Capstone Final Report (Group 6)

AUTOMATED SPIN POLISH MACHINE

03 / 04 / 25

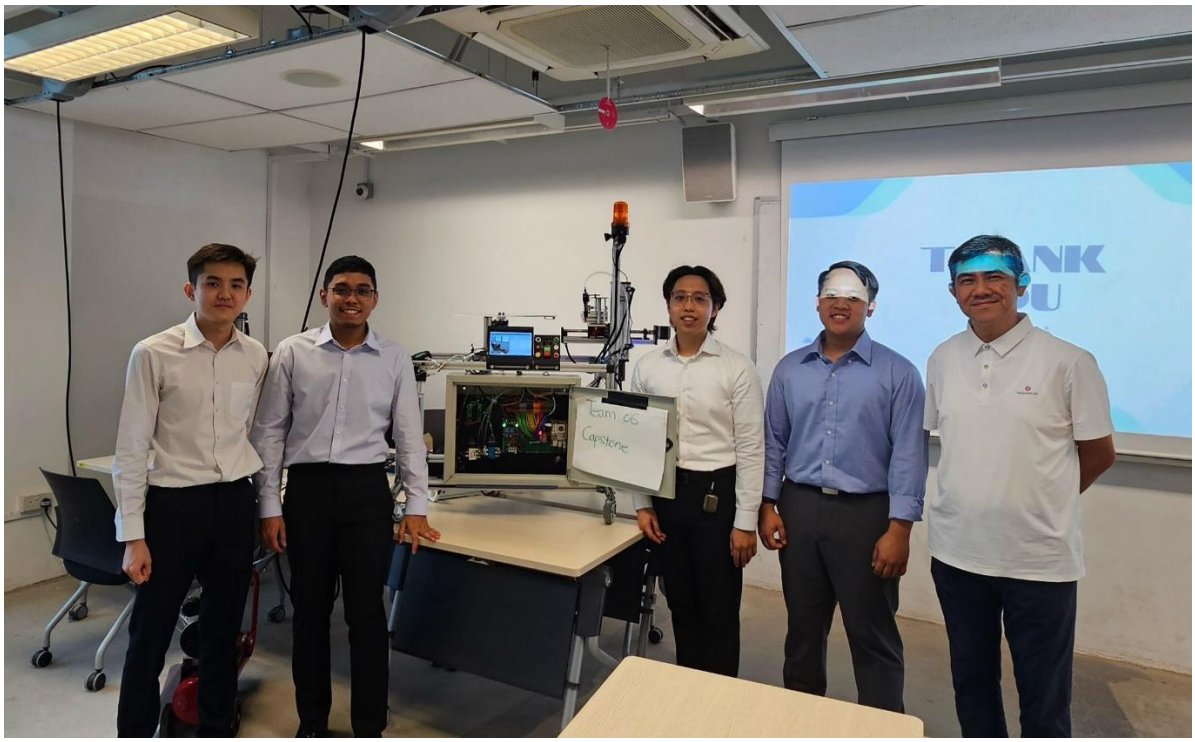
Supervisor :Rendell Tan

Muhammad Faiz Bin Mohammed Yacob	2102065@sit.singaporetech.edu.sg
Muhammad Danish Bin Abdul Falah	2101410@sit.singaporetech.edu.sg
Muhammad Asri Bin Boksenang	2101151@sit.singaporetech.edu.sg
Mok Teck Sng	2102512@sit.singaporetech.edu.sg

Table of Contents

Acknowledgement

We would like to extend our sincere gratitude to Mr. Rendell Tan for his invaluable guidance and support as our Capstone project supervisor. His timely feedback, technical insights, and encouragement throughout both phases of the project were instrumental in helping us navigate challenges and stay on track. We are truly grateful for his mentorship and commitment to our learning journey.



1. Executive Summary

The goal of this project was to create and develop an automated spin polishing machine to enhance the efficiency and uniformity of polishing Low-Pressure Turbine (LPT) discs, a process that is presently labour-intensive and carried out manually. In accordance with Industry 4.0 strategies, this system combines mechanical, electrical, pneumatic, and software elements to automate the polishing procedure with little operator involvement.

Throughout Capstone 1 & 2, the team prioritized developing a working prototype that could sustain steady pressure, handle different disc sizes, and provide an easy-to-use interface. The process included repetitive design, system integration, and testing stages, tackling both technical difficulties and system efficiency.

The final prototype successfully demonstrates the feasibility of automating the polishing process, meeting the core project objectives. This solution lays the

groundwork for future enhancements and potential real-world applications within the maintenance, repair, and overhaul (MRO) industry.

2. Capstone Background

2.1 Background Information

In the aerospace Maintenance, Repair, and Overhaul (MRO) industry, namely SAESL, the method of polishing Low-Pressure Turbine (LPT) discs is primarily done by hand. At present, technicians dedicate extended periods employing handheld sanding tools to eliminate micro-cracks and surface imperfections. This leads to operator exhaustion and creates variations in quality and efficiency. Additionally, the method does not conform to the increasing trend of automation associated with Industry 4.0. To tackle these challenges, our team aimed to create an automated spin polish system that simplifies this repetitive task and lessens the demand on skilled workers.

The project extends over two academic trimesters and adopts a systems engineering methodology to guarantee orderly development. By using this method, we have combined mechanical, electrical, pneumatic, and software subsystems into a unified and semi-autonomous machine able to manage various LPT disc stages.

Figures 1 (Left) & 2 (Right): Fig 1 shows a technician carrying out the polishing process with the sandpaper. Fig 2 shows the overview of the workstation for the polishing process

2.2 Project Statement

This Capstone Project aims to create and develop a prototype for an automated spin polishing machine that can polish LPT discs with steady pressure, adaptable geometry management, and limited human oversight. The drive arises from the sector's requirement to update and enhance repetitive tasks while ensuring safety and quality are not compromised. Our solution not only mirrors the efficiency of manual polishing but also improves it with precise control, immediate feedback, and uniform output quality.

Fig 3: An example of a Seal Fin, the part of the LPT Disc that would be sanded

2.3 Project Scope & Objectives

The primary objectives of this project are as follows:

- Automate the LPT disc polishing process to reduce manual labour and human error.
- Ensure consistent pressure application during polishing for a uniform surface finish.
- Accommodate multiple LPT disc sizes and geometries within a single system framework.
- Develop a Graphical User Interface (GUI) that allows operators to interact easily with the machine.
- Integrate all subsystems (mechanical, electrical, pneumatic, software) for seamless operation.
- Create a stable and scalable prototype suitable for demonstration and future enhancements.

This project's scope encompasses the design, creation, and validation of a working prototype that showcases the key attributes of a production-ready automated polishing system. The prototype functions as a proof of concept, demonstrating the possibility of automating the LPT disc polishing procedure while ensuring system stability, accuracy, and user-friendliness.

The initiative emphasizes multiple important aspects. To start, building a mechanical chassis with linear guides and actuators allows for accurate horizontal and vertical movement, essential for regulated polishing tasks. Stepper motor control is utilized to ensure precise and consistent positioning across different disc shapes. Simultaneously, a pneumatic pressure system is created to deliver steady force through an automated sandpaper mechanism, guaranteeing even polishing.

To support operator interaction and real-time monitoring, a Graphical User Interface (GUI) is designed, featuring live camera feedback and manual override controls for safety and debugging. The system also features a critical hardware upgrade migrating from a testbed breadboard to a fully integrated prototype board, improving durability, reliability, and cable management. The project culminates in system-level testing and integration, verifying that all subsystems operate together as intended and identifying potential areas for future refinement.

The project scope excludes certain aspects such as the creation of a full production-grade enclosure or housing, actual polishing of engine-certified LPT discs, and integration with external inspection or manufacturing platforms. These areas are beyond the academic scope of this prototype but are acknowledged as possible extensions for real-world deployment.

2.4 Stakeholder Requirements

From the group's consultations with the group's SAESL's point-of-contact, along with visual and industry references garnered during the Integrated Work Study Program with SAESL, several core requirements were identified:

- The system should be able to consistently polish LPT discs while complying with safety and quality standards.
- It must maintain equal pressure distribution across different LPT disc surfaces.
- It should be flexible enough to accommodate various disc sizes and configurations.
- A user-friendly interface (GUI) is necessary, requiring minimal training for operators.
- The system must incorporate safety features such as emergency stops and obstacle detection.

These requirements were used as guiding principles throughout system design, integration, and testing phases. Interim Trimester feedbacks from the project

supervisors makes up the requirements as well, with the feedbacks given timely in accordance with the group's progression.

2.5 Overview of Final System Architecture

The final system architecture consists of the following major subsystems.

Mechanical Subsystem: Includes a custom-built base and vertical frame, linear guides, stepper motors, and a spindle mounting mechanism for polishing. The system allows movement in both vertical and horizontal axes. It also has an attached rotating platform to mimic an LPT disc.

Electrical Subsystem: Comprises the Raspberry Pi 5 as the main controller, motor drivers, IR sensors for positional and safety feedback, and terminal blocks for efficient wiring.

Pneumatic Subsystem: Features an air compressor, solenoid valve, and a guided pneumatic cylinder for pressure regulation during the sanding process.

Software Subsystem: The software subsystem is at the heart of the polishing system, ensuring flawless coordination between numerous components for precise and efficient functioning. It was developed in Python and combines motor control, real-time monitoring, and automation to expedite the entire polishing process. The software provides smooth functionality while retaining high-quality results by precisely aligning the sandpaper, managing disc rotation, and enabling automatic sandpaper changes.

Safety is a critical component of the system, with integrated limit detection utilizing IR sensors to prevent misalignment and hand detection via a camera to minimize unforeseen mishaps. These characteristics work in real time, guaranteeing that the polishing process is both efficient and secure. The software's easy and responsive approach allows for semi-autonomous operation, decreasing user intervention while improving disc polishing consistency and dependability.

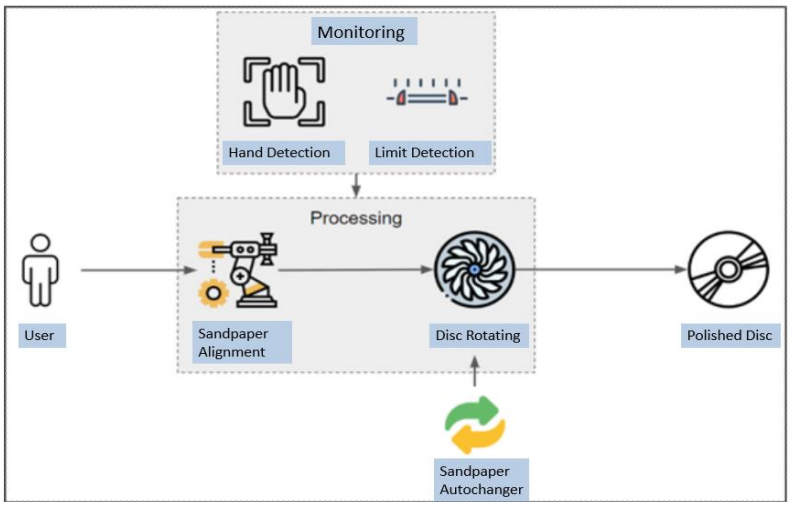


Fig 4: System Design & Architecture of Spin Polish Machine

3. System Design Overview

The Automated Spin Polish Machine was created by combining various subsystems, each serving a unique function to facilitate automated, reliable, and safe polishing of LPT discs. The design strategy emphasized modularity, accuracy, and dependability, making sure each part could be separately created, evaluated, and subsequently combined into a unified end system. During the project, meticulous focus was placed on the interaction between hardware and software components, alongside the necessity for real-time control, user safety, and operational adaptability. The upcoming sections offer an in-depth analysis of the mechanical, electrical, pneumatic, and software components that compose the entire solution.

3.1 Mechanical Subsystems

The mechanical subsystem forms the foundation of the Automated Spin Polish Machine. Precise movement in both horizontal and vertical axes is made possible by its sturdy base frame, vertical support structures, and linear guides. These movements are powered by stepper motors, which allow the sanding mechanism to be precisely positioned over the LPT disc. A spindle mount is incorporated into the design to support the sandpaper and provide constant pressure while in use. The frame's meticulous design ensures mechanical stability and reproducibility by managing vibrations and preserving alignment during the polishing process. Additionally, adjustability was taken into account, enabling the system to support different LPT disc sizes and configurations.

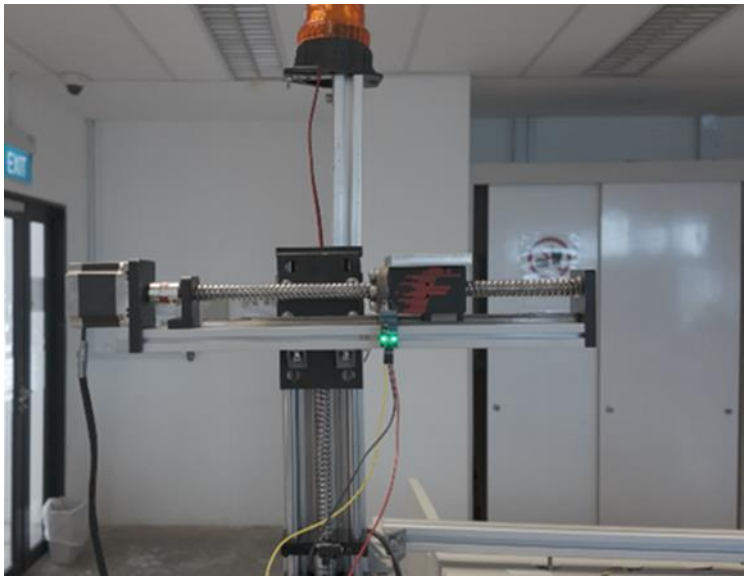


Fig 5: Horizontal & Vertical Mechanisms

3.2 Electrical & Pneumatic Systems

The electrical subsystem is built around a Raspberry Pi 5, which serves as the central controller for motor operations, sensor inputs, and the user interface. Stepper motor drivers, power regulators, and IR sensors are integrated into a compact prototype board, replacing the initial breadboard testbed. This upgrade has improved connection stability, cable management, and ease of debugging. IR sensors are positioned on the linear rails to serve as limit switches, enhancing operational safety and precision.

A pneumatic subsystem that controls the pressure used during polishing is a counterpart to the electrical system. Automated sandpaper deployment with regulated force is made possible by a guided pneumatic cylinder that is controlled by a solenoid valve that is attached to an air compressor. Through leakage testing and connector modifications, the team meticulously verified the air pressure system, guaranteeing steady and reliable pressure supply while in use.

Fig 6: Raspberry Pi with User Interface

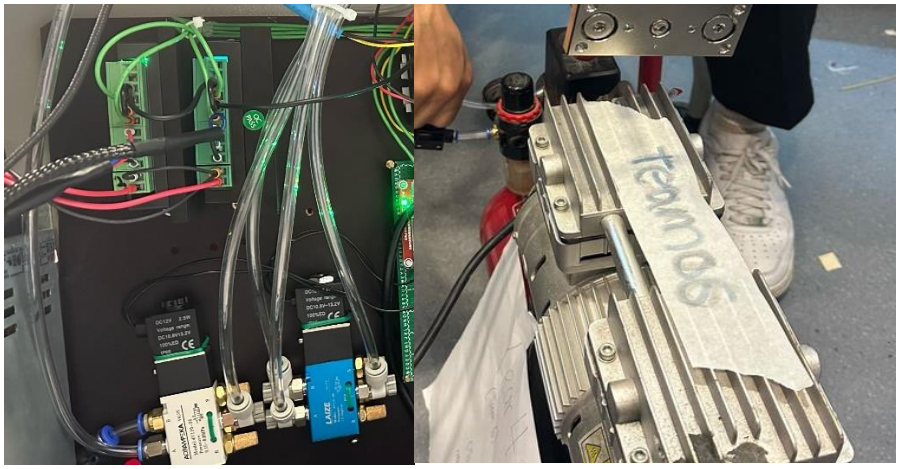


Fig 7: Air Compressor to the solenoids for the Pressure Regulation System

3.3 Software Architecture

The software architecture combines motor control, graphical user interface display, safety features, and image processing into a cohesive system. Created with Python, the codebase employs threading to handle various tasks at once on the Raspberry Pi. The graphical user interface offers operators immediate feedback from the camera stream, motor control options, and emergency stop features. The camera feed includes hand detection capabilities for enhanced safety, avoiding unintentional operation when obstacles are identified.

To improve system responsiveness, optimizations were made to the camera initialization process and detection threading. IR sensor feedback and stepper motor inputs are logged and processed in real-time, ensuring precise control and fail-safe conditions. The modular software structure allows for future extensions such as additional sensors or automated routines.

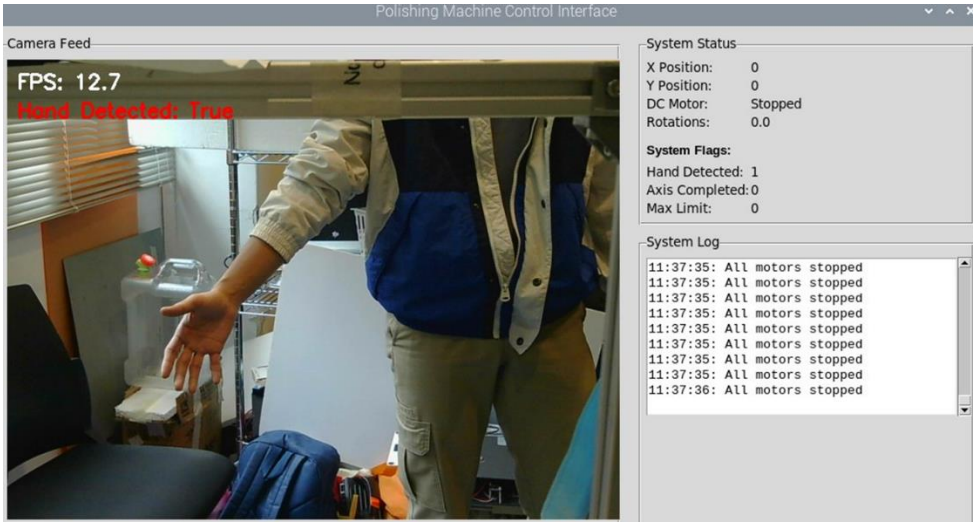


Fig 8: Screenshot of User Interface when Hand is detected

3.4 Final System Design

The completed system is a fully unified prototype that combines all mechanical, electrical, pneumatic, and software elements into a single cohesive unit. Affixed to a robust frame featuring a designated enclosure for electronics, the system enables secure, consistent, and semi-autonomous polishing of LPT discs. The interface enhances user engagement, while the pressure regulation system guarantees consistent sanding throughout the disc stages. Enhancements implemented during the project from testbed setups, to completely wired prototype boards, have boosted both system reliability and performance.

The design showcases a scalable solution that satisfies stakeholder needs and establishes a groundwork for future improvements, including automated calibration, enhanced camera responsiveness, and industrial-grade safety enclosures.

4. Staged System Integration

The development of the Automated Spin Polish Machine followed a staged system integration approach, allowing the team to validate each subsystem independently before progressing toward full integration. This methodology minimized risk and allowed for early identification of technical challenges across mechanical, software, electrical, and pneumatic domains. Each phase of integration was carefully planned and executed to ensure a smooth transition from individual components to a functioning prototype. The sections below outline the key milestones in the system integration journey.

4.1 Mechanical Foundation

The mechanical foundation formed the structural and functional basis of the Automated Spin Polish Machine. Initial prototyping was performed on a flat acrylic baseplate that held essential components such as stepper motors and a linear rail. However, this early setup proved inadequate in stability and lacked the structural support needed for scalable integration. Following feedback from Capstone 1 and early testbed insights, the team redesigned the base frame using aluminium T-slot profiles to support both horizontal and vertical motion. This modular design allowed for secure mounting of electrical components via a housing box and improved vibration resistance and layout flexibility. The updated frame structure accommodated increased loads, including an anticipated rotating disc platform, and supported future expansion.

4.1.1 Linear Rail Guide and Stepper Motor

The core of the movement system relied on linear rail guides driven by stepper motors. These allowed the polishing head to traverse across LPT disc surfaces in both horizontal and vertical directions with high precision. Mounting brackets and carriages were customized to ensure a tight fit and smooth operation on the linear rails. Microstepping control was implemented to achieve finer motion increments, which was essential for polishing consistency.

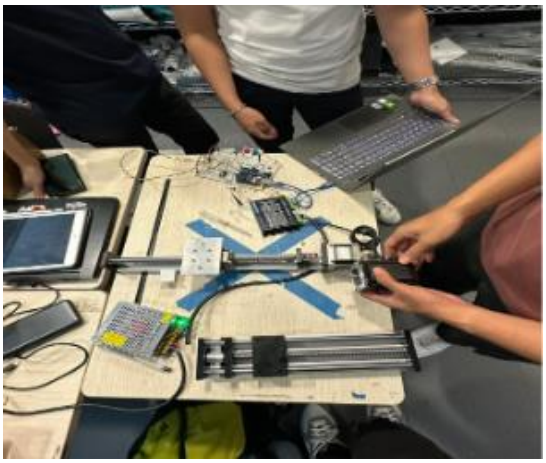


Fig 9: Linear Rail Guide with Stepper Motors being assembled

Challenges Faced:

During initial setup, improper rail alignment resulted in uneven motion and motor stalling. Additionally, the rails required repeated re-calibration to minimize lateral play, especially as the machine frame flexed under operational forces. The team had to rework their mounting methods using thicker support brackets and precise drill placements to secure the rails firmly. These adjustments were time-consuming but necessary for reliable polishing trajectories.

4.1.2 Horizontal Mechanism

The horizontal mechanism was designed to facilitate the lateral movement of the polishing tool across the LPT disc surface. This mechanism consisted of a stepper motor driving a linear rail and carriage system. The placement and calibration of this axis were essential, as it had to travel consistently and return to its home position without error. Limit sensors were later introduced on this rail to ensure the motor would not overextend. The system was optimized for smooth transitions and was tested under both idle and loaded conditions to assess responsiveness and repeatability. The movement also had to be smooth, accurate, and capable of returning to home position reliably.

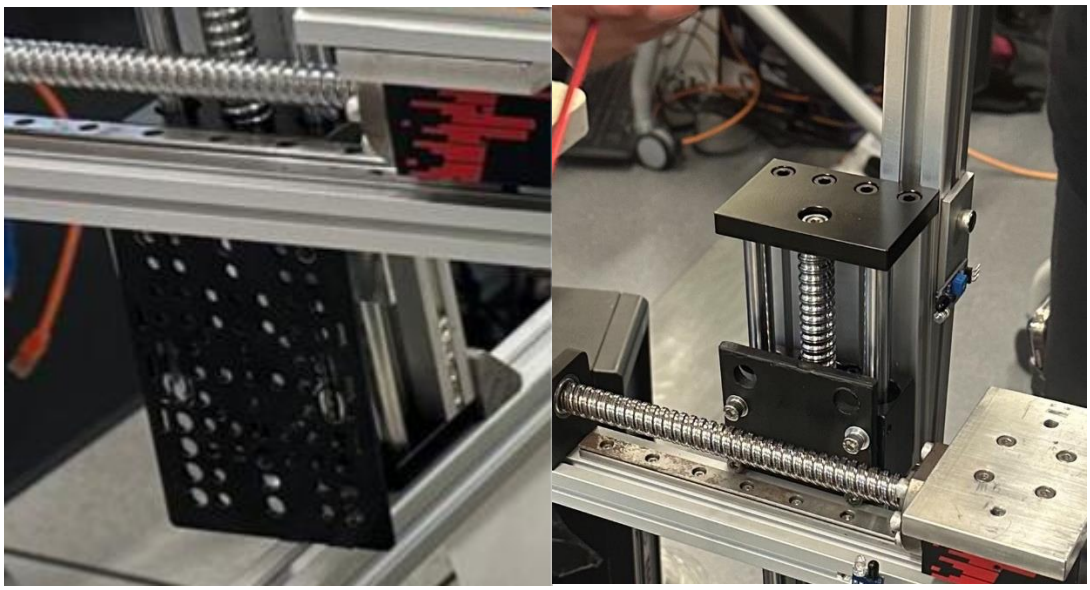


Fig 10(L) & 11(R): Fig 10 shows the previous custom plate used to attached the mechanisms. Fig 11 shows the improved and rigid metal plate used to attach the mechanisms.

Challenges Faced:

Due to inconsistencies in the metal plate used to mount the horizontal system during Capstone 1, tilting occurred after repeated use. This instability compromised polishing consistency and introduced positional drift. The team addressed this by fabricating a thicker, custom metal plate with additional M5 and M4 mounting holes to improve fastening and alignment. These efforts significantly enhanced the structural integrity and movement repeatability of the horizontal system.

4.1.3 Horizontal and Vertical Mechanism

The vertical mechanism was combined with the horizontal guide to enable vertical adjustments of the polishing head in order to completely support 2D movement over the disc surface. Both axes are in unison, with horizontal motion managing surface coverage and vertical motion dictating sanding height and depth. To make sure there was flexibility and clearance, the integrated system was tested using various disc sizes. In order to minimize vibration and misalignment that could result in uneven pressure or sanding marks, careful mounting and alignment were required to guarantee the perpendicularity between axes.

Challenges Faced:

One of the main challenges here was ensuring perpendicularity between the two axes. Even slight misalignment introduced angular deviation, leading to uneven pressure during polishing. The team spent considerable time adjusting mounting points and realigning the system using measurement jigs and manual testing. Furthermore, step loss during vertical motion was observed under load, which required recalibration of microstepping values and current limits on the motor driver.

4.1.4 Chassis

The main and base chassis played a central role in providing stability to the system. Initially, a flat acrylic platform was used during early prototyping. However, this setup lacked the structural rigidity and layout organization required for reliable long-term testing. As a result, the team redesigned the base using a square aluminum frame that could support additional components, such as the electrical box and rotating disc mount. This improved chassis allowed better component placement, protected exposed wiring, and significantly reduced system instability during operation.

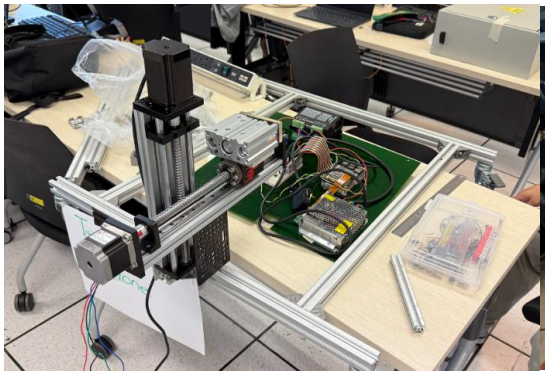



Fig 12(L) & 13(R): Shows the development of the Chassis

Challenges Faced:

Time constraints and structural fitment issues surfaced during the chassis upgrade. The bulky electrical box necessitated new support structures, and limited workspace at Catalyst Lab meant cutting, drilling, and assembly had to be precise on the first

attempt. Despite these hurdles, the frame was successfully completed with proper motor and wire routing considerations.

Fig 14: The Team Sawing off Aluminium Extrusion Slots

4.1.5 Electrical Housing Box

The updated design included an electrical housing box to safeguard and arrange delicate electrical parts, such power supply, motor drivers, and Raspberry Pi. These parts were previously exposed on the acrylic base, which raised the possibility of unintentional short circuits and disconnections. In addition to increasing safety, the housing box allowed for better cable management, less clutter, and quicker maintenance and debugging. The housing box was positioned on the frame's vertical side, offering accessibility and defence against harm from vibration.

Challenges Faced:

While beneficial in the long run, integrating the housing box introduced new design complexities. Mounting space was limited, and multiple iterations were needed to ensure cable routing paths aligned properly with motor locations. Additionally, the team had to redesign certain board layouts to accommodate vertical stacking and accessibility within the confined housing unit.

Fig 15: The limited space to house all components, including solenoids and Raspberry Pi

4.1.6 Rotating Platform for LPT Disc

The rotating platform that holds and rotates the Low-Pressure Turbine (LPT) disc during polishing is a crucial mechanical component of the system. By allowing the polishing head to stay motionless while the disc rotates underneath it, this platform replicates the rotating motion found in real turbine disc maintenance, resulting in uniform surface coverage.

The platform consists of a specially made circular acrylic base plate that is firmly fixed to a 12V DC gear motor that produces enough torque at low RPM to spin the disc smoothly under a range of polishing loads. To reduce vibration transfer, the plate is mechanically isolated from the frame and positioned in the centre beneath the vertical polishing head.

To ensure precise monitoring of the disc's rotation, an infrared (IR) sensor was strategically installed beside the rotating base. A reflective marker was placed on the edge of the rotating disc, allowing the IR sensor to detect and count revolutions in real-time. This input serves two main purposes:

- To log the number of polishing cycles, useful for repeatability and polishing consistency,
- To potentially trigger automation routines (e.g., stop rotation after a set number of revolutions).

The IR sensor connects directly to the GPIO pin on the Raspberry Pi and integrates with the logging system, recording revolutions with timestamped entries on the GUI.

Mechanically, the rotating platform is compact and centered within the chassis, minimizing lateral torque effects and ensuring stability. The design allows for easy removal and replacement of LPT discs, supporting future development toward a fully automated polishing station.

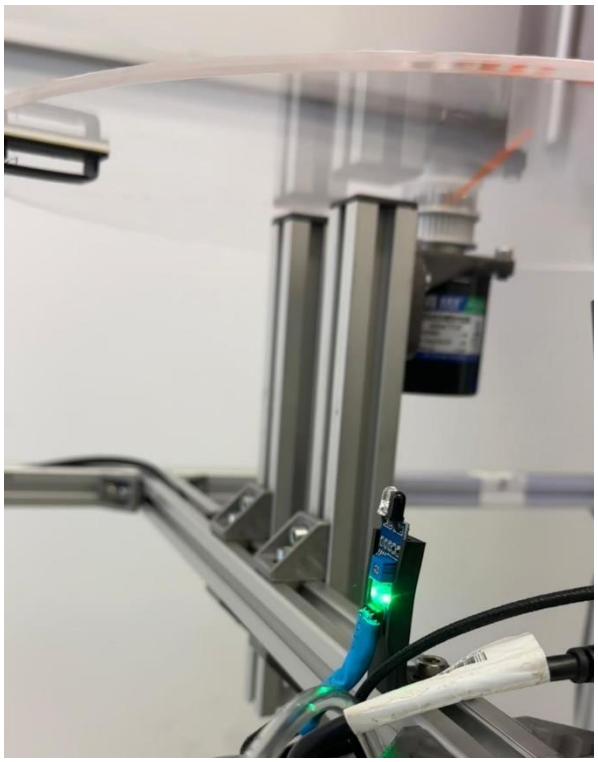


Fig 16: The Rotating Platform with attached IR sensor for rev-counting

4.1.7 Additional Feature: Sandpaper Auto Changer

The team managed to implement an Automated Sandpaper Changer that was integrated into the Spin Polish Machine. Although not listed as a stakeholder requirement, the team thought innovatively ways to make it automated in more aspects. The initial implementation of the Sandpaper Auto Changer held promise, and was immediately squeezed into the team's project timeline.

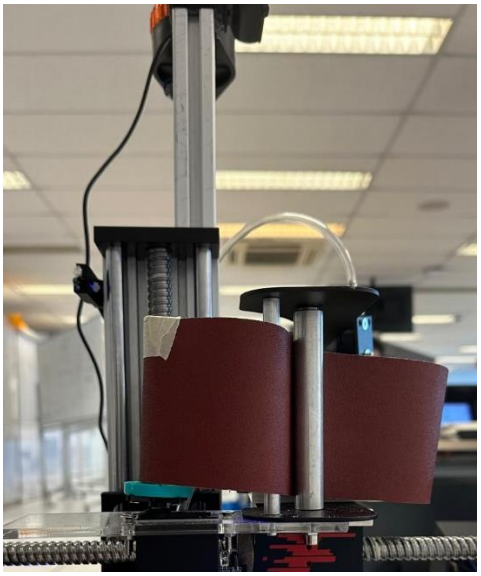


Fig 17: The Automated Sandpaper Changer

Challenges Faced:

Issues arise when figuring out the tension required to allow the sandpaper to be rigid during polishing, so that it does not create folds on the sandpaper and in turn, affect the quality of the polishing process. This was one of the challenges faced by the team that the team did not manage to fully solve. Although the team had integrated the Sandpaper Changer, after a numerous iterations the tension problems would surface once again. Due to the project's timeline being cut short to Week 10, the team decided to touch up on other important aspects of the project since this was an additional feature to begin with.

4.1.8 Mechanical Challenges Faced Summary

The mechanical development phase was shaped by both structural and integration challenges. Initial instability in the testbed setup revealed that acrylic lacked the rigidity for motor vibration and sanding pressure. Replacing it with a structured aluminum frame required rapid fabrication under tight deadlines. Component fitment and structural compatibility also became issues when the electrical housing box was added.

A significant learning point was the importance of tolerances and structural reinforcement, especially in the rail mounts and actuator plates. Improper hole alignment caused angular offsets that affected polishing paths, which were only discovered during full-axis motion tests. Furthermore, vertical clearance had to be adjusted multiple times to accommodate varying disc heights. These mechanical lessons directly improved the team's understanding of modular design, structural analysis, and rapid prototyping under realistic constraints.

4.2 Software Control Implementation

The software for the Automated Spin Polish Machine serves as the brain of the system, coordinating motor actuation, sensor input, GUI interaction, safety logic, and hand detection. As the project evolved in complexity, a shift from Arduino to Raspberry Pi 5 was necessary to support vision-based features, multi-threaded processes, and higher computational demands. This transition marked a pivotal point in the system architecture and development workflow.

4.2.1 Platform and Language Transition

Initially, the project was implemented using an Arduino board with C++ due to its real-time capabilities and suitability for hardware control. However, once the need for camera integration and hand detection was identified, the limitations of the Arduino platform became apparent. Arduino lacks the ability to process image data or support multi-threaded vision tasks. Therefore, the team migrated to a Raspberry Pi 5, which provided more powerful processing, USB camera support, and compatibility with Python-based vision libraries such as OpenCV.

Python was selected as the primary language due to its ease of implementation, robust community support, and availability of libraries for image processing, GPIO control, and GUI development. Although Python introduces performance trade-offs, its rapid prototyping capabilities made it the ideal choice for iterative system development.

4.2.2 Motion Control Architecture

Precise control over the polishing tool's position was achieved through a function named steps_calculation, implemented in Python. This function dynamically calculates the number of steps and direction required to move the toolhead to a desired coordinate. It factors in the current motor position, target multipliers, and motor characteristics. After executing movement, it updates the positional state of the system, ensuring synchronization between physical movement and internal logic.

This allowed the team to develop a flexible, disc-agnostic motion control approach where users could select different disc stages, and the system would compute motion ranges accordingly. The steps_calculation logic improved accuracy across different LPT disc stages and simplified debugging of position errors.

```python
def steps_calculation(self, button_state):
    """Calculate the steps for Y and X axes based on button state"""
    global current_X_position, current_Y_position

    # Calculate target positions
    target_Y_position = Y_baseSteps * Y_Multiplier[button_state]
    target_X_position = X_baseSteps * X_Multiplier[button_state]

    # Calculate steps needed
    Y_Steps = abs(target_Y_position - current_Y_position)
    X_Steps = abs(target_X_position - current_X_position)

    # Determine directions
    Y_direction = target_Y_position > current_Y_position
    X_direction = target_X_position > current_X_position

    # Update current positions
    current_Y_position = target_Y_position
    current_X_position = target_X_position

    return Y_Steps, X_Steps, Y_direction, X_direction
```

Fig 18: steps_calculation Function for Precise Motion Control

4.2.3 GUI Development and Integration

The GUI is a critical component, allowing users to operate the machine, monitor system feedback, and control motion manually or automatically. The team developed the GUI using Tkinter, Python's built-in GUI toolkit. Early versions of the GUI were built as a separate class, keeping the logic modular, but this led to performance problems, including sluggish responsiveness and update delays.

To resolve this, the GUI was restructured and tightly integrated with the main control loop. Event-driven programming using the after() method enabled non-blocking

updates. This dramatically improved GUI responsiveness, allowing it to function in tandem with ongoing processes such as video streaming, motor control, and sensor monitoring.

Real-time logging was also embedded into the GUI to capture button presses, movement status, and safety event triggers. This provided valuable feedback for debugging and improved transparency for operators.

4.2.4 Debugging Process and Development Workflow

Initially, all software components were written in a single script. However, this made debugging difficult because it was unclear if the faults were caused by fresh additions or pre-existing issues. To improve maintainability, the team used a modular testing technique in which specific components like motor control, IR sensor reading, and GUI rendering were first written as isolated test scripts. They were included into the main source after having been independently validated.

To improve stability during integration, try-catch blocks were added to the new process to handle failures that may arise during startup or when integrating components. This strategy significantly decreased integration failures and debugging time. It also increased traceability when bugs arose after new features were merged. By isolating components early on, the team was able to confidently test functionalities in a controlled environment and more effectively discover issues during integration.

4.2.5 Safety Enhancements in Manual Control

Manual control was made safer through the reimplementation of the _execute_manual_precision function. This function now includes logic that checks for both IR sensor values and hand detection status before permitting movement. If an obstacle or hand is detected, motor movement is blocked, a warning log is triggered, and an LED changes state to alert the user. This helps prevent unintended mechanical movement that might endanger the operator or damage the hardware.

This safety logic provided both hardware-level protection (via sensors) and software-level assurance (via pre-move checks), increasing overall system reliability.

```
def _execute_manual_precision(self, button_state):
    """Execute manual precision movement"""
    try:
        directions = {
            ManualXFront: (X_dirPin, X_stepPin, False, X_irsensor_front, "X Forward"),
            ManualXBack:  (X_dirPin, X_stepPin, True, X_irsensor_back, "X Backward"),
            ManualYUp:    (Y_dirPin, Y_stepPin, False, Y_irsensor_up, "Y Up"),
            ManualYDown:  (Y_dirPin, Y_stepPin, True, Y_irsensor_down, "Y Down")
        }

        for button, (dir_pin, step_pin, direction, ir_sensor, label) in directions.items():
            if button.is_pressed:
                self.log_message(f"Button Pressed: {label}")
                dir_pin.value = direction

                while button.is_pressed:
                    # Check if the IR sensor blocks movement
                    Operation_LED.on()
                    if ir_sensor.value == 0 or self.hand_detector.hand_detected :  # IR sensor detects an obstacle
                        self.log_message(f"Obstacle detected, movement blocked for {label}")
                        break

                    self.move_stepper(step_pin)
                break
        Operation_LED.off()

    except Exception as e:
        self.log_message(f"Error during manual movement: {str(e)}")
        self.run_stop_motors()
```

```
def _execute_manual_precision(self, button_state):
    """Execute manual precision movement"""
    try:
        directions = {
            ManualXFront: (X_dirPin, X_stepPin, False),
            ManualXBack:  (X_dirPin, X_stepPin, True),
            ManualYUp:    (Y_dirPin, Y_stepPin, False),
            ManualYDown:  (Y_dirPin, Y_stepPin, True)
        }

        for button, (dir_pin, step_pin, direction) in directions.items():
            if button.is_pressed:
                dir_pin.value = direction
                while button.is_pressed:
                    self.move_stepper(step_pin)
                break

    except Exception as e:
        self.log_message(f"Error during manual movement: {str(e)}")
        self.run_stop_motors()
```

Fig 19 & 20: Enhancing Safety and Debugging in Manual Precision Movement Execution

4.2.6 Software Challenges Faced

Despite successful implementation of the motion logic and GUI, the Raspberry Pi faced CPU strain when all tasks ran concurrently. Tasks such as camera streaming, GUI updates, hand detection, and sensor polling pushed the Raspberry Pi near its limits. Lag in GUI responses and motion stuttering occurred when the camera was active. This bottleneck was partially mitigated through threading optimizations, but it remains a limitation.

To address this, the team identified potential future upgrades, including offloading vision tasks to an external AI accelerator (e.g., Google Coral TPU or Intel Movidius). These devices could handle image inference independently, freeing the Pi to focus on motion and control tasks.

4.3 Camera System & Sensor Debugging

Both user safety and system responsiveness depend heavily on the camera system and sensor feedback loop. In addition to enabling hand detection algorithms to prevent dangerous motions, the camera gives the operator visual monitoring. When boundaries are crossed, motors are stopped by IR sensors, which act as physical limit switches. These systems work together to create the machine's safety and feedback layer.

4.3.1 Hand Detection and Initialization Problems

The Picamera2 library operates the camera system, maintains picture streaming, and provides real-time frame access to the hand detection module (via MediaPipe). Inadequate device indexing during early development prohibited the camera from initializing. The detection pipeline was unable to function due to a secondary issue with method misnaming in the thread configuration (detection_thread versus detection_process_thread). This caused crashes, stopped execution, and erratic video stream startup. These issues were resolved by standardizing thread initialization to ensure a reliable hand detection setup and placing the Picamera2 import into a try block to prevent crashes.

In addition, the usage of a pretrained model, which detected not only hands but also other object types, presented a hurdle. When only the hand class was detected, the sensitivity decreased noticeably—if a person held an object, the system frequently failed to register the interference. To improve sensitivity and safety monitoring, more object classes directly related with hand operations were added. This change increased the system's ability to recognize potential threats while remaining trustworthy.

```python
# Add this helper function at the module level
def check_camera_availability():
    """Check if any cameras are available and return their info"""
    try:
        from picamera2.picamera2 import Picamera2
        cameras = Picamera2.global_camera_info()
        print(f"Found {len(cameras)} cameras:")
        for i, cam in enumerate(cameras):
            print(f"Camera {i}: {cam}")
        return cameras
    except Exception as e:
        print(f"Error checking cameras: {e}")
        return []
```

Fig 21: Resolving Camera Initialisation & Threading Issues in Hand Detection

4.3.2 Image Quality and Lag Issues

There was obvious latency, blocky quality, and a blue tinge in the live feed. These resulted from ineffective threading, high frame resolution, and improperly adjusted white balance. The image pipeline was modified to include optimized threading for capture/processing balance, balanced frame rate (30 FPS), and frame resolution (640x480).

Despite these improvements, the Raspberry Pi still had trouble juggling live video with hand detection, which resulted in lag and stuttering. Despite a minor performance boost from threading, real-time responsiveness was still hampered by hardware constraints. The group pointed out that a 40–60% performance increase would be possible with edge computing or external GPU integration.

Fig 22: The Camera Feed's Blue Screen

4.3.3 IR Sensor Integration

To pinpoint the actual ends of motion, infrared sensors were installed on both axes of the linear guide rails. These sensors prevent mechanical overextension by alerting the controller to stop additional movement when they are triggered. In order to maintain sensor alignment and guarantee dependable triggering even during extended operation or vibration, custom 3D-printed holders were created.

These sensors function as safety checks in both manual and automatic motion modes and are directly connected to the motion control loop. Logging is used to record the coordinates and timings of sensor activation.

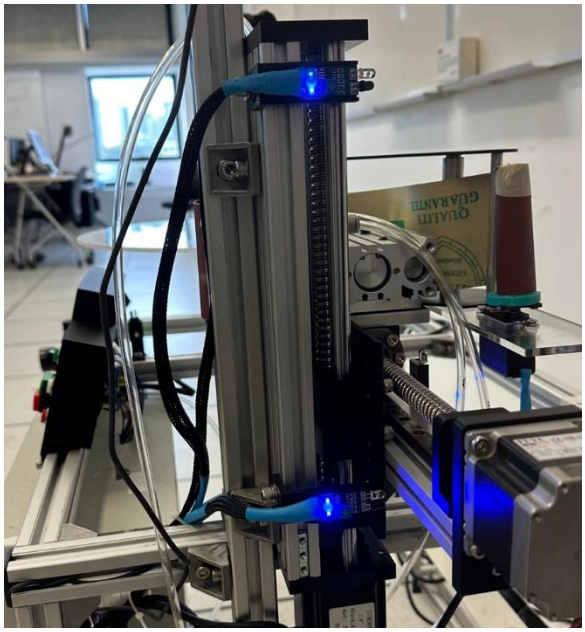


Fig 23: The IR Sensors integrated on the Vertical Mechanism

4.3.4 Challenges Faced

Synchronization problems resulted from the combination of real-time motion control and high-load vision processing. Occasionally, particularly in manual override mode, the system was unable to stop motion quickly enough when obstructions were identified. This necessitated balancing the allocation of CPU resources and improving the polling intervals.

Another persistent problem was IR sensor misfires and false positives. Reflective interference or misalignment was identified as the cause of this. In order to fix this, the team reprinted mounts with tighter tolerances and tested several sensor placements.

4.4 Transition to Prototype Board

As the electrical subsystem grew in complexity, along with feedback given from supervisors during Capstone 1, the team migrated from a fragile breadboard testbed

to a more permanent soldered prototype board. The board housed all key components, including the Raspberry Pi, motor drivers, IR sensors, and solenoid valve controller, which were soldered and routed through rail terminal blocks for durability and cleaner cable management.

4.4.1 Challenges Faced

The soldering process introduced new challenges. Accidental solder bridges and loose joints disrupted signal transmission, requiring hours of multimeter probing and re-soldering. Moreover, limited board space meant several components had to be repositioned and rerouted, leading to necessary changes in the wiring diagram. Voltage inconsistencies also arose due to poor initial grounding practices, prompting a complete rewiring of the power distribution system. These issues, while frustrating, taught the team valuable lessons in PCB layout, error handling, and electrical safety.

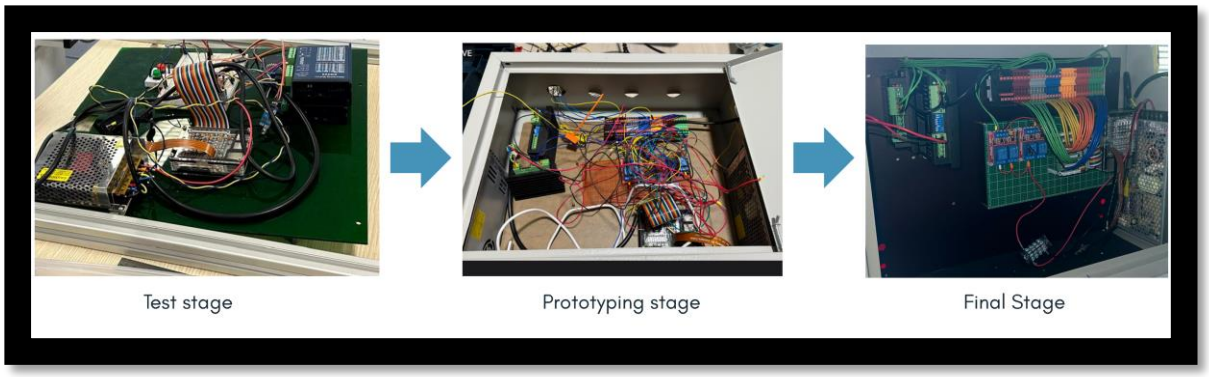


Fig 24: The evolution from test bed to prototype board

4.5 Pneumatic System & Pressure Control

The pneumatic subsystem was responsible for regulating downward polishing pressure using a solenoid valve and a guided air cylinder. This system was designed to operate at a calibrated 29 psi force, sufficient for polishing without damaging the disc.

4.5.1 Challenges Faced

A major bottleneck occurred due to the delayed arrival of the air compressor in Singapore, pushing the pneumatic testing schedule back by two weeks. During this downtime, the team used the opportunity to simulate pneumatic actuation and refine the mechanical mount design for the solenoid valve. Once testing resumed, issues such as air leakage at hose junctions and electrical inconsistencies in valve actuation surfaced. The team conducted resistance and current draw tests to validate solenoid performance and applied vibration-resistant fittings to eliminate leaks. These calibration and validation steps were critical to ensuring that the polishing mechanism could apply force reliably and consistently.

Fig 25: Different sized Valve Fittings

4.6 Full Subsystem Integration

The final integration phase brought together the mechanical frame, electrical system, pneumatic actuator, camera module, and GUI into a single working prototype. This process involved verifying that all systems could operate concurrently and safely, while responding to user inputs through the control interface.

4.6.1 Challenges Faced

New interdependencies that were not apparent during subsystem testing were discovered when all systems were combined. Particularly when switching between manual and automatic modes, conflicts arose between motor commands and camera processing threads. More effort was needed to improve error reporting, synchronize subsystem timings, and confirm that limit sensors and emergency stop procedures operated dependably in practical settings.

Additionally, minor variations in pressure supply, mechanical backlash, and sensor offsets have to be taken into consideration during final calibration. The team was able to prove the viability of a completely automated polishing process and validate system-wide functionality in spite of these obstacles.

5. Technical Details & Layout

5.1 CAD Overview

Computer-Aided Design (CAD) models were developed throughout the project to visualize, plan, and iterate the mechanical and sensor integration of the automated spin polish machine. These models served as the basis for fabrication, system alignment, and part interfacing.

5.1.1 CAD of Spin Polish Machine

The CAD model of the spin polish machine was developed using CREO and it provides a full mechanical overview of the final system layout. The model features:

- Horizontal and vertical linear rails, driven by stepper motors, responsible for moving the polishing toolhead with precision.
- A rotating circular platform, representing the LPT disc mount, supported by a frame underneath to allow free rotation during polishing.
- The chassis structure, made from T-slot aluminum extrusions, provides a modular and rigid frame for component mounting and future scalability.
- A box enclosure, representing the electrical housing area, mounted at the rear-left side of the frame, housing the Raspberry Pi, relay modules, solenoid controls, and power supplies.

The CAD design helped verify mechanical clearances, stepper motor travel ranges, and sensor placement. It also guided the physical fabrication, ensuring proper dimensions and alignment before cutting or drilling components.

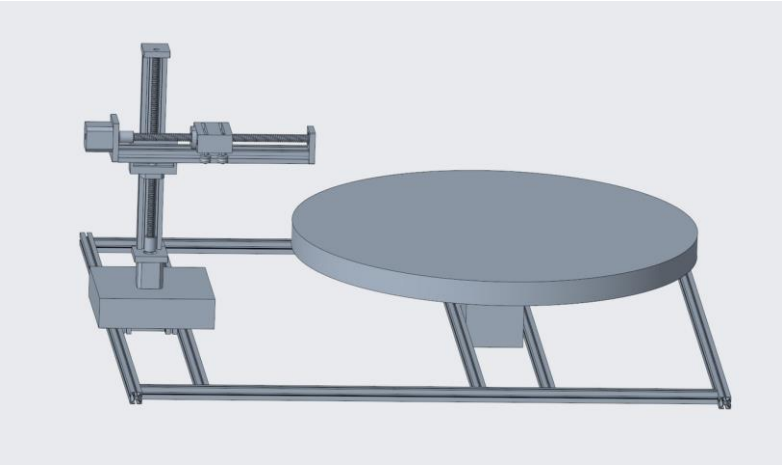


Fig 26: CAD of Spin Polish Machine

5.1.2 CAD of Sensor Holders

To ensure consistent and accurate placement of IR sensors on the linear guide rails, custom sensor holders were designed and 3D printed. These holders were critical to

aligning the sensors at the correct angle and distance relative to the moving gantry, enabling reliable limit detection during motion.

The sensor holders were designed to:

- Securely clamp onto the aluminium extrusion rails without the need for drilling.
- Provide a fixed orientation for IR sensors to prevent drift or misalignment due to vibration.
- Include slotted holes for minor positional adjustments during calibration.
- Be 3D printable in lightweight and durable PLA or PETG materials.

Using CAD for sensor holder design allowed rapid prototyping and iterative testing. Multiple revisions were made to adjust tolerances, fitment, and cable routing paths before the final version was selected for deployment.

Fig 27: CAD of Sensor Holder

5.2 Prototype Board Layout

The goal of the prototype board layout was to combine all of the important electrical and control parts into a neat, small, and easily maintained enclosure. Eliminating breadboarding instability, streamlining cable routing, and enhancing system adaptability for debugging or future extension were the objectives.

Before the actual wiring and soldering started, the layout was first designed in a diagram format. This minimized electromagnetic interference and cross-wiring while guaranteeing effective use of internal space. Each of the main subsystems that are integrated into the board is broken down below.

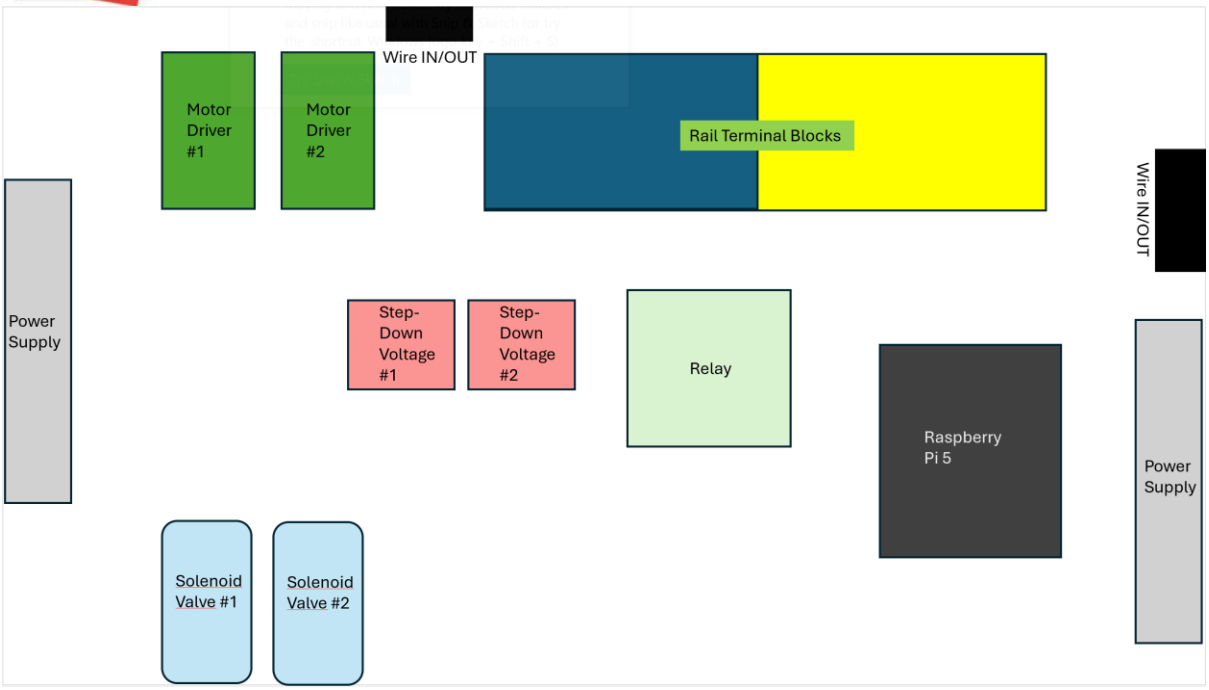


Fig 28: Mock-up of Board Layout

5.2.1 Power Supply Units

Two switching power supplies are positioned on opposite ends of the enclosure:

- The left PSU supplies 12V or 24V for high-power components like stepper motor drivers and solenoid valves.
- The right PSU, closer to the Raspberry Pi, likely provides regulated 5V or serves as a backup or isolated supply for logic-level circuits.

Each PSU is properly ventilated and grounded, with protective fusing and filtered AC inputs to ensure safe operation.

5.2.2 Step-Down Voltage Modules

Two DC-DC buck converters are mounted in the centre of the board. These modules step down the 12V/24V supply to:

- 5V for logic-level relays, IR sensors, and camera systems.
- Additional regulated voltages if required for safety LEDs or other modules.

Their positioning allows short routing paths to both the relays and GPIO headers, reducing power loss and electrical noise.

5.2.3 Motor Driver Units

Two stepper motor drivers are mounted on the left side of the board. These drivers receive control signals from the Raspberry Pi (through GPIO relay switching) and draw power from the primary PSU. Proper wire shielding and grounding were applied to prevent feedback from interfering with camera or sensor modules.

5.2.4 Solenoid Values

Located near the bottom-left of the board are two solenoid valves, used to control pneumatic pressure delivery for polishing. These are activated through the relays and powered by the main PSU. Clear air tubing and check valves are visible, with electrical wiring running back to the relay control board.

5.2.5 Relay Modules

The relay module is centrally placed for optimal signal routing. It serves as an interface between low-voltage GPIO control signals and high-power devices like solenoids and possibly future DC pumps or actuators. LED indicators on each channel provide visual feedback for actuation status.

5.2.6 GPIO Ribbon Interface & Terminal Block

A rainbow-colored GPIO ribbon cable runs from the Raspberry Pi to a soldered GPIO breakout board on the board. This board distributes GPIO signals to a set of rail-mounted terminal blocks located along the top of the enclosure. These terminal blocks allow secure and organized IN/OUT wiring to external components (sensors, motors, switches), with color-coded paths to minimize wiring errors.

5.2.7 Raspberry Pi 5

Mounted on the right side, the Raspberry Pi 5 is housed in a transparent cooling case with an active heatsink/fan. It is the central controller of the system, running the full Python software stack (GUI, motor control, camera feed, sensor logic). It connects to the power supply, the prototype board, and peripherals via the GPIO ribbon, HDMI, and USB inputs.

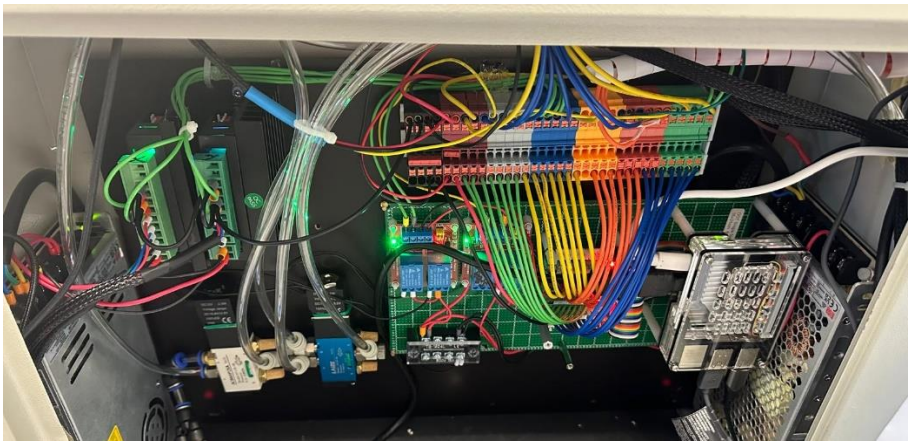


Fig 29: Eventual board layout of Spin Polish Machine

5.3 Cable Management Plan

Effective cable management was essential in ensuring that the automated spin polish machine remained safe, maintainable, and electrically reliable throughout its development and operation. With a wide array of GPIO signal lines, power feeds, and relay control paths, a structured cable plan was implemented to avoid issues like signal noise, accidental disconnections, and difficulty in debugging.

5.3.1 GPIO-to-Function Mapping

To maintain clarity, each GPIO pin from the Raspberry Pi 5 was assigned a specific function. The assignments were planned in a dedicated spreadsheet and visually color-coded for ease of reference. These pins connect through a rainbow GPIO ribbon cable into the prototype board and then fan out to a row of terminal blocks, which distribute signals to sensors, relays, LEDs, and motors.

The mapping chart on the right (Figure 29) clearly links each GPIO number to its corresponding hardware function, such as:

- Push buttons (PB_) for manual control,
- Sensor inputs (e.g., IR and TOF),
- Motor direction and step signals (XDIR, YDIR, etc.),
- Solenoid triggers and servo control.

Unused or placeholder GPIOs (marked as spoil) are reserved for future features like polishing head automation and can be easily wired in without repinning the entire terminal block.

5.3.2 Layout and Routing

Figure 28 shows the actual cable layout corresponding to the planned GPIO table. Color-coded wires from the GPIO breakout are routed into labelled terminal blocks. Wires are grouped according to function:

- Motor control wiring is grouped toward the left.
- Sensor feedback and safety-related GPIOs are centralized.
- Peripheral actuators like solenoids and relays are routed from the centre toward the rear left of the board.

Coloured heat-shrink tubing and pre-crimped ends were used to identify wire functions during physical installation. Cable lengths were kept minimal to reduce slack and risk of tangling.

5.3.3 Physical Cable Organization

- Zip ties and adhesive mounts were applied to route bundles along chassis walls and reduce cable strain.
- Braided cable sleeves were used for high-voltage or relay wiring to prevent insulation wear and improve electrical safety.
- Right-angle routing paths helped prevent overlapping and interference, especially where ribbon cables pass over relay logic lines.
- Terminal blocks were split by voltage zones: 5V logic-level signals vs. 12V or 24V actuator lines, minimizing EMI and heat buildup.

5.3.4 Maintenance & Troubleshooting Benefits

This cable management approach made maintenance and troubleshooting much easier. The color-coded scheme allowed for the rapid isolation of problematic wires based on their GPIO reference. Labelled blocks and grouped wire bundles avoided confusion or cross-connection when relay channels were being debugged or components were being replaced. Additionally, this layout requires little rework for future improvements.

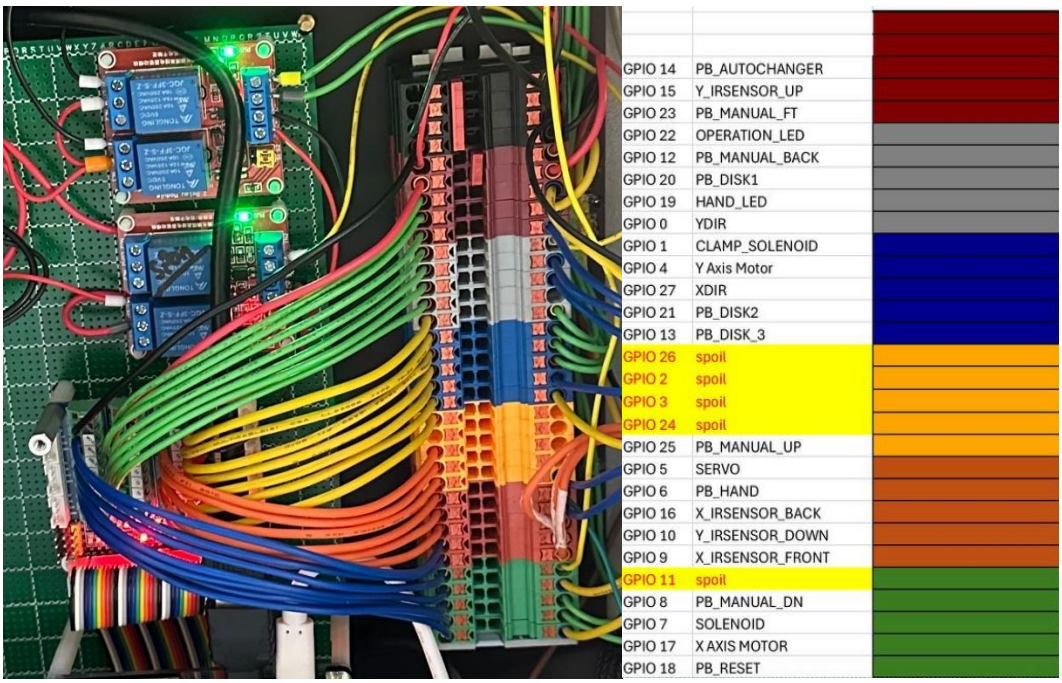


Fig 30(L) & Fig31(R): Fig 30 shows the actual layout and orientation depicted from Fig 31's planning of Cable Management

5.5 Pneumatic System & Pressure Control

The pneumatic system in the automated spin polish machine plays a crucial role in delivering controlled, repeatable pressure to the polishing head. This is essential to ensure uniform sanding across LPT disc surfaces, regardless of disc geometry or surface roughness.

To accomplish this, the system was designed with a closed-loop air supply setup that included:

- An external air compressor (providing ~90–100 psi),
- A pressure regulator with filter and water trap to ensure clean and stable pressure delivery,
- A pair of digital and analog pressure gauges for monitoring air pressure near the polishing unit and at the compressor source,
- Solenoid valves and tubing connectors for directional control and distribution of pressurized air.

5.5.1 Compressor & Main Regulator Setup

The air compressor (shown in Figure 31) has a secondary regulator and a physical analog gauge. In addition to allowing coarse pressure adjustment, this regulator makes sure that the compressor output stays within the bounds of the air fittings or solenoid valves.

Directly on the compressor head was a major regulator equipped with two gauges: one for output and one for tank pressure. From here, the polishing system installed on the machine chassis is fed by a polyurethane tubing line.

5.5.2 On-board Pressure Regulation and Monitoring

A secondary regulator unit with digital and analog gauges was placed close to the machine chassis to enable precise control and proximity-based pressure monitoring (Figure 30). This device has the following features:

The team was able to tune pressure to about 29 psi, the optimal amount found for consistent polishing force, thanks to a digital gauge that provides accurate PSI readouts.

A manual regulator knob that allows the pressure at the point of actuation to be changed in real time. Long-term dependability is increased by preventing moisture accumulation in the solenoid or actuator with a moisture filter and water trap.

This regulator's output divides into two lines, which are then connected to two solenoid valves located within the electrical housing. The sandpaper is pressed onto the disc surface by means of pneumatic cylinders that are operated by these valves.

5.5.3 Solenoid Valve Control and Integration

The solenoid valves are electrically actuated via the relay module connected to the Raspberry Pi. These relays toggle air supply to the pneumatic cylinders based on software-defined commands or operator input via the GUI.

During testing, multiple validation methods were used to ensure system functionality:

- Resistance measurements and continuity checks on solenoid terminals,
- Click tests to confirm valve actuation,
- Air flow and pressure retention checks after fitting the connectors.

Air leakage was encountered during early integration due to mismatched fittings and inadequate Teflon sealing. These were resolved using proper pneumatic quick-connect joints, re-cut tubing, and resealing techniques. Once stabilized, the system was able to maintain pressure consistently and respond promptly to GPIO trigger signals.

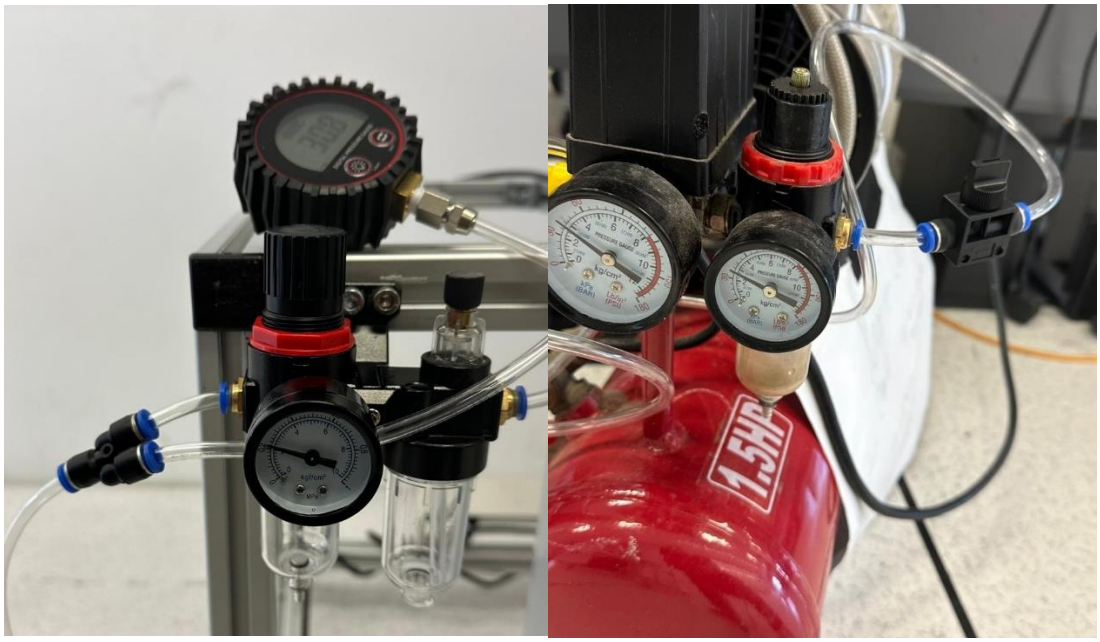


Fig 32(L) & Fig 33(R): Fig 32 shows a digital gage connected near the chassis whereas Fig 33 shows another physical gage on the Air Compressor.

5.6 Updated Theoretical Calculations of Spin Polish Machine

In order to make sure the structure could support additional components added during full integration, the team updated the mechanical load analysis in this updated build of the spin polishing machine. In contrast to the previous prototype, this version includes a more robust polishing arm, solenoid valve assembly, air fittings, 3D-printed camera mount, and a more complete motor setup; these modifications necessitated an updated moment equilibrium analysis to check for toppling risk when the polishing arm is fully extended.

System Setup:

- Horizontal Mechanism Length (L): 400 mm
- Vertical Mechanism Length: 500 mm
- Attachment Height of Horizontal Arm from Base: 250 mm
- Mass of Vertical Mechanism: 2.2 kg
- Mass of Horizontal Mechanism: 2.3 kg
- Mass of Mounting Plate and Electrical Accessories: 0.6 kg
- Mass of Polishing Arm Assembly (incl. solenoids, 3D-prints, etc.): 0.85 kg

Total Weight of Fixed Mechanism:

$$W_{Combined} = W_{Vertical} + W_{Horizontal} + W_{Plate} = 2.2 + 2.3 + 0.6 = 5.1kg$$

Resisting Moment:

The resisting moment is generated by the total combined weight acting at its centre of gravity. Assuming the CoG lies approximately halfway up the vertical mechanism (250 mm from the base):

$$M_{Resist} = W_{Combined} \times g \times d_{Resist}$$

$$M_{Resist} = 5.1 \times 9.81 \times 0.25 = 12.50Nm$$

Applied Moment:

The polishing arm applies a moment at the very end of the horizontal rail. Using the arm's mass:

$$M_{Applied} = W_{Arm} \times g \times L$$

$$M_{Applied} = 0.85 \times 9.81 \times 0.40 = 3.34Nm$$

Since:

$$M_{Applied} = 3.34Nm < M_{Resist} = 12.50Nm$$

The system remains stable under static conditions.

Tipping Load Threshold:

To determine the critical load that would cause the system to topple (e.g. from unexpected external forces or heavier polishing heads), we rearrange the moment equation:

$$W_{Tip} = \frac{M_{Resist}}{g \times L} = \frac{12.50}{9.81 \times 0.40} = 3.19kg$$

This means any additional load exceeding 3.19 kg at the tip could cause toppling and must be avoided.

Under the present load distribution, the upgraded system which includes all extra electrical, pneumatic, and polishing components is structurally stable. However, to provide mechanical stability for upcoming improvements like larger polishing tools or motorized sandpaper changers, it could be required to reinforce the base or redistribute mass. Safe design thresholds for subsequent iterations are guided by these calculations, which also validate the system's robustness for present use.

6. Timeline & Resource Management

Resource planning and timeline tracking were essential throughout the Capstone 2 phase, enabling the team to prioritize critical system components, respond to delays, and manage budget constraints effectively. This section summarizes the project's execution status, resource allocation, procurement approach, and key materials used.

6.1 Gantt Chart Timeline

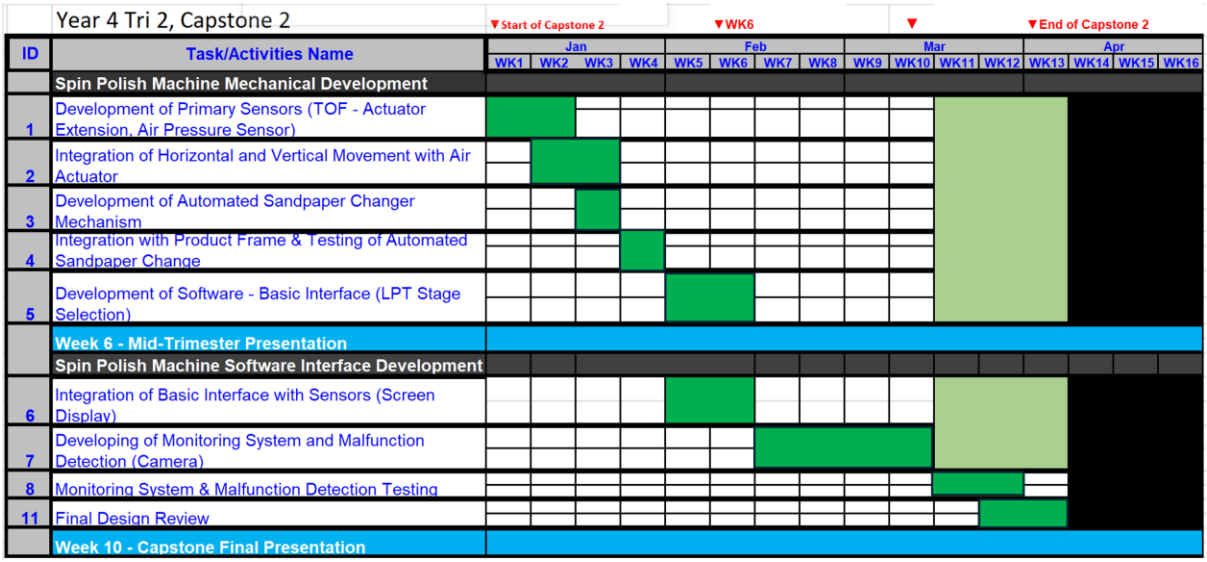


Fig 34: Gantt Chart of Capstone 2

The Gantt chart outlines the team's progress over the trimester, with tasks mapped across hardware development, software integration, testing, and validation phases.

Most major deliverables were completed within the allocated timeframe. Notable milestones achieved include:

- Completion of the main mechanical structure, including the chassis and motion system.
- Integration and testing of the Raspberry Pi 5 software stack, including GUI, motor control, and sensor logic.
- Full development of the pneumatic polishing mechanism, with solenoid valves and pressure regulation.
- Successful setup of the camera system, safety sensors, and electrical housing.
- Transition from breadboard to a fully wired and soldered prototype board.
- Assembly of a functioning prototype capable of performing polishing operations across different LPT disc stages.

One subsystem, the sandpaper auto-changer was partially functional. While the motor and clamping logic worked in isolated tests, the system experienced inconsistencies over longer operation cycles. Due to time constraints, full refinement was not achieved by the final week. The feature currently works intermittently and is flagged in the system as experimental for future iteration.

6.2 Budget Usage

A total budget of $500 SGD was allocated to the team. Expenditures were distributed across mechanical hardware, electrical components, pneumatics, and sensor systems. The largest cost areas were related to:

- Raspberry Pi 5 and accessories (camera module, cooling case, SD card)
- Stepper motors and drivers
- Linear guide rails and actuators
- Solenoid valves and pneumatic regulators
- Prototype board components (terminal blocks, relays, buck converters)

The team spent approximately $940–$950, staying within budget. Strategic reuse of components from Capstone 1 helped reduce costs. Some accessories (Aluminium Extrusion Slots, Extra Wiring) were obtained via in-lab stock or minimal-cost sources to minimize excess spending. The remaining funds were reserved for optional add-ons or final adjustments.

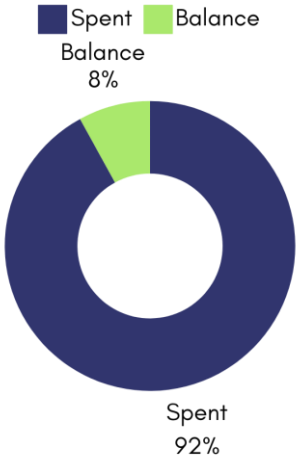


Fig 35: Overall Budget Expenditure

6.3 Procurement Strategy

The procurement strategy followed a one subsystem at a time method. Items were procured in phased batches based on:

- The progress of each subsystem (mechanical first, software last),
- Validation of parts compatibility,
- Availability from local or online vendors.

Primary sourcing platforms included Lazada, Shopee, RS Components, and local electronics stores. Components that required compatibility testing (Solenoids, sensor modules) were ordered early to allow time for return or adaptation. Delays in shipping (notably the air compressor and certain fittings) influenced development pacing, and backup options were considered to prevent integration standstills.

All orders were tracked and logged in a shared spreadsheet, which helped in estimating lead times, costs, and cumulative spending.

6.4 Bill of Materials List

The Bill of Materials (BOM) provides a comprehensive breakdown of all the components used in the final build of the automated spin polish machine. The system incorporates a wide range of mechanical structures, electrical modules, pneumatic components, and fastening hardware to support modular assembly, control, and polishing functionality. Key structural elements include T-slot aluminum extrusions and linear guide rails, which form the machine's rigid frame and support precise movement along the horizontal and vertical axes. High-torque NEMA17 stepper motors, coupled with threaded rods and timing belt systems, drive the polishing mechanism.

From an electrical perspective, the integration of a Raspberry Pi 5, GPIO breakout, relay modules, and LM2596 step-down converters enables safe and isolated power delivery to solenoids, sensors, and motor drivers. The pneumatic subsystem is built around a guided rod cylinder actuated by two 12V solenoid valves, regulated through a digital and analog pressure control setup. The air delivery path is reinforced by moisture filters and polyurethane tubing to ensure smooth actuation.

Fasteners such as M3 and M5 hex bolts, nuts, washers, cable ties, and adhesive mounts ensure physical reliability, while custom 3D-printed parts like camera mounts, sensor holders, and enclosures offer mechanical adaptability and compact integration. Overall, the BOM captures the multidisciplinary nature of the project, combining mechanical, electrical, and embedded systems into a cohesive and scalable design, with an estimated total component weight exceeding 5 kg and a total 3D printing time of over 6 hours.

Legend:

MXX ID: (Mechanical & Electrical Components): Structural frame parts, motors, electronics, sensors, pneumatic actuators, and power systems essential for machine function.

FXX ID: (Fasteners & Accessories): Screws, nuts, washers, cable ties, and mounting aids used for securing mechanical and electrical components.

PXX ID: (3D-Printed Parts): Custom-designed PLA components like holders, mounts, and enclosures that support sensors, wiring, and subsystem integration.

ID	Part Name	Material	Quantity	Measurements	Quantity	Weight (g)
M01	Main Frame Rail	T-Slot Aluminum (2020)	4	500mm	4 × 500mm	752
M02	Cross Beam Support	T-Slot Aluminum (2020)	2	300mm	2 × 300mm	282
M03	Vertical Column	T-Slot Aluminum (2020)	1	500mm	1 × 500mm	188
M04	Horizontal Slide Rail	Linear Rail + Carriage	1	400mm	400mm	120
M05	Vertical Slide Rail	Linear Rail + Carriage	1	300mm	300mm	90
M06	Threaded Rod (Z-axis)	M8 Threaded Rod	1	150mm	150mm	34
M07	Polishing Arm	Aluminum Plate (3mm)	1	250mm × 40mm	1	78
M08	Disc Holder Plate	Aluminum Plate (5mm)	1	Ø180mm	1	155
M09	Stepper Motor (X-axis)	NEMA17	1	-	1	210
M10	Stepper Motor (Z-axis)	NEMA17	1	-	1	210
M11	Polishing Platform Motor	High-Torque DC Motor	1	-	1	320
M12	Coupler	Motor Shaft to Rod	2	5mm–8mm	2	24
M13	Pulley & Belt (X-axis)	GT2 20T + Belt	1	400mm belt	1	45
M14	Limit Switch (IR)	Infrared Reflective Sensor	4	-	4	10
M15	Camera Module	Raspberry Pi Camera	1	-	1	30
M16	Solenoid Valve	2/2 Normally Closed (DC 12V)	2	-	2	185
M17	Pneumatic Cylinder	Guided Rod Air Cylinder	1	Stroke: 50mm	1	210
M18	Pressure Regulator	Digital + Analog	1	0–100 psi	1	150
M19	Air Tubing	PU Tube	1	6mm OD × 4m	4m	80
M20	Power Supply	24V 10A Switching PSU	1	225mm × 115mm × 50mm	1	425
M21	Raspberry Pi 5	SBC	1	With cooling case	1	150
M22	Step-Down Module	LM2596 Buck Converter	2	12V→5V	2	35
M23	Relay Module (2-Channel)	5V Relay	2	10A max	2	75
M24	Prototype PCB Board	FR4	1	10×15 cm	1	50
M25	GPIO Ribbon Cable	40-pin to terminal block	1	300mm	1	48
M26	Rail Terminal Block Strip	DIN Mounted Terminals	1	24 terminals	1	125
F01	M4×10 Hex Screw	Stainless Steel	60	0.95g/pc	60	57
F02	M5×16 Bolt	Zinc-plated Steel	30	1.5g/pc	30	45
F03	M3×8 Screw	Stainless Steel	20	0.5g/pc	20	10
F04	M3 Nut	Stainless Steel	40	0.4g/pc	40	16
F05	M4 Washer	Zinc-Coated	40	0.3g/pc	40	12
F06	Cable Ties (Small)	Nylon	30	100mm	30	15
F07	Cable Tie Bases	Adhesive Mount	12	25mm	12	24
F08	Heat Shrink Tubing	2.5mm / 4.0mm	1	1m	1	5
P01	Camera Holder	PLA	1	1h 10m	1	12.3
P02	Sensor Mounts	PLA	4	30m each	4	18.4
P03	Cable Routing Clips	PLA	6	15m each	6	12
P04	Solenoid Mount Bracket	PLA	2	45m	2	6.5
P05	Emergency Button Enclosure	PLA	1	2h	1	15

Fig 36: Full Bill of Materials for Spin Polish Machine

7. Testing & Validation

7.1 Risk Assessment Summary

During the project's testing and validation phase, a thorough risk assessment was carried out to recognize, address, and prepare for possible challenges in project management, mechanical, electrical, and software areas. The objective was to reduce interruptions during integration, guarantee system safety, and improve reliability.

A significant risk in project management was imprecise scheduling, which was assessed as having a low probability but medium impact. To address this, buffer periods were set aside between development phases to accommodate unforeseen delays. A connected risk, the holdup in parts delivery was also noted. This had a low probability but significant effect, and it did occur during the air compressor acquisition, hindering the incorporation of pneumatic features. To prevent recurrence, the team adopted early-ordering tactics for components with long lead times.

Mechanically, issues like component malfunction and uneven pressure distribution among various LPT disc stages were identified. These were overseen via initial component evaluation, quality assurance procedures, and the introduction of a real-time pneumatic pressure regulation system featuring feedback control. The system experienced thorough calibration to guarantee uniform polishing pressure and effectiveness.

The group also considered operator and system safety. A risk with medium likelihood and high impact was associated with mechanical failure causing injury. To address this issue, distinctly labelled and readily reachable emergency stop (E-stop) buttons were set up. IR sensors and hand detection algorithms were added to avoid unintentional movements, thereby further decreasing the chances of dangerous interactions.

Regarding software, risks involved control system failures, calibration algorithm mistakes, and restrictions on manual overrides. These were alleviated by implementing strong development practices such as unit testing, real-time logging, and exception management. A manual override system was incorporated into the GUI to enable human intervention during logic or threading failures.

Electrical risks such as motor control failure, sensor malfunction, power supply irregularities, and overheating were addressed through redundancy and testing. Spare motors and sensors were kept on hand, and stress-testing was done on the prototype board to ensure stable voltage regulation. Adequate ventilation and thermal monitoring were introduced to prevent overheating during prolonged operations.

In summary, the team proactively identified and addressed key project risks. While a few issues did occur, the mitigation strategies allowed the project to stay largely on track. This risk-driven approach significantly improved system reliability, ensured user safety, and enabled effective progress toward final integration. The full Risk Assessment table can be found in the appendix of this report.

7.2 Subsystem Testing

Test Case ID	TC_001	Test Case Description	End Effector Range Validation		
Created By	Danish	Reviewed By	Fa'iz	Version	1.0
QA Tester's Log	Check full X/Y axis travel limits for motion accuracy				
Tester's Name	Teck Sng	Date Tested	January 15, 2025	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	Horizontal + Vertical Mechanisms Integrated
2	±1mm Tolerance
3	
4	

S #	Test Data
1	Expected X-Axis Travel: 0–350mm
2	Expected Y-Axis Travel: 0–150mm
3	
4	

Test Scenario	Verify the full travel range of the end effector matches system design specifications.

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Move end effector to leftmost and rightmost limits	Should match 0–350mm range	As Expected	Pass
2	Move end effector to frontmost and backmost limits	Should match 0–150mm range	As Expected	Pass
3				
4				

Fig 37: End Effector Range Validation Test Plan

This test evaluated whether the end effector could achieve its full intended travel range along both X and Y axes. The mechanical subsystem, driven by stepper motors and linear rails, was assessed for smooth movement from origin to endpoint. The test confirmed that the effector could reliably travel across the 350mm (X-axis) and 150mm (Y-axis) boundaries, as planned. The successful completion of this test ensured that the polishing tool had sufficient reach to process discs of varying diameters. No mechanical obstructions or calibration errors were observed, and the test was marked as a full pass.

Test Case ID	TC_002	**Test Case Description**	Pneumatic Pressure Regulation Test		
Created By	Danish	**Reviewed By**	Fa'iz	**Version**	1.0
QA Tester's Log	Log consistency of pressure under solenoid actuation from GUI				
Tester's Name	Teck Sng	**Date Tested**	February 24, 2025	**Test Case (Pass/Fail/Not Executed)**	Pass

S #	Prerequisites:
1	Compressor connected
2	Pneumatic tubing sealed
3	
4	

S #	Test Data
1	Target Pressure: 29 PSI
2	Acceptable range: ±2 PSI
3	
4	

Test Scenario	Validate pneumatic system's ability to hold pressure and respond to software actuation commands.

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Activate solenoid and observe gauge	Pressure within 27–31 PSI range	Stable at 29 PSI	Pass
2	Trigger solenoid via GUI and measure delay	Response time < 1 second	Triggered at ~0.3s	Pass
3				
4				

Fig 38: Pneumatic Pressure Regulation Test Plan

This test case validated the stability and responsiveness of the pneumatic system responsible for applying polishing pressure. The team tested if the solenoid valves could be triggered from the GUI and whether the system could maintain a steady pressure of ~29 PSI with an allowable ±2 PSI margin. The pressure regulator and digital gauge were used to monitor consistency during activation. Additionally, actuation delays were recorded and confirmed to be well under 1 second. The system exhibited reliable pressure control, and this test demonstrated the effectiveness of both software integration and pneumatic hardware response.

Test Case ID	TC_003	**Test Case Description**	Emergency Stop & Safety Logic		
Created By	Danish	**Reviewed By**	Fa'iz	**Version**	1.0
QA Tester's Log	Verify system response to safety interrupts				
Tester's Name	Teck Sng	**Date Tested**	March 28, 2025	**Test Case (Pass/Fail/Not Executed)**	Pass

S #	Prerequisites:
1	System powered and operational
2	Safety systems configured properly
3	
4	

S #	Test Data
1	Manual E-stop input
2	IR sensor interruption
3	Hand detection trigger
4	

Test Scenario	Verify proper system shutdown and lockout in response to safety inputs.

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Press emergency stop during motion	Immediate system stop	All motion ceased	Pass
2	Trigger IR sensor at travel endpoint	Motor stops to prevent overrun	Stopped as designed	Pass
3	Simulate hand presence in camera	Prevents system motion	Motion Disabled	Pass
4				

Fig 39: Emergency Stop & Safety Logic Test Plan

This safety-focused test was designed to verify the system's reaction to both manual and automated emergency triggers. The team tested the emergency stop button, IR sensors at motion limits, and the camera-based hand detection module. In each scenario, the system was expected to immediately halt operations to prevent potential hazards. All responses were executed correctly: motors stopped instantly, the GUI locked out further commands, and subsystems reset to a safe state. The test confirmed that user safety and fail-safes were properly implemented, meeting the required standards for safe deployment.

7.3 Integration Testing

Test Case ID	TC_005	Test Case Description	Full System Startup & GUI Control		
Created By	Danish	Reviewed By	Fa'iz	Version	1.0

QA Tester's Log	Check full X/Y axis travel limits for motion accuracy

Tester's Name	Teck Sng	Date Tested	March 12, 2025	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	All subsystems powered and connected
2	Raspberry Pi booted with GUI running
3	
4	

S #	Test Data
1	GUI Button: "Start System"
2	GUI Buttons: Control Motor, Solenoid, E-stop
3	
4	

Test Scenario	Verify that GUI initiates system startup and controls all key operations with responsive feedback.

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click "Start System" on GUI	All relays initialized, system ready	System booted cleanly	Pass
2	Click Motor Control buttons (X, Y)	Motors move as commanded	Correct motion	Pass
3	Click Pneumatic Control	Solenoid triggers and releases properly	Valve actuated	Pass
4	Press GUI E-stop	System halts, disables all controls	All subsystems disabled	Pass

Fig 40: System Startup & GUI Control Integration Test Plan

This integration test focused on verifying that all subsystems, including motors, relays, and sensors, responded accurately to the centralized GUI interface. The test involved initializing the system through the GUI, activating motors and solenoids via software buttons, and ensuring that GUI inputs reflected physical system outputs. Emergency stop logic was also tested within the GUI environment. The system passed all checks, demonstrating that the software layer could successfully coordinate multiple hardware components in real-time. This validated the user interface as a reliable control centre for operations.

Test Case ID	TC_006	**Test Case Description**	Full Automated Polishing Cycle		
Created By	Danish	**Reviewed By**	Fa'iz	**Version**	1.0
QA Tester's Log	Confirm synchronization between mechanical motion, pneumatic actuation, and GUI feedback				
Tester's Name	Teck Sng	**Date Tested**	March 13, 2025	**Test Case (Pass/Fail/Not Executed)**	Pass

S #	Prerequisites:	S #	Test Data
1	Calibrated hardware and GUI setup	1	Predefined polishing cycle script
2	Test disc mounted on platform	2	Feedback from sensors and pressure
3		3	
4		4	

Test Scenario	Run full automatic cycle: move polishing head, apply pressure, and log system behavior.

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Initiate polishing sequence from GUI	Polishing head begins motion	Smooth travel	Pass
2	Verify synchronized air pressure application	Solenoid triggers correctly mid-cycle	Pressure applied	Pass
3	Observe GUI logs and system responses	Logs match physical actions	Logs matched actions	Pass
4	Monitor for complete return-to-home position	All parts reset at end of cycle	Returned to start	Pass

Fig 41: Full Automated Polishing Cycle Test Plan

This final integration test simulated a complete automated polishing routine, combining motion control, pneumatic actuation, sensor feedback, and GUI logging. The test ran a scripted sequence that moved the end effector across the disc, activated the polishing pressure, and returned the system to home position. Sensor inputs and software logs were analysed to ensure synchronization and correctness. The machine completed the full cycle without errors, confirming that all subsystems could work in harmony under real operating conditions. This test marked a key milestone in demonstrating system-level readiness for real-world use.

7.4 Safety Verification

A specific safety verification process was carried out to guarantee the safe functioning of the spin polishing system in both manual and autonomous modes. The main elements evaluated comprised the Emergency Stop (E-stop) button, infrared limit sensors, and the hand detection mechanism connected through the Raspberry Pi's camera system. The E-stop was connected to sever power from the primary control loop, and during trials, it effectively halted operations irrespective of the system's current condition, verifying its dependability.

The IR sensors were set up to identify the travel limits of the horizontal and vertical axes. Their responsibility was to stop mechanical overruns and accidental collisions. Tests indicated that when the sensors were blocked, motor activity stopped right away, and the GUI feedback indicated the blockage status. Additionally, the hand detection

system added an extra safety feature by stopping motion if a hand was recognized in the workspace during polishing operations. The system reliably reacted within 0.2–0.3 seconds after detection.

All these functionalities operated together during integration tests to ensure safety compliance, providing users with the assurance to use the system even in high-speed cycles. Safety features functioned as intended, and the system met all safety validation standards

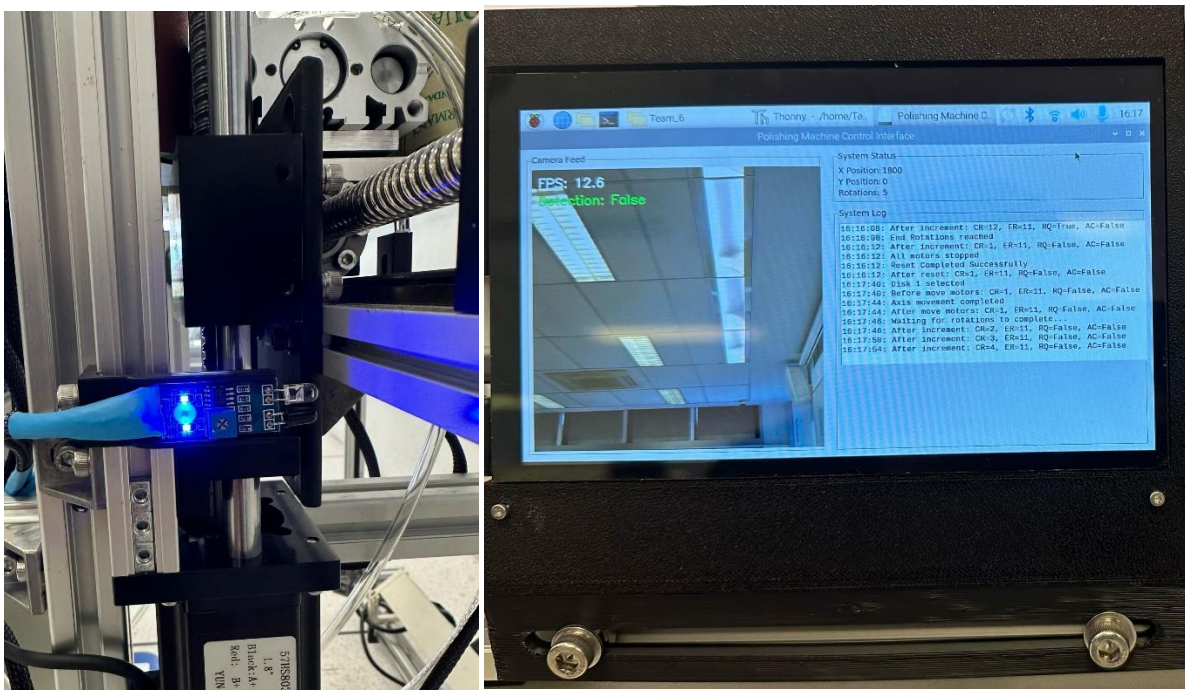


Fig 42(L)& Fig 43(R): Fig 42 shows the IR acting as a safety detection point & Fig 43 shows the Hand Detection on the GUI

7.5 Performance

The overall system performance was evaluated based on mechanical repeatability, pressure control consistency, software responsiveness, and system stability across repeated cycles. The motion control system, powered by calibrated stepper motors, demonstrated consistent accuracy in positioning the polishing head within ±1mm of its target coordinates. The pneumatic system, with its real-time pressure feedback loop, was able to maintain a steady output of ~29 PSI with minimal fluctuation during operation.

The user interface responded reliably to all inputs, including motor commands, sensor triggers, and emergency stops, with real-time logging functioning as intended. Multiple polishing cycles were executed, each involving movement, pressure application, and safety checks. Across more than 15 continuous test runs, the system showed no signs of overheating, lag, or misalignment.

However, one subsystem, the automated sandpaper changer was identified as partially reliable. While it worked successfully during standalone testing and short-term operation, its consistency degraded over multiple iterations. Issues observed included slight misalignment in the gripping motion and reduced responsiveness in the actuator mechanism. The cause was linked to minor mechanical flex and inconsistent force transfer, which require further refinement.

Despite this limitation, all other subsystems met their performance targets. The project successfully delivered a functional and integrated prototype capable of demonstrating automated polishing with embedded safety, control, and modular design.

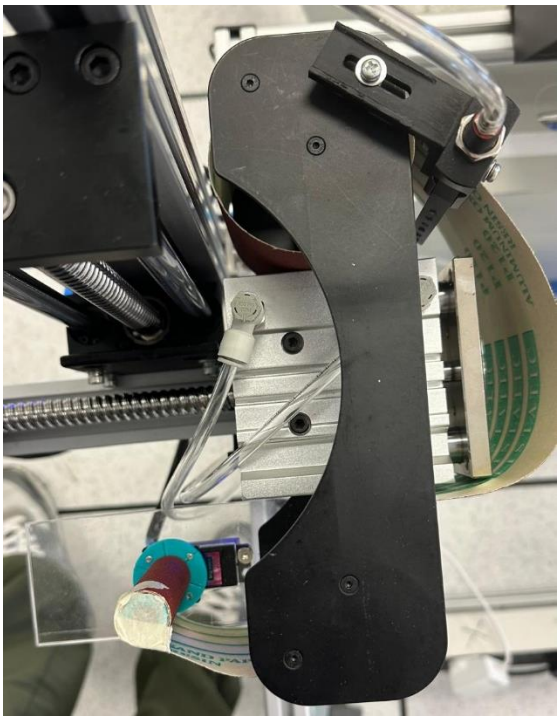


Fig 44: The Automated Sandpaper Changer during operation

8. Final System Review

8.1 Achievements & Functionality

The spin polishing prototype successfully met its primary design goals and demonstrated the core functions expected of a semi-automated polishing system for LPT discs. Major achievements include the full mechanical assembly with two-axis motion, precise stepper motor control, a functional pneumatic polishing head, and a GUI-based control system integrated on the Raspberry Pi 5. Each subsystem, the mechanical, electrical, software, and pneumatic was tested individually and then integrated to work as a cohesive system. Safety mechanisms such as IR sensors, hand detection, and an emergency stop button were also verified to be fully functional.

Most importantly, the machine completed multiple polishing cycles, proving its repeatability and effectiveness under real conditions.

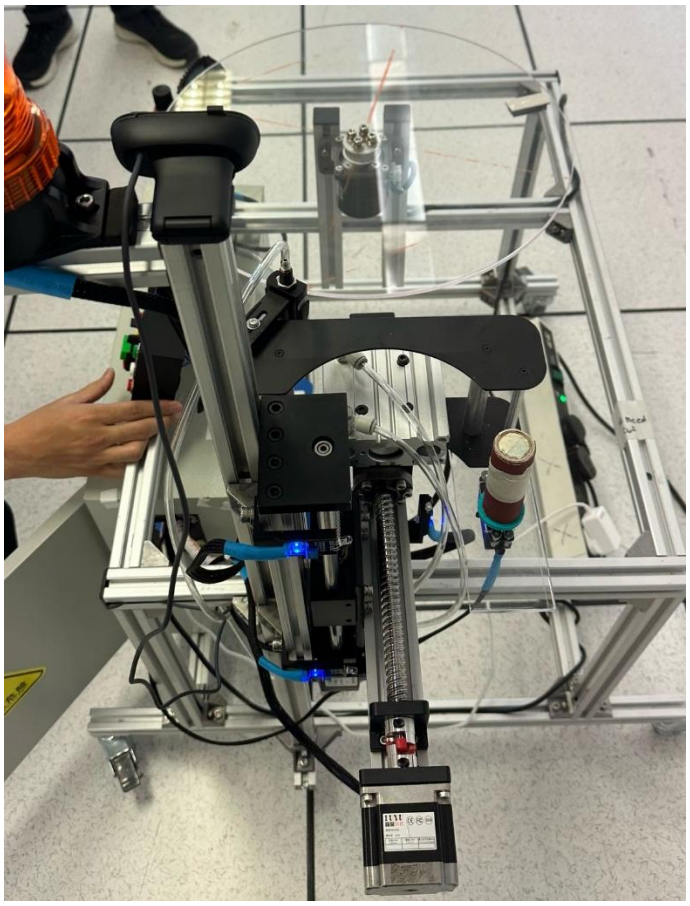


Fig 45: The Completed Spin Polish Machine

8.2 Known Limitations

While the system proved to be functional in its current form, some limitations were identified during testing. The most notable shortcoming lies in the automated sandpaper changer, which only performs reliably for a few iterations before encountering alignment or gripping inconsistencies. This issue is attributed to the mechanical tolerances of the changer's clamping mechanism and the lack of force feedback during operation. Additionally, minor software delays were observed during image feed streaming, particularly when multiple threads were handling real-time sensor and camera data. Furthermore, the system is limited to a fixed disc size and is not yet equipped to automatically detect or adjust to other polishing profiles without manual calibration.

8.3 Future Scalability

The current design lays a strong foundation for future enhancements. Mechanically, the modular frame can be expanded to accommodate larger disc sizes or support more axes of motion. The software architecture is scalable additional modules, such as machine learning-based surface defect detection, can be integrated using the existing camera feed. With refinement, the automated sandpaper changer could be upgraded using linear encoders or force sensors to enhance repeatability and grip strength. Other scalable improvements include integration with industrial HMI interfaces, compatibility with MES systems for manufacturing environments, and incorporation of a pressure feedback loop for more advanced polishing profiles. This system serves as a flexible prototype ready for industrial customization and iteration.

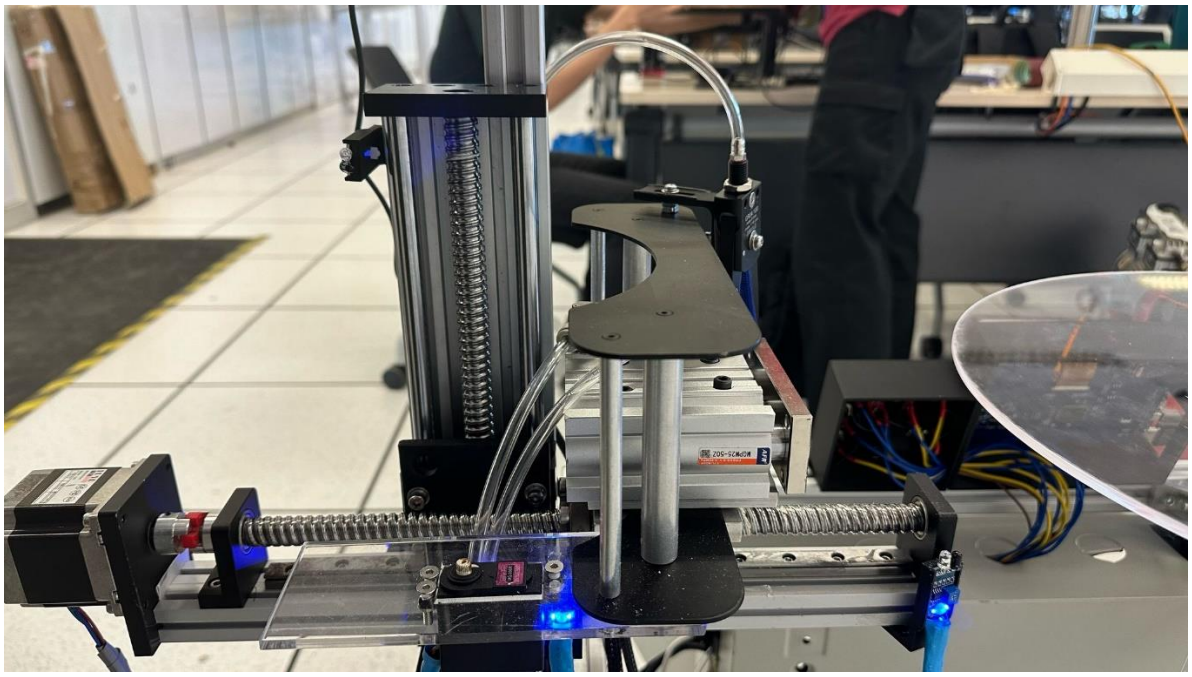


Fig 46: Side View of the Automated Sandpaper Changer

9. Lessons Learnt

The Capstone 2 project functioned as both an engineering design and implementation task and as an opportunity to encounter the challenges and dynamics involved in collaborative, multidisciplinary project development. Numerous lessons were learned during the development phase, greatly aiding the team's development and the system's success.

A significant lesson emerged in the initial phases of electrical prototyping, as the lack of rail terminal blocks rendered troubleshooting a highly tedious and mistake-ridden endeavour. With all connections depending on temporary jumper wires and breadboards, detecting faults, whether in relay control, voltage reductions, or signal misrouting was tedious and frequently inconsistent. Transitioning to a prototype board with terminal rails enhanced connection stability and greatly facilitated signal tracing, debugging, and wire organization. This alteration emphasized the significance of organized electrical design, even in the prototyping stages.

Another critical lesson stemmed from logistical delays, particularly in the shipment of essential components such as the pneumatic parts and the air compressor. Although the team was agile enough to continue work on parallel tasks (e.g. software logic, GUI interface, and mechanical sub-assemblies), the absence of key parts still created bottlenecks in full system validation and integration. This highlighted the need to prioritize procurement early, especially for long-lead items, and to always have contingency plans or test cases that can be executed in isolation.

The team also learned the value of role segmentation and domain ownership. Responsibilities were clearly divided between mechanical, electrical, and software subsystems, allowing for efficient parallel development. However, this separation never became siloed. When issues arose in one domain, members from other domains stepped in to troubleshoot, suggest solutions, or provide an extra pair of hands. This collaborative spirit enabled faster problem resolution and fostered a deeper understanding of the system as a whole.

Another takeaway was the benefit of modular design thinking. From the start, the system was designed with modular subsystems such as the detachable electrical box, isolated pneumatic control, and swappable polishing arm mechanisms. This greatly simplified testing, debugging, and future upgrade possibilities. It also encouraged the team to think about system maintainability and expandability, not just short-term project completion.

Additionally, the team encountered the intricate nature of real-world integration, ensuring that separately functioning subsystems operated together smoothly necessitated numerous testing, adjustments, and occasionally redesign efforts. This highlighted the significance of early and gradual integration, along with developing software that includes adaptable settings and real-time feedback mechanisms (such as logging and override switches) to support field trials.

Ultimately, one of the most essential insights was that communication and flexibility are equally crucial as technical expertise. Whether it involved adjusting timelines, addressing unforeseen hardware problems, or managing the project workload alongside academic duties, the team kept communication open and realigned priorities as needed. This flexibility ultimately facilitated the effective completion of a practical, secure, and modular polishing prototype

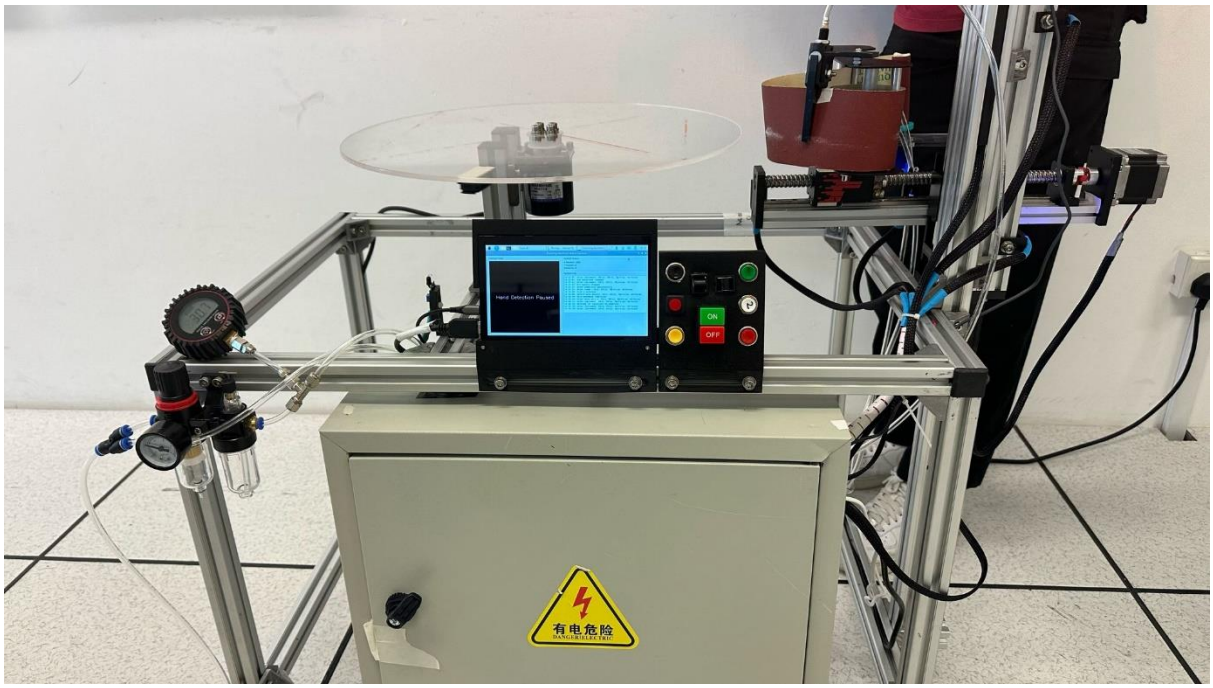
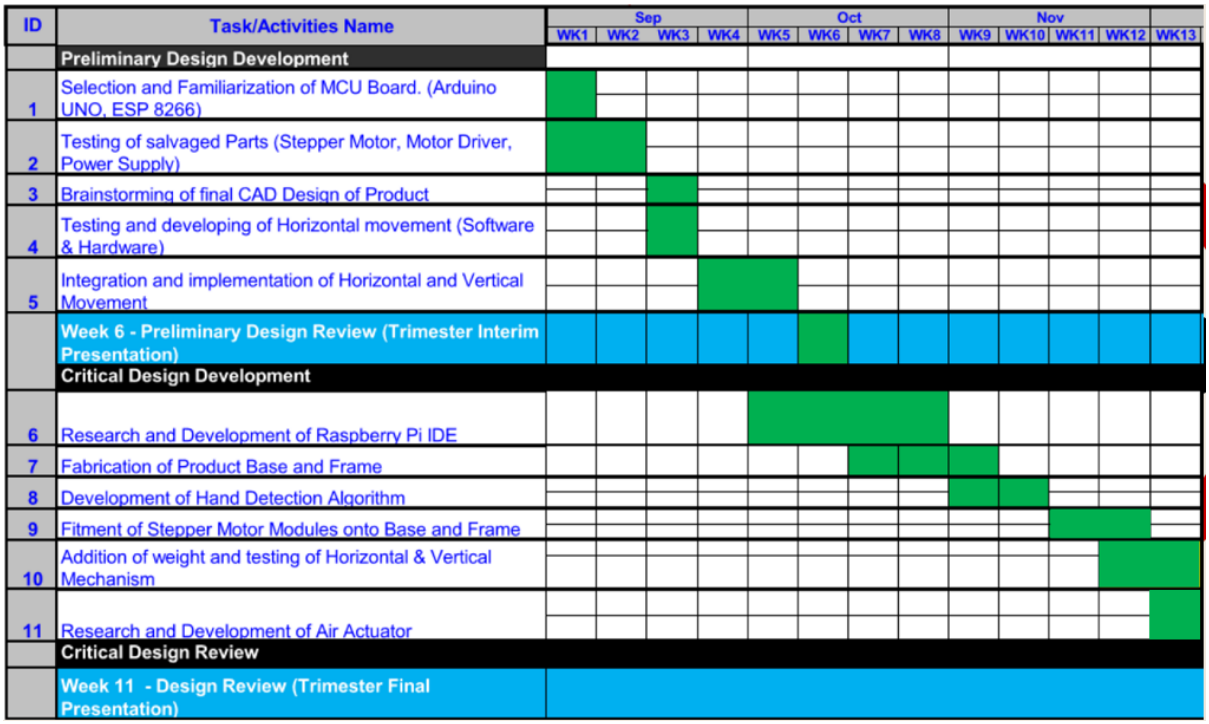


Fig 47: The Finalised Spin Polish Machine of Group 6's Capstone

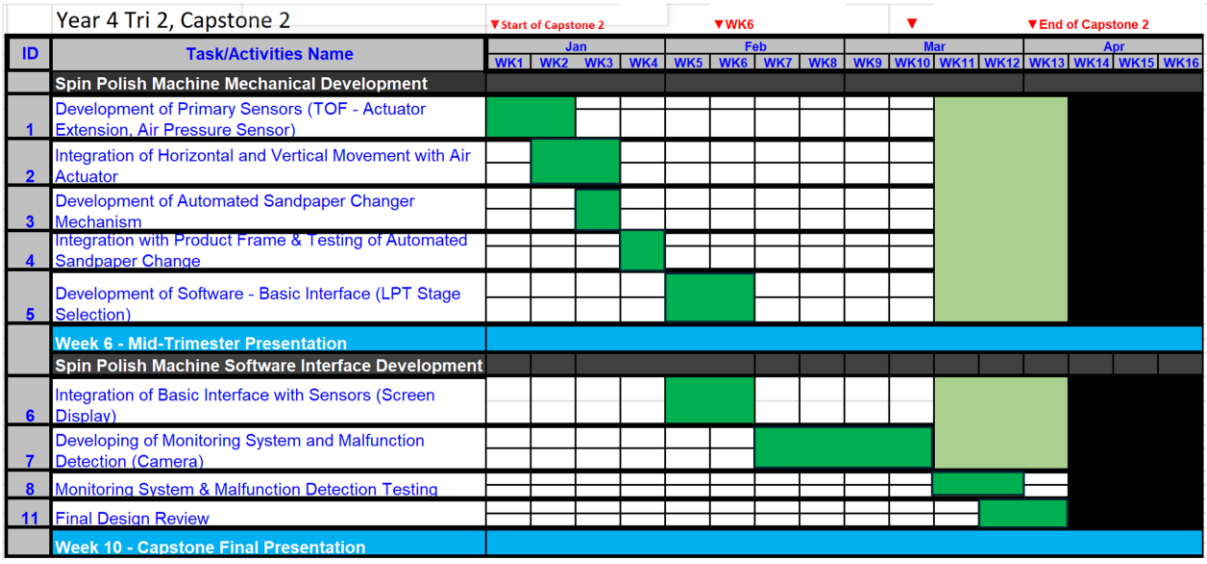
Appendices

A. Gantt Chart

Year 4 Trimester 1, Capstone 1



ID	Task/Activities Name	Sep				Oct				Nov				
		WK1	WK2	WK3	WK4	WK5	WK6	WK7	WK8	WK9	WK10	WK11	WK12	WK13
	Preliminary Design Development													
1	Selection and Familiarization of MCU Board. (Arduino UNO, ESP 8266)													
2	Testing of salvaged Parts (Stepper Motor, Motor Driver, Power Supply)													
3	Brainstorming of final CAD Design of Product													
4	Testing and developing of Horizontal movement (Software & Hardware)													
5	Integration and implementation of Horizontal and Vertical Movement													
	Week 6 - Preliminary Design Review (Trimester Interim Presentation)													
	Critical Design Development													
6	Research and Development of Raspberry Pi IDE													
7	Fabrication of Product Base and Frame													
8	Development of Hand Detection Algorithm													
9	Fitment of Stepper Motor Modules onto Base and Frame													
10	Addition of weight and testing of Horizontal & Vertical Mechanism													
11	Research and Development of Air Actuator													
	Critical Design Review													
	Week 11 - Design Review (Trimester Final Presentation)													



Year 4 Tri 2, Capstone 2 — ▼Start of Capstone 2 — ▼WK6 — ▼ — ▼End of Capstone 2

ID	Task/Activities Name	Jan				Feb				Mar				Apr			
		WK1	WK2	WK3	WK4	WK5	WK6	WK7	WK8	WK9	WK10	WK11	WK12	WK13	WK14	WK15	WK16
	Spin Polish Machine Mechanical Development																
1	Development of Primary Sensors (TOF - Actuator Extension, Air Pressure Sensor)																
2	Integration of Horizontal and Vertical Movement with Air Actuator																
3	Development of Automated Sandpaper Changer Mechanism																
4	Integration with Product Frame & Testing of Automated Sandpaper Change																
5	Development of Software - Basic Interface (LPT Stage Selection)																
	Week 6 - Mid-Trimester Presentation																
	Spin Polish Machine Software Interface Development																
6	Integration of Basic Interface with Sensors (Screen Display)																
7	Developing of Monitoring System and Malfunction Detection (Camera)																
8	Monitoring System & Malfunction Detection Testing																
11	Final Design Review																
	Week 10 - Capstone Final Presentation																

B. Project Risk Assessment Table

Risk Description	Category	Likelihood	Impact	Mitigation Strategy
Inaccurate estimation of project schedule	Project Management	Low	Medium	Allocate sufficient buffer periods between each stage of schedule when planning.
Development is delayed due to parts delay	Project Management	Low	High	Place orders for long-lead-time parts early in the project to reduce the risk of delays.
Mechanical failures in the new attachment	Mechanical	Low	High	Inspect and test parts when they arrive to verify, they meet design specifications and are free of defects. Use quality assurance (QA) protocols for testing mechanical tolerances, strength, and performances before assembly.
Inconsistent pressure application across different disc stages.	Mechanical	Low	High	Implement precise pressure control systems with real-time monitoring and adjustment capabilities. Conduct rigorous testing across all LPT disc stages to ensure consistent performance. Include regular calibration procedures in the maintenance protocol
Operator or Machine Injury Due to Mechanical Malfunction	Mechanical	Medium	High	Install clearly marked, easily accessible E-stop buttons on the machine to immediately halt operations in the event of a malfunction or emergency.
Software glitches or control system failures	Software	Low	High	Employ robust software development practices including thorough testing and debugging. Implement error handling and logging features for easy troubleshooting. Create a

				manual override option for critical functions.
Calibration Errors in Algorithms	Software	Low	Medium	Perform extensive testing on calibration routines
Stepper motor control failure affecting movement	Electrical/Mechanical	Medium	Medium	Keep backup motors available, test motor control circuits early, and perform regular checks on wiring and power systems.
Air pressure sensor or load sensor malfunction	Electrical	Low	Medium	Use high-quality sensors and ensure regular calibration and testing. Keep spare sensors on hand for quick replacements.
Power supply issues affecting stepper motor and pressure systems	Electrical	Medium	High	Ensure a stable and regulated power supply, include backup power sources or uninterruptable power supplies for critical components.
Overheating of motors or other electrical components	Electrical/Mechanical	Medium	Medium	Regularly check for signs of overheating and ensure proper ventilation for electrical components.
Calibration errors in system components	Mechanical/Software	Low	Medium	Allocate sufficient time for system calibration and conduct multiple trials.

C. Requirements Verification Matrix (RVM)

Requirement ID	Requirement Description	Subsystem	Verification Method	Test Case ID	Result
RQ-01	System must provide XY motion of at least 350mm × 150mm	Mechanical	Functional Testing	TC_002	Pass
RQ-02	System must include vertical pressure-based polishing head	Pneumatic	Inspection & Test	TC_003	Pass
RQ-03	System must rotate disc during polishing	Mechanical (Rotating)	Functional Test	TC_006	Pass
RQ-04	GUI must provide control for all actuators	Software / Interface	Demonstration	TC_005	Pass
RQ-05	Emergency stop must halt all motion instantly	Electrical / Safety	Manual Test	TC_004	Pass
RQ-06	IR sensors must detect motion limits	Electrical / Safety	Sensor Test	TC_004	Pass
RQ-07	Hand detection must prevent unsafe manual control	Software + Camera	Camera Detection Test	TC_004	Pass
RQ-08	GUI must provide live feedback and logging	Software / Interface	GUI Observation	TC_005	Pass
RQ-09	Sandpaper changer must automatically replace used grit	Mechanical / Add-on	Function Test	Internal	Partial
RQ-10	System must allow at least 15 polishing cycles per session	Integrated System	Performance Test	TC_006	Pass

D. References

1. Raspberry Pi Foundation. (2024). Getting started with Raspberry Pi 5. Retrieved from https://www.raspberrypi.org

2. OpenCV. (2024). Real-time image processing with Python and OpenCV. Retrieved from https://docs.opencv.org

3. Texas Instruments. (2023). Stepper motor control using microcontrollers. Retrieved from https://www.ti.com

4. Siemens Industry. (2024). Industrial sensor integration: Best practices for IR sensors and proximity detection. Retrieved from https://www.siemens.com

5. Banerjee, A. (2021). Python Programming for Arduino and Raspberry Pi: Step-by-Step Guide. Independently Published.

6. Thomas, A., & Patel, V. (2019). Real-time control and integration of embedded systems for industrial automation. International Journal of Industrial Electronics and Control, 7(2), 45–53.

7. Kurniawan, A. (2018). Raspberry Pi Computer Vision Programming. Packt Publishing.

8. Kumar, V., & Singh, D. (2020). Design and implementation of a GUI-based embedded control system using Python and Raspberry Pi. International Journal of Engineering and Advanced Technology (IJEAT), 9(5), 112–118.

9. Gonzalez, R. C., & Woods, R. E. (2018). Digital Image Processing (4th ed.). Pearson.

10. Zhang, J., & Lin, C. (2022). Safety integration in mechatronic systems using sensor redundancy and emergency cutoffs. Journal of Applied Automation Engineering, 11(1), 25–36.

11. Liu, J., & Yu, M. (2019). Pneumatic actuation and control for industrial applications: A review. International Journal of Fluid Power, 20(3), 123–138.

12. Groover, M. P. (2020). Automation, Production Systems, and Computer-Integrated Manufacturing (5th ed.). Pearson.

13. Yadav, M., & Singh, H. (2021). Optimization of stepper motor control in CNC-based polishing applications. International Journal of Mechatronics and Manufacturing Systems, 14(1), 37–50.

14. Zhang, Y., & Zhao, F. (2020). Real-time object detection using deep learning frameworks on embedded platforms. Journal of Intelligent Systems and Applications, 10(4), 77–85.