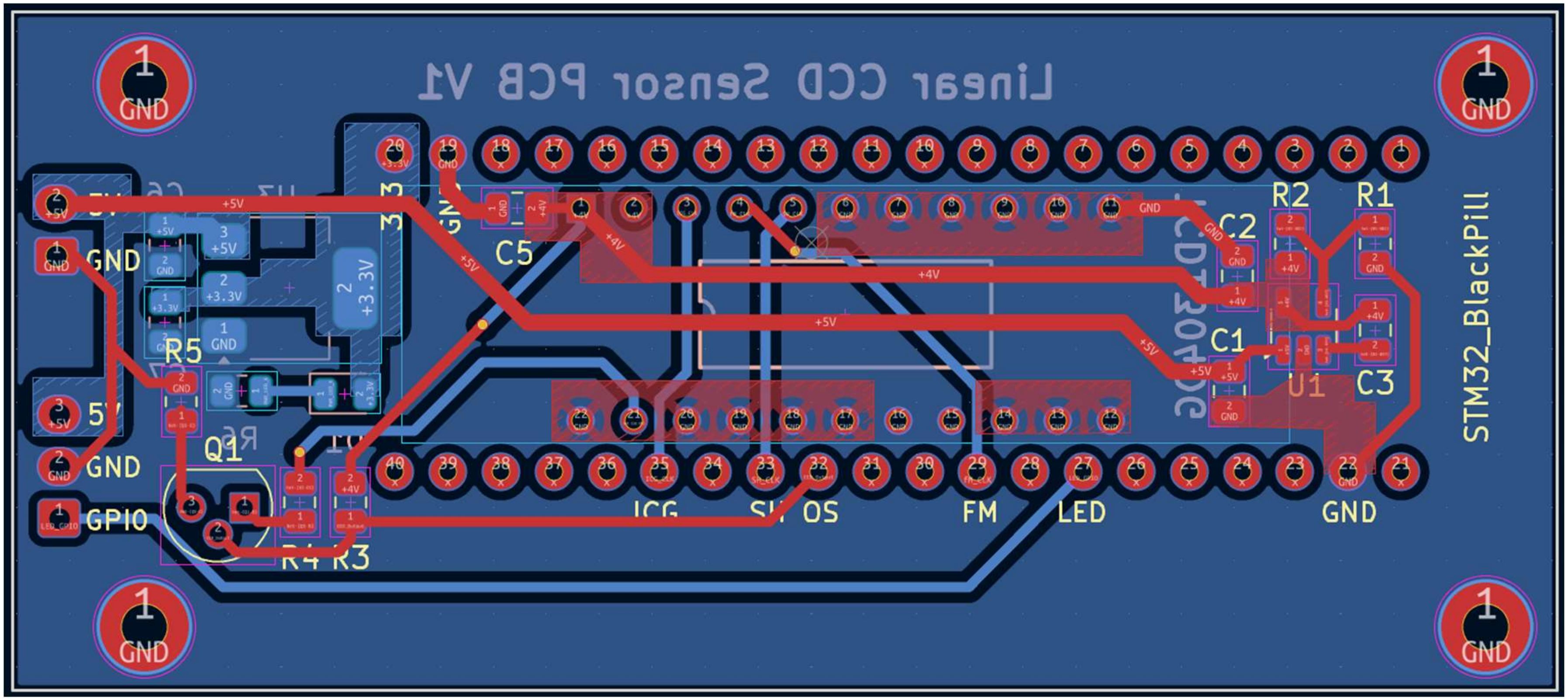


5V Buck Converter PCB

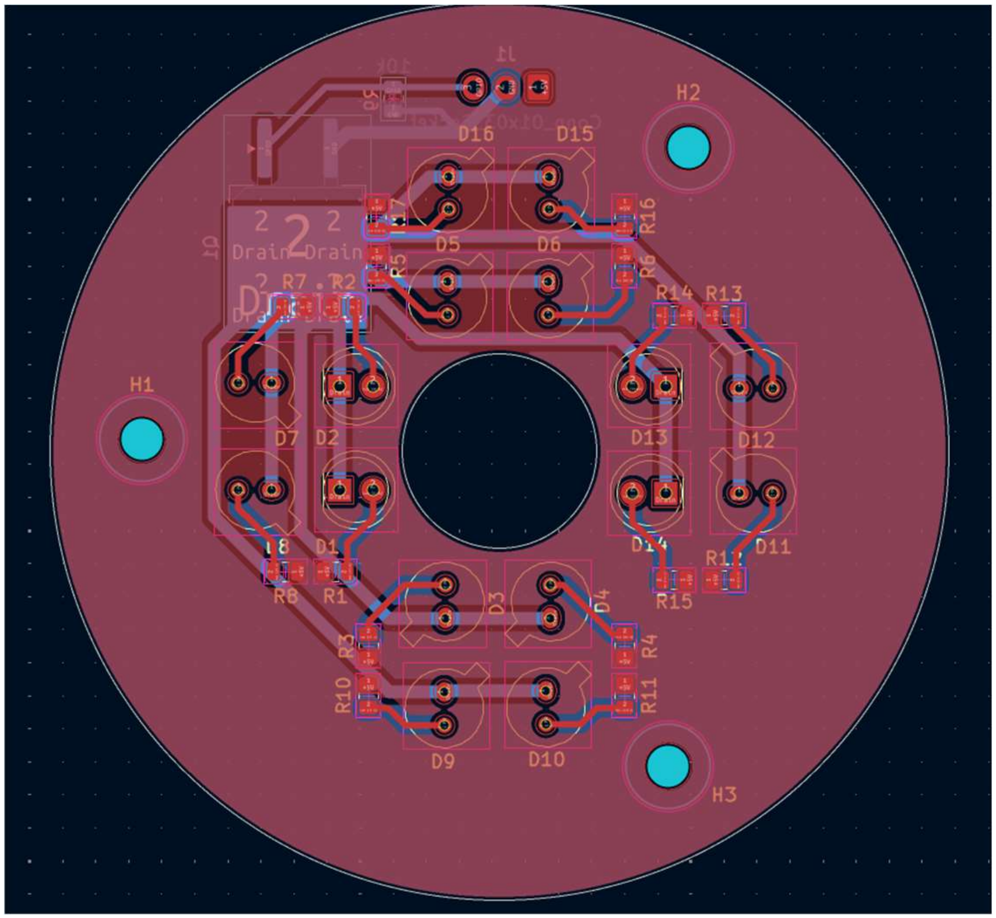


3.3V LDO PCB

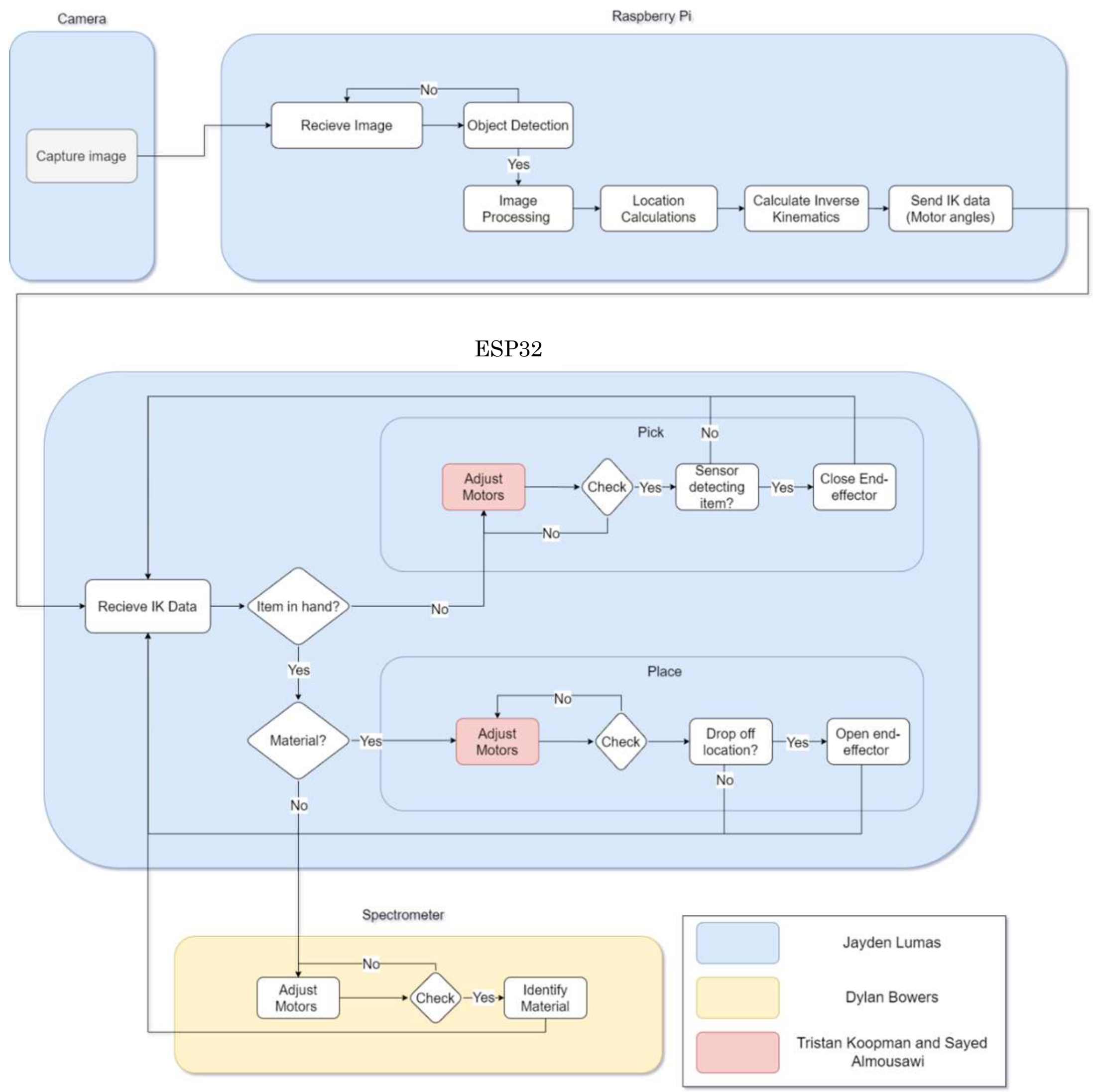
Sensor PCB Layout



LED PCB Layout



Software Flowchart

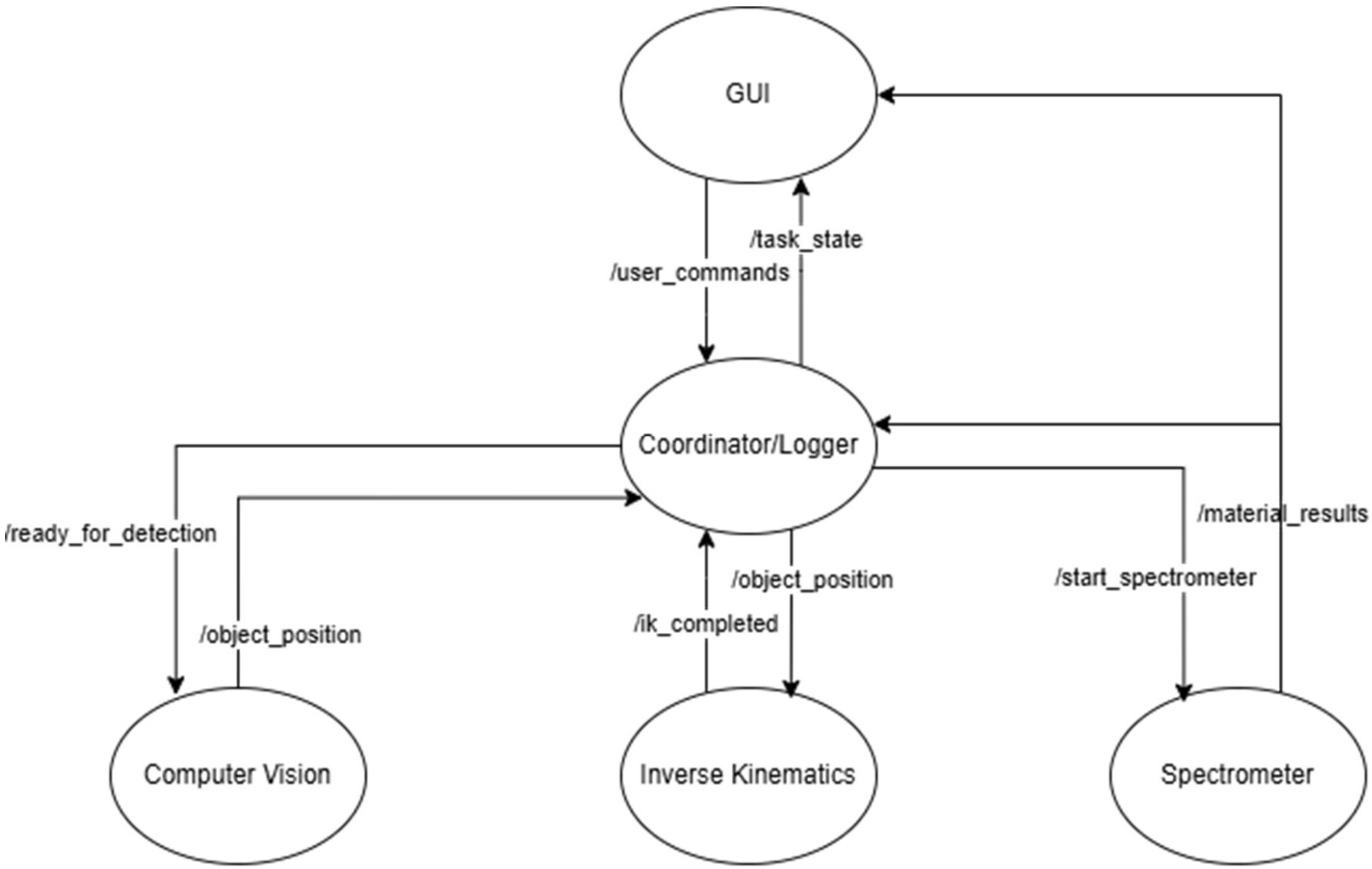


Robot Operating System 2 Overview

ROS2 offers a complete ecosystem of tools that allow us to visualize, simulate, and control our robot more efficiently than manually coding every movement. These tools work together to improve our development speed, motion planning accuracy, and system integration.

- **MoveIt2:** Handles inverse kinematics, trajectory planning, and collision avoidance automatically, removing the need to manually calculate joint angles or motion paths. It generates optimized motion plans that respect joint constraints and workspace limits.
- **RViz:** Provides a 3D interface to visualize the robot, its camera view, and real-time sensor data. It helps us debug and verify robot behavior, frame transforms, and arm motion without needing to deploy to hardware immediately.
- **Gazebo:** A 3D physics simulator that lets us test our robot in realistic environments with gravity, collisions, and inertia. Allows safe experimentation with motion and vision pipelines before deployment to physical systems.
- **ros2_control:** Standardizes how we interface with motors and actuators. Enables clean separation between hardware interfaces and commands, making our code modular and hardware-independent.
- **tf2:** Tracks spatial relationships between the camera, arm joints, and objects. Converts object positions from camera coordinates to arm coordinates for precise gripping.
- **cv_bridge + image_transport:** Connect OpenCV image processing to ROS2 image streams. Allows real-time computer vision (e.g., object detection or edge tracking) to integrate directly with robot actions.

ROS2 Design


GUI
/user_commands
/task_state
Coordinator/Logger
/ready_for_detection
/object_position
/ik_completed
/object_position
/start_spectrometer
/material_results
Computer Vision
Inverse Kinematics
Spectrometer

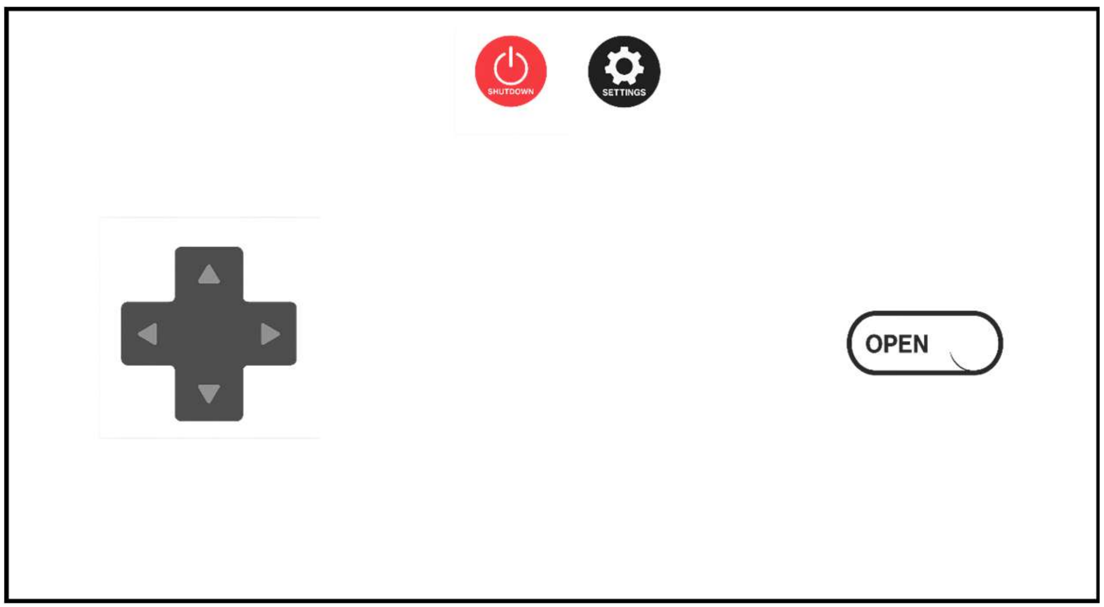
Computer Vision & Machine Learning

Raspberry Pi 5 is well-suited for computer vision and machine learning due to its powerful quad-core CPU, 8GB of RAM, and support for hardware acceleration through the Coral TPU. Its broad software ecosystem enables real-time object detection using YOLOv8, as well as edge detection for unknown or unclassified objects, ensuring flexible identification and sorting in dynamic environments.

- 8GB RAM and 64GB storage allow for smooth operation of computer vision pipelines, ROS2 nodes, and local data storage.
- Coral USB Accelerator (TPU) enables efficient and fast machine learning inference without taxing the CPU.
- Supports YOLOv8, OpenCV, and TensorFlow Lite; well-documented and community-supported for rapid implementation.
- ML models are used for real-time classification of recyclable objects; fallback edge detection is used for unknown items.
- Output from ML pipelines feeds into ROS2 for coordinated robotic control and sorting decisions.

Graphical User Interface

The TR.A.S.H. graphical user interface provides operators with real-time control and monitoring of the robotic system directly from the Raspberry Pi. Designed for both automation and manual override, the GUI allows users to position the arm, operate the gripper, review sorting analytics, and respond to system alerts through an intuitive and responsive layout. It acts as the central hub for interacting with all subsystems in a clear and user-friendly format.



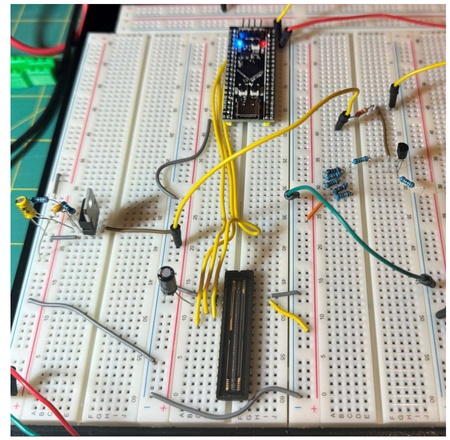
Spectrum Compare Software

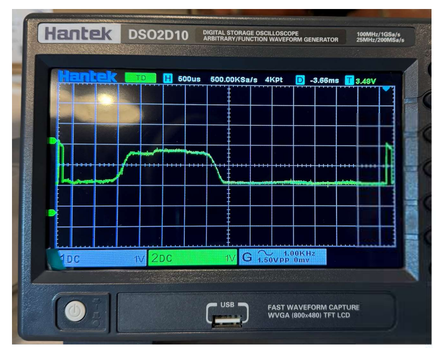
- Procedures for code to resolve the object with spectrum information

Receive unknow spectrum → Take the difference between the unknown and control spectrum to eliminate background noise → Compare the unknown spectrum to the spectrum library → Last print out the material type

Prototyping & Testing – Hardware

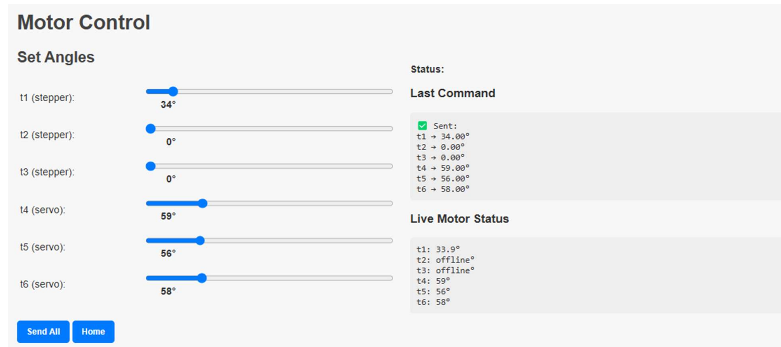
- Servo Motors
  - Tested with direct ESP32 PWM control.
  - Verified smooth movement and holding angles.
  - Upgraded to PCA9685 driver for multi-servo control testing.
  - Solved MCU resetting issues by adding a second buck converter.
- Stepper Motors
  - Used MKS Servo42C with UART commands.
  - Verified movement, direction, speed via manufacturer GUI
  - PID tests showed reliable position holding under load.
- CCD Sensor
  - Breadboarded driver circuit using STM32 timers.
  - Adjusted analog signal amplification with 2N3906.
  - Synced ADC timing; verified output with oscilloscope.
- Camera
  - Switched to Arducam due to Pi Cam V3 shortage.
  - Mounted overhead; verified clear image feed.
  - Used YOLOv8 for object detection testing

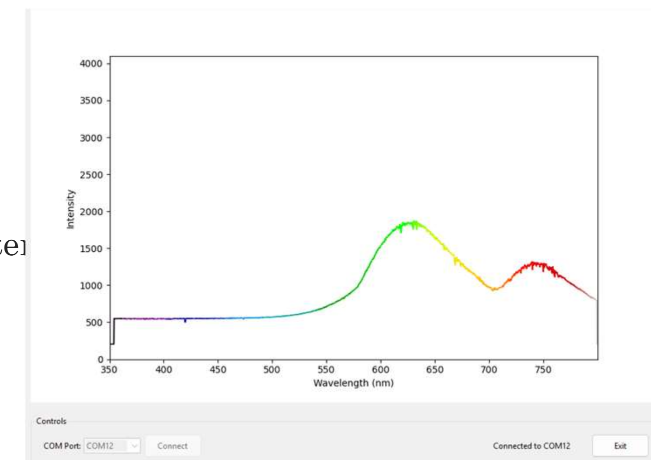




Prototyping & Testing – Software

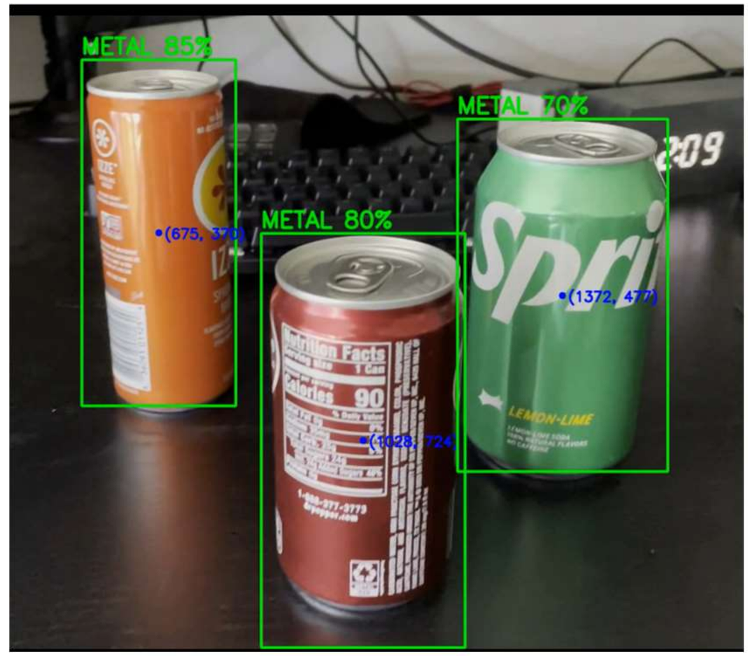
- Motor Control
  - Custom UART protocol for stepper motor commands.
  - ESP32 sends angle/speed; verified feedback from encoders.
  - Developed a web-based GUI for motor control Testing and Demo.
- Spectrometer Firmware
  - STM32 ADC + DMA handles CCD output.
  - Developed Python plotting tools for debugging.
  - Synced sampling timing for clean signals.
- Computer Vision
  - YOLOv8 and AWS Rekognition tested.
  - Detected object types and positions, aiming for a center point of (940,540).
  - Verified ROS integration and object-to-arm coordination.





Prototyping & Testing – Software

- Motor Control
  - Custom UART protocol for stepper motor commands.
  - ESP32 sends angle/speed; verified feedback from encoders.
  - Developed a web-based GUI for motor control Testing and Demo.
- Spectrometer Firmware
  - STM32 ADC + DMA handles CCD output.
  - Developed Python plotting tools for debugging.
  - Synced sampling timing for clean signals.
- Computer Vision
  - YOLOv8 and AWS Rekognition tested.
  - Detected object types and positions, aiming for a center point of (940,540).
  - Verified ROS integration and object-to-arm coordination.



Budget

- Optical components and Motors are the most expensive parts of this project.
- Our goal is to spend less than the budget we set as much as possible.

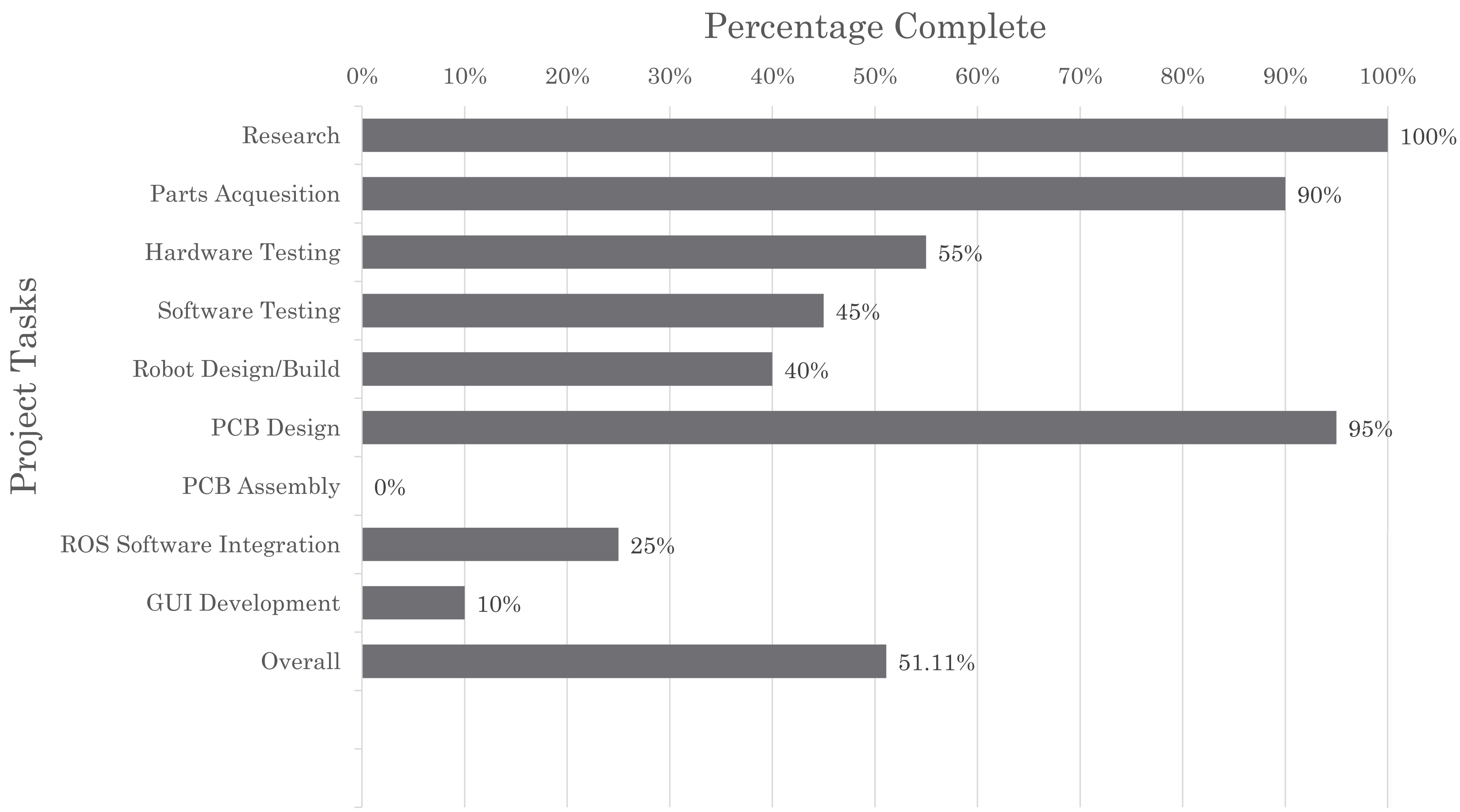
Sub-System	Description	Budget
Development	Dev board, jumper wires, and breadboards	$50
PCB and Electricals	PCBs, ICs, resistors, switches, etc.	$120
Motor System	Motors, drivers, and encoders	$220
Frame/Robotic Arm	Construction/assembly components, PLA filaments, mounting hardware	$180
Optics and Sensors	Sensors, lenses, light sources,	$700
Computer Vision System	Raspberry Pi, Camera, etc.	$140
Total		$1,410

BOM

- So far, we spent about $1400
- ~$620 in optical components

Item	Total Price
LEDs	$193.36
Slit	$137.08
CCD Sensor	$6.95
Lenses	$282.69
Stepper Motors	$66.00
Servo Motors	$41.97
Stepper Drivers	$33.00
Raspberry pi 5 kit	$135.99
Coral Accelerator	$85.59
Camera	$70.00
PCB	$60.00
ESP32	$2.95
STM32 Dev Board	$14.99
Electronic Components	$65.54
Power Supply	$18.99
Carbon Fiber Rods	$63.87
PLA Filament	$39.98
Bearings Pack	$17.59
Assorted Hardware	$60.00
Total	$1396.54

Timeline



Percentage Complete
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Project Tasks
Research
100%
Parts Acquesition
90%
Hardware Testing
55%
Software Testing
45%
Robot Design/Build
40%
PCB Design
95%
PCB Assembly
0%
ROS Software Integration
25%
GUI Development
10%
Overall
51.11%

Work Distribution

Jayden Lumas

- Machine Learning
- ROS Software
- GUI

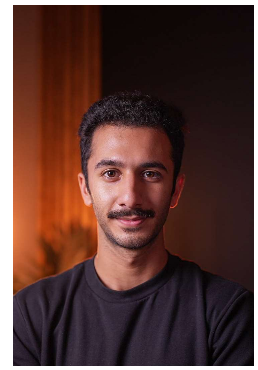
Dylan Bowers

- Spectrometer/Optical Design
- Material Identification Software

Tristan Koopman

- Robot Arm Design
- Housing Design
- Structural Build and assembly

Sayed Almousawi



- Hardware Design
- Linear CCD Sensor Driving Hardware/Software
- Motor Control Software
- PCB Design
- Wiring and Soldering
- Website