

P20 – Filament Spoolers

Design Documentation for DMMS Autumn 2024



Project/ID

Plastic Filament Extruder for 3D Printing (P20)

Clients

Richard Jamie

Marc Carmichael

Team

Jeremy Chu (14178077)

Johnathan Harris (13246160)

Duc Do (14317655)

Mark Luu (13199118)

Affiliation

UTS / ProtoSpace

Table of Contents

1. Project Summary	5
1.1. Problem Statement	5
1.2. Project Objectives.....	5
1.3. Project Deliverables	5
2. Project Budget and Tasks	7
2.1. Project Budget and Plan	7
2.2. Project Tasks and Activities	8
3. Responsibilities and Expectations	9
3.1. Cooling Subsystem	9
3.1.1. Responsibilities:.....	9
3.1.2. Expectations:	9
3.2. Measuring Subsystem	9
3.2.1. Responsibilities:.....	9
3.2.2. Expectations:	9
3.3. Tensioning Subsystem	10
3.3.1. Responsibilities:.....	10
3.3.2. Expectations:	10
3.4. Spooling Subsystem	11
3.4.1. Responsibilities:.....	11
3.4.2. Expectations:	12
3.5. Overall System	12
4. Communication and Key Stakeholders.....	13
4.1. Progress Report	13
4.2. Scheduling and Form of Contact	14
4.3. Accessibility	14
5. Status and Progress	15
5.1. Cooling Subsystem	15
5.1.1. Status:.....	15
5.1.2. Progress:	17
5.2. Measuring Subsystem.....	17
5.2.1. Status:.....	17
5.2.2. Progress:	19
5.3. Tensioning Subsystem	20

5.3.1.	Status:.....	20
5.3.2.	Progress:	22
5.4.	Spooling Subsystem	23
5.4.1.	Status:.....	23
5.4.2.	Progress:	23
5.5.	Overall System	24
6.	Known Issues and Challenges	26
6.1.	Initial Assumptions	26
6.2.	Risk Analysis	26
7.	Recommendations.....	28
7.1.	Cooling Subsystem.....	28
7.2.	Measuring Subsystem.....	28
7.2.1.	Changing for different filament material.....	28
7.2.2.	Laser sensors data change if move filament through after a period	29
7.2.3.	Coding cannot take consistence measurement.....	29
7.3.	Tensioning Subsystem	30
7.4.	Spooling Subsystem – Winding.....	30
7.4.1.	Accommodate different sized spools.....	30
7.4.2.	Better Winding gears/pulley system.....	31
7.5.	Spooling Subsystem – Leveling.....	31
7.5.1.	x-end stopper to home leveller.	31
7.6.	Overall recommendations.....	31
8.	References	33
9.	Appendix	34
9.1.	Microsoft Teams Planning - Jeremy	34
9.2.	Meeting Minutes	34
9.3.	Assembly Instructions	34
9.4.	Technical Drawings – Jeremy and Jonathan	34
9.5.	Wiring Schematics – Mark	35
9.5.1.	Measurer Wire Schematic	35
9.5.2.	Light Sensor Wiring Schematic (Random Nerd Tutorials, 2022).....	36
9.5.3.	Calliper Sensor Measuring Wire Schematic (Potential) (Electronoobs, 2019)	36
9.5.4.	Cooling Wire Schematic	37
9.5.5.	Stepper Motor Control for Winding/Tensioning Wire Schematic	38

9.6.	Coding	39
9.6.1.	Cooling systems.....	39
9.6.2.	Tensioning systems	41
9.6.3.	Measuring systems.....	41
9.6.4.	Level - Winding system	43
9.7.	Measuring System – Pictures of what was attempted.....	45

1. Project Summary

1.1. Problem Statement

UTS staff, in specific, UTS ProtoSpace and FEIT, require the design of a filament spooler compatible with existing extruding technology that can produce 3D printing filament with consistent diameter (1.75 mm \pm 0.3 mm) and quality using recycled 3D prints and experimental materials. A system will be designed, allowing for the implementation of the filament spooler into existing operation, clarifying the feasibility and capabilities of the recycled or experimental 3D printing extruded filament in physicality.

1.2. Project Objectives

Key objectives are presented here as determined based on the functional requirements list and the evaluation criteria.

- **Consistent Quality Filament Produced** – The filament produced is of a diameter 1.75 mm with a tolerance of \pm 0.3 mm; the filament produced has consistent material properties throughout and deformation is minimised.
- **Customisable Settings** – The filament spooler has adjustable settings to accommodate a wide range of materials.
- **Modular Design** – The filament spooler is easy to assemble, upgrade, and has scalability for more general applications.
- **Ease of Maintenance and Resources** – The system of the filament spooler is easy to inspect, allowing for simple troubleshooting and official parts can be sourced domestically for change/replacement.
- **Acceptable Efficiency** – The filament spooler can produce a suitable amount of filament with minimal wastage.

1.3. Project Deliverables

Key Goals

- **Filament Spooler** – A functional model of the filament spooler capable of producing consistent-quality 3D printing filament using recycled 3D prints and experimental materials.
- **Design and Next Steps Documentation** – Basic instructions of operation, detailed technical drawings, schematics, and specifications of the filament spooler design, including assembly instructions and Bill of Materials (BOM).
- **Filament Quality Assessment** – Documentation of filament quality assessment procedures and results, mainly the measurements of diameter but may include material other properties.

Stretch Goals

- **Machine Performance Evaluation** – A report that evaluates the performance of the filament spooler involving efficiency, general reliability, and sustainability (e.g., plastic saved from landfill and electricity usage).
- **Integration Plan** – A proposed plan for the integration of the filament spooler at UTS, this could potentially involve collaboration with faculties such as Design, Architecture, and Building (DAB), FEIT Students, UTS Startups.
- **Operation Manual** – A user-friendly manual providing instructions for operating, maintaining, and troubleshooting the filament spooler.

2. Project Budget and Tasks

2.1. Project Budget and Plan

For this project, the P20 Team worked with a \$600 AUD budget with flexibility regarding increases should it be required. This was presented in a client contract document where some key milestones were also proposed (see Table 1.) and agreed upon prior to starting work. It should be noted however that these were general milestones with the final delivery of the product occurring much later and in segments (agreed upon by the clients).

Table 1. Original Key Milestones Agreed Upon on 28/03/24			
Task – Description	Expected Working Time	Approx. Start Time	Approx. Finishing Date
Project Scoping – Stakeholders interviewed to develop requirements and scope. Research conducted to guide activities and defined scope boundaries. Functional requirements and evaluation criteria will be created to validate the deliverables.	~4 weeks	Start Week 2	End Week 4
Translate and ideate – Stakeholder statements and needs are further refined to develop design concepts. Multiple designs will be presented and reviewed for the suitability of the project.	~3 weeks	Start Week 4	End Week 4
Refined Ideation – Design concepts, specifications, and ideas converge to a solution based on stakeholder(s) feedback and assessment. Rudimentary CAD and sketches will be developed to form a baseline for the chosen solution.	~1 week	Start Week 4	End Week 5
Prototype and Testing – Building a rudimentary model and testing its viability will be conducted to enable for refinement according to the evaluation criteria and functional requirements.	~6 weeks	Start Week 6	End Week 9
Deliverables (Documentation) – Design documentation, quality assessments, integration plans, performance evaluation, and an operation manual will be made.	~7 weeks	Start Week 6	End Week 12
Final Product (Gold Copy) – A product in an acceptable working state for the client is delivered.	~11 weeks	Start Week 2	End Week 12

Note that the week start, and ends are according to the UTS timetable as of 2024 – Autumn.

2.2. Project Tasks and Activities

There are many tasks and activities which were shared amongst P20 group members, these are nested amongst the design process (see Figure 1.). As such, only specific descriptions of tasks and activities will be briefly discussed.

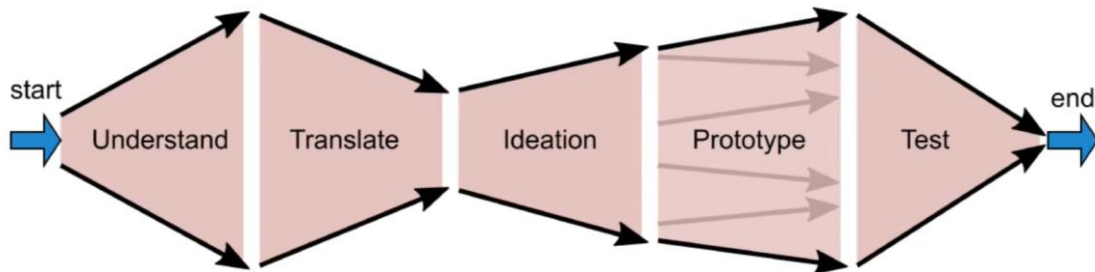


Figure 1. Overview of the Design Process (Carmichael, 2017).

The main project tasks and activities undertaken in this project were:

- **Scheduling tasks** – Tasks to be scheduled amongst team members to prevent overlap, set-up accountability, and increase efficiency.
- **Analysis and measurement of starting parameters**
 - Measurement of the Desktop SJ35 parameters (general dimensions, extrusion parameters, etc.).
 - Analysis of previous groups work (in this case there was none).
 - Properties of plastic pellets.
 - Review of existing technology (literature review).
- **Ordering of materials** – Important to be done ASAP as delays are common (in this case there was a 4–5-week delay).
- **CAD modelling**
 - Rapid prototyping of components in computer space.
 - Construction of models for assembly.
 - Sketching of drawings for documentation.
- **Building and testing**
 - Translation of computer sketches to tangible components in a physical system.
 - Testing of performance of the constructed system using experiments.
 - Further iteration of constructed system.
- **Documentation** – Documentation of processes experienced, achievements, how to replicate experiments/results is *crucial* for the further development of an idea.

3. Responsibilities and Expectations

3.1. Cooling Subsystem

3.1.1. Responsibilities:

The cooling subsystem is an essential role of the extrusion process that manages the immediate heat dissipation of the filament and thereby, shrinking to a desired diameter. The responsibilities include having uniform rapid cooling, modular design such that the layout is interchangeable to accommodate and maintain compatibility of the other subsystems, and a design architecture that can self-adjust accordingly to variable conditions throughout the duration of the extrusion operation. Implementing controlled cooling minimises the effect of filament deformation and sustains the ideal material properties of the filament suitable for 3D printing.

3.1.2. Expectations:

The subsystem is expected to be simplistic, but modular like a closed feedback loop where real-time information such as the temperature and diameter size of the filament is known and self-calibrates to ensure the filament maintains within the 1.75 +/- 0.3 mm range. Cooling of filament to form a durable surface is crucial for it to be processed further in measuring and tensioning stages.

3.2. Measuring Subsystem

3.2.1. Responsibilities:

The measuring subsystem is a control design that allows the extruded filament to maintain the expected diameter. The laser system is controlled by an Arduino. The major function is to determine the drop in lux through the filament and thereby, use this conversion to determine the instantaneous diameter upon contact. The system is expected to be autonomous and does not require further input after starting operation.

3.2.2. Expectations:

The systems are expected to measure and convert the lux to the millimetres which is the diameters of the filament. This data is then used to ensure the filament is at the required 1.75mm diameter. If the filament is too small, the system will send a signal to the tensioning system to slow down so that the filament diameter increases. Similarly, If the filament is too big, the system will send a signal to the tensioning systems to increase the speed of pulling to decrease the diameters of filament.

3.3. *Tensioning Subsystem*

3.3.1. Responsibilities:

The tensioning system is the element which enables the precise control of filament diameter by effect of shear thinning. The assembly consists of Arduino-controlled rollers which grip and “pull” on freshly extruded and cooled filament at specific speeds to achieve diameter consistency. The relationship explained in brief is as follows:

- Faster roller speed = Thinner filament diameter, risk of snapping or stretching the filament too thin.
- Slower roller speed = Larger filament diameter, risk of clogging machine should the filament bunch up.

As such, it is critical that the extrusion rate of the extruder, the tensioning speed is finely tuned with the specific properties of a material to acquire a “goldilocks” zone of filament diameter (1.75 mm \pm 0.3 mm). Ideally, the speed of this system is informed by the results from the measuring subassembly.

3.3.2. Expectations:

The system is expected to be easily controlled using Arduino code using a specified speed according to the material extruded and the extrusion machine parameters. The system is also expected to not deform by squishing filament through its rollers, like soft pliable material such as TPU should be used to prevent unintentional deformation. Alternatively, an alternative spring type may be used according to material desired to be extruded and spooled.

3.4. Spooling Subsystem

3.4.1. Responsibilities:

Level Winding

The Level winding system is a location-based system that allows the extruded filament to be evenly distributed on an empty spool. This system is designed in sync to the Spooling. The Filament adjustor as shown in figure 2 helps maintain even winding by ensuring that each rotation will allow the filament adjustor to move in the x-direction incrementally to ensure the spool is neatly wound on during the process.

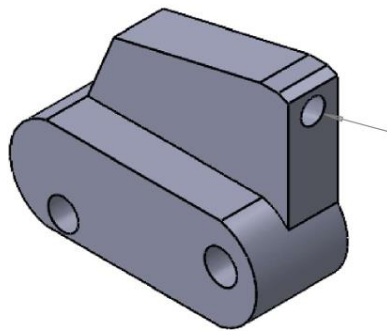


Figure 2. Filament Adjustor

Spooling

The Spooling system is a speed-based system that's function is to wind the filament tightly around the spooler. The speed of this system is controlled in reference to the speed of the tensioning system. The system is a constant speed and is faster than the speed of the tensioner to ensure no build up filament between the two systems.

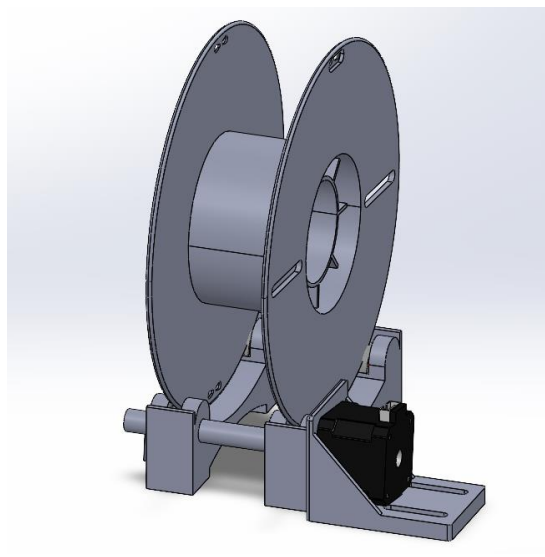


Figure 3. Spooling Sub-Assembly

3.4.2. Expectations:

Due to synchronisation problems. The leveller was not calibrated to provide even spooling for the entire duration of the systems operation. This occurred due to integration problems of the spool's rotational mathematics with the code of the spool. We also used two different Arduinos to control the tensioning – spooling system even though these two systems are intrinsically linked. This meant that one Arduino would have been perfect to control the system. The problem was that there were only two power pins in the Atmega 2560 for powering the motors (3.3V & 5 V). If we had more time to work on the system, we would have made an external power system using a breadboard to allow the joint system to be controlled from the one Arduino.

The filament extruder is an independent system that can ultimately change the systems performance. To ensure the system was producing desirable results, we set the filament extrusion rate at a constant that was considered optimal for the PLA material. These results were extrusion rate 400 (derived 5.5 RPM). Designing the tensioning –spooling system code to this setting was the only way to ensure quality filament and winding of the filament.

3.5. Overall System

Our filament extruder system for consists of four main subsystems (see Figure 4.). These systems are the cooling section, measurement section, tensioning section and the spooling section. Each group member has been assigned a subsystem as their main responsibility to design and prototype for proof of concept and be suitable for the overall assembly and operation of the extrusion process.

The Cooling system uses the air-cooling approach to lower the temperature of the filament at a constant rate to ensure the filament is rigid before reaching the tensioning system. This system works independently from the other system and has its own controller/autonomous functionality to change the speed of cooling.

The measurement system employs a laser and a light intensity sensor to detect light passing through the filament, with control implemented via an Arduino board and the Arduino IDE app. The system relies on rigorous testing to validate conversion of lux reading to diameter and relay real-time information to the other systems to adjust.

The tensioning system uses a belt and pulley-based design to drive rollers. The rollers drive the filament through to the next sub-system that is the spooling system. Pressure is placed on the filament to create friction to pull the filament. The system works based on the extrusion rate of the filament extruder and the measurements taken from the measuring system to calibrate the speed of the pulling action.

The spooling system uses a screw-based guide to evenly wind the filament on the spool. The system is linked to the tension system to match the speed of the spooler, however,

may go faster should the spooling system spool faster than what is being extruded/tensioned, the spooling system will slip such that the filament does not break.

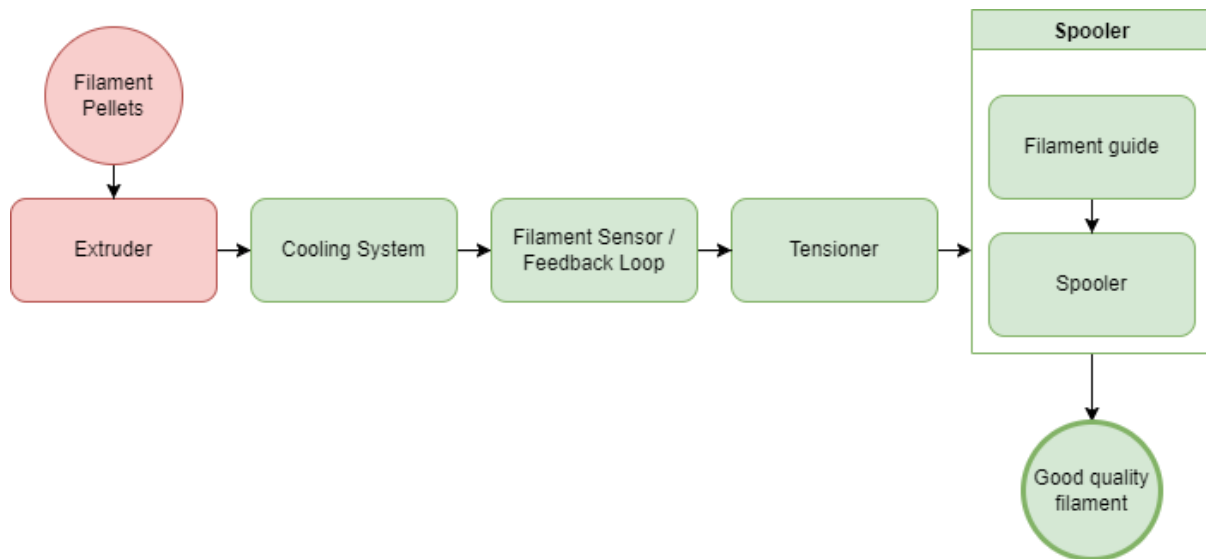


Figure 4. Components of the system to be designed. Red indicates features out of our main control, green highlights features required to be researched and developed by P20.

4. Communication and Key Stakeholders

4.1. Progress Report

The key stakeholders involved in this project were Mark Carmichael, Richard Jamie. Richard Jamie oversaw the filament extrusion project on site, and we went to him for advice and assistance in developing the project wherever he saw fit.

Richard suggested the idea of magnetic mounts to help remove the fans when feeding the filament through the system, this helped the safety issue of burning the user by contact with the extruder.

We had fortnightly meetings with Mark to provide updates on the progress of the assessment. During these meetings he would provide support on how to mitigate any problems that the project was presenting. He was able to help fix the spooling system by suggesting a spooling system that would slip when there was filament tension between the tensioning system and the spooling system. He also recommended a closed-feedback loop by using sensors to communicate if the filament is not the right diameter and automatically make the necessary adjustments.

4.2. *Scheduling and Form of Contact*

Every time we wanted to use the Protospacer Lab, Jeremy would email Richard for approval of using the lab for intended day and period.

4.3. *Accessibility*

Here is a brief list of important areas that we needed access to:

- Access to UTS Protospace
- Access to UTS Protolab
- Access to UTS Protospace 3D Printing
- FEIT Induction Access
- Access to Mechatronics Lab (Optional)

5. Status and Progress

5.1. Cooling Subsystem



5.1.1. Status:

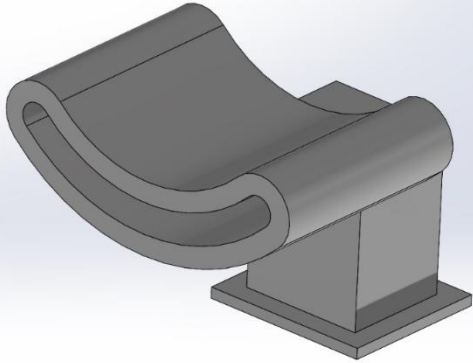
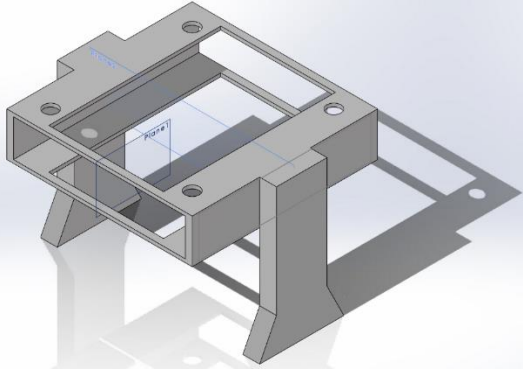
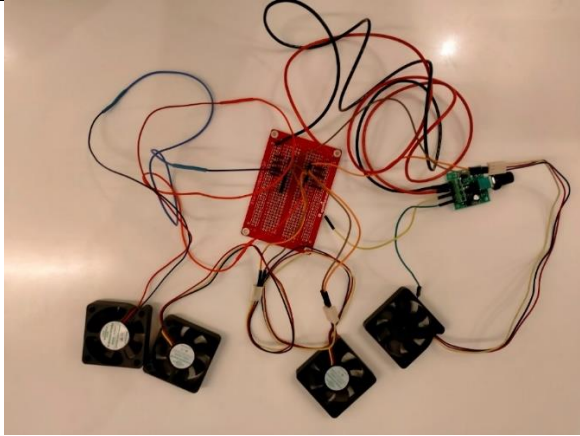
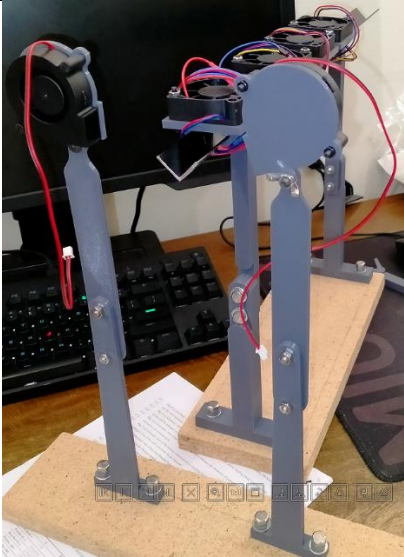
The previous handover did not have an active cooling subsystem. It did, however, have components available for air cooling. Therefore, for this semester, rapid physical prototyping and testing was crucial to identify all internal and external issues concerning cooling the filament. The existing subsystem composes of the following:

- 3D-printed mounts for the system for height adjustment with the extruder nozzle
- Two DC 12V blower fans positioned at the extruding end for rapid cooling
- Aluminium V-shaped trough to guide the filament and provide additional heat dissipation
- Four BLDC 12V axial fans to cool uniformly along the trough
- All fans are controlled by manual PWM controllers for speed control

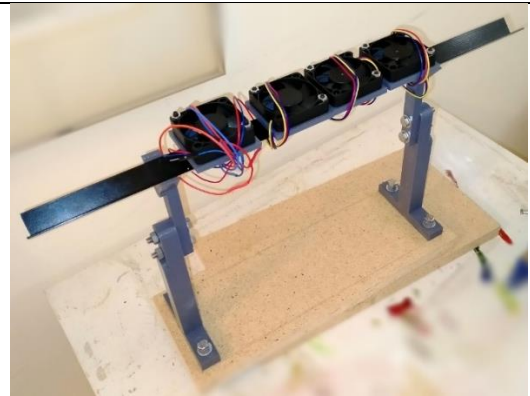
Now, simple modularity by having manual PWM controller toggles for both axial and blower fans are used to regulate voltage hence controlling the speed output of the fans. 3D printed mounting parts was necessary for ease of rapid prototyping and redesign should future groups intend to find more optimal design choices that meet the original project deliverables. The existing fans chosen were based on a balance between RPM speed, CFM (cooling coverage) and operational life that would justify cost-effectiveness as well as overall performance over extended periods of operation. Initially, a crescent-shaped trough was chosen to satisfy any size of filament produced and prevent the filament from reshaping on its own. However, the V-shaped was used instead for two reasons, proper design for 3D printed height support and for the fans ease of mounting.

Table 2. Cooling System Developments

	
<p>Previous iteration: Crescent aluminium trough choice for unhindered movement of the filament to the next system.</p>	<p>Current Iteration: V-shaped aluminium trough for easier integration of 3D printed mounts and supports in rapid prototyping</p>

	
<p>Previous trough support iteration: Awkward shape and tolerance with crescent trough (Also had slight warping despite listed product dimensions)</p>	<p>Previous fan mount iteration: Screw mounts and support of fans needed to be designed on top of the trough, also was not fully 3D print compatible.</p>
	
<p>Current system (Axial): Axial Fans all connected to a breadboard and controlled by a single manual PWM controller for speed control</p>	<p>Current system (Blower): Blower/Centrifugal Fans connected to a breadboard (same one as axial) using a separate manual PWM controller for instantaneous cooling at the filament upon immediate extrusion. Each fan has its own support with adjusters to angle the fans as required for optimal cooling.</p>

Current cooling layout: Mounted fans on a V-shaped aluminium trough held by screwed-in V-shaped supports.



5.1.2. Progress:

Further testing with the existing subsystem is essential to determine if further improvements should be implemented. Referring to the “Machine Testing Experiments and Results” document, three experiments have been conducted to rigorously test the consistency and performance up to 6 hours of continuous operation. The cooling system is effective functionally. However, it still requires more data for verification on cooling efficiency and knowing exactly which areas of improvements can dramatically increase results.

5.2. Measuring Subsystem

5.2.1. Status:

When reviewing the handover documents of the previous group, they did not have a clear measuring system as they were using the principle of measuring the displacement of the magnet and measuring the strength of it. (Figure 5).



Figure 5. Measuring filament diameter base on the strength of the magnet and displacement (Hendra P Syahputra, 2022)

There was little explanation written about how it works or if there was any existing model left behind, so it was assumed that the product failed to produce accurate reading. Therefore, rapid researching and testing was urgent to determine the best device to measure the filament diameters. Based on the research from the file “Analyse Filament Measure”, we can see there are four main concepts that can be tested which are the ultrasonic sensor, light sensor, digital caliper and the time-of-flight sensor.

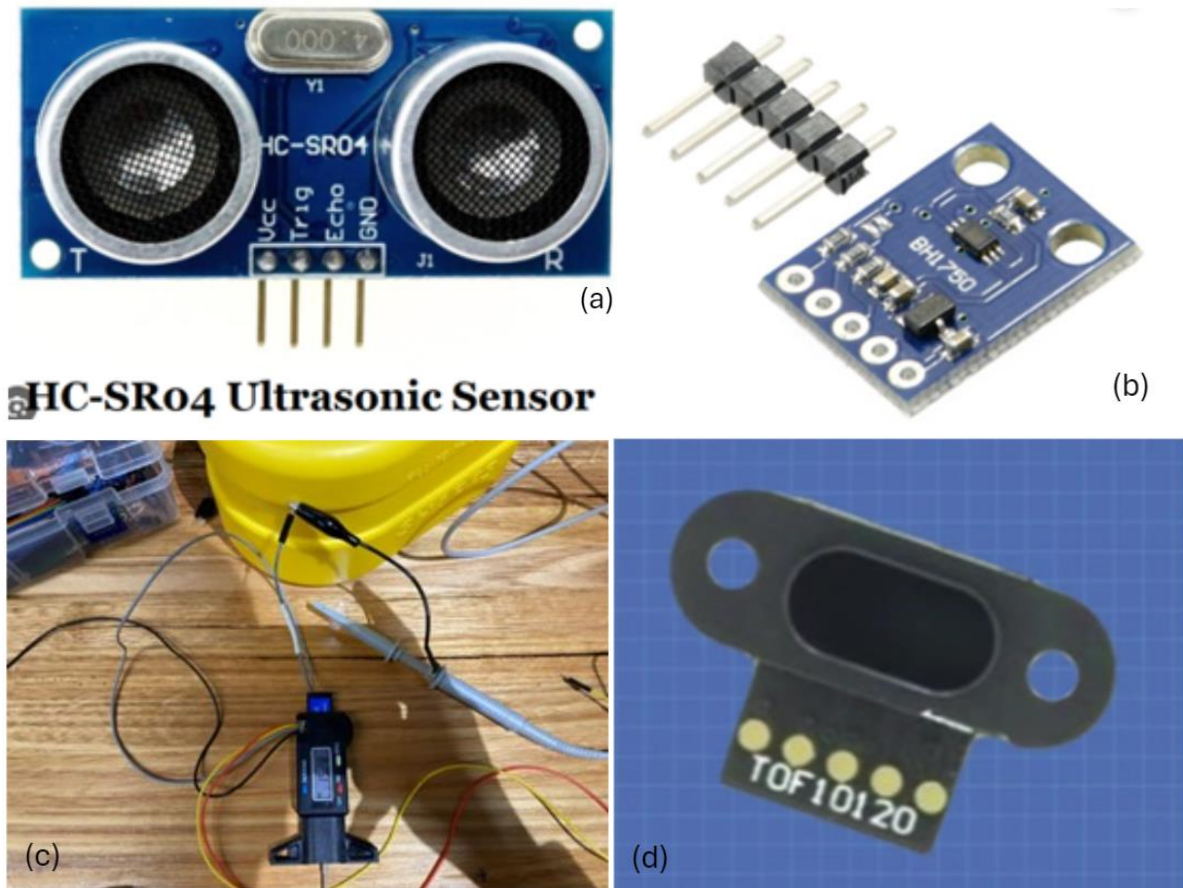


Figure 6. (a) Ultrasonic sensor; (b) Light sensor; (c) Digital calliper; (d) Time-of-flight sensor

Based on the four potential options, we decided to go with the light sensor and the digital caliper. The reason is based on the literature review from lee et al., 2018, the extruder projects that have produced promising results and for the caliper, it was based on a YouTube tutorial from Electronoobs, 2019 which showed practical implementation of the redesigned caliper producing accurate readings.

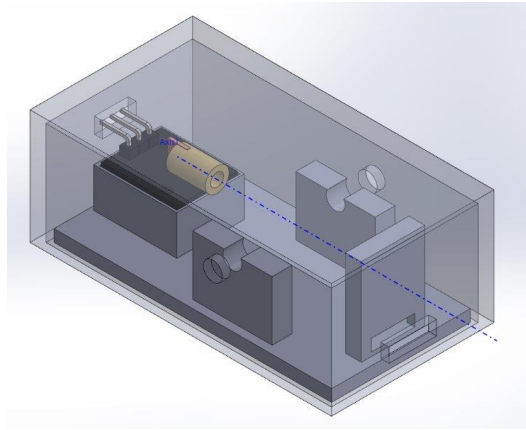


Figure 7. Laser and Light sensor model

Based on the information above, the current system design is a combination of the laser sensor and the calliper to measure the filament diameters but due to incorrect equipment purchase and error of bit count of the calliper, this was not fully realised. For the laser sensor, the first design was getting mixed and nonsensical data which we discussed is due to the filament not being stable on the holder for the laser to detect plus the sensor picking up reflected light from inside of the chamber. (note: I have also tested the ultrasonic sensor, but it can only go down to 1cm which is not enough resolution for the job).

5.2.2. Progress:

Due the continuous error of the bit count and the pin output being incorrect (figure 8), the best decision moving forward was to remove the calliper from the design and follow through with using only the laser sensor. Although there were still issues for example, with the data being mixed with the create model in figure 7 due to the filament not being stable for the sensor to detect it correctly, troubleshooting was easier to handle.

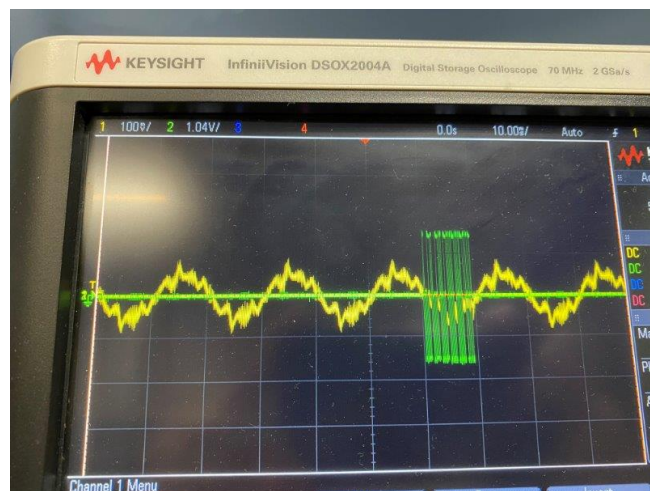


Figure 8. Fail Result from Calliper from incorrect bit count.

For the new design for the laser (figure 33 in the appendix), a silicon base hole was included to allow the filament to run through without any issues. Additional redesigns were creating a smaller hole size reduced the error of filament vertical free motion as it passed through and having the position of the filament closer to the laser for better direct contact on the filament, marginally improving data readings. The other change to the design was adding a filter for the laser when reaching the sensor (a piece of paper). This is to ensure the data receiving is correct.

5.3. *Tensioning Subsystem*

5.3.1. *Status:*

The previous group did not have any tensioning mechanism or parts needed for one; this was concerning as by review of literature finds that a tensioning subassembly is a critical component which maintains a desired filament diameter. As such, the rapid prototyping of this subsystem was performed as early as Week 4. These can be briefly viewed in Figures 9 – 10. The final tensioner device constructed is illustrated in Figure 11. To see the design files, please refer to the Appendix.

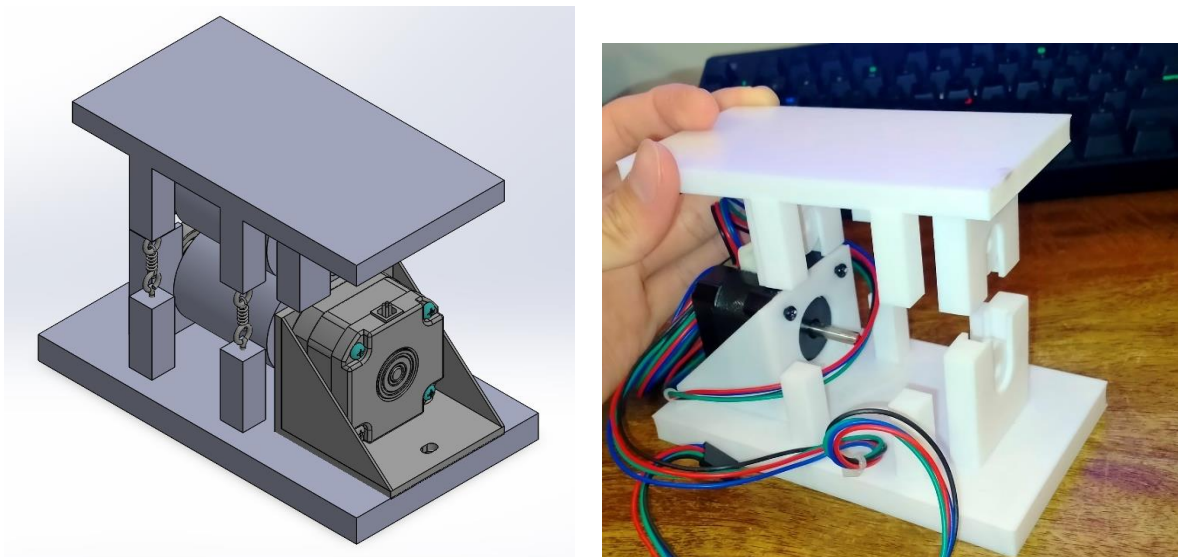


Figure 9. Shows the first iteration of the tensioning device, left being the CAD model, right being the partially realised model. The 4x tension spring design was determined unfeasible due to complex geometry, poor manufacturability, and reliability concerns.

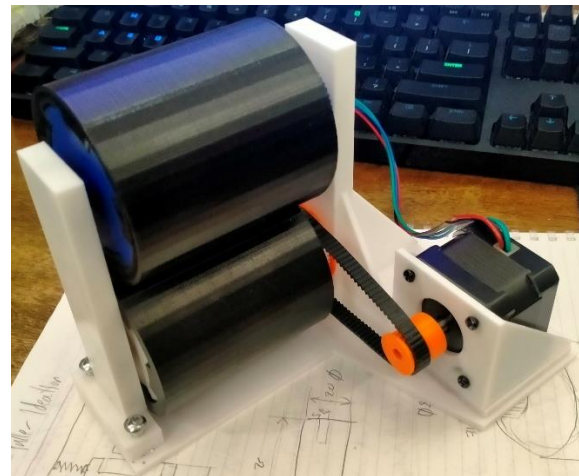
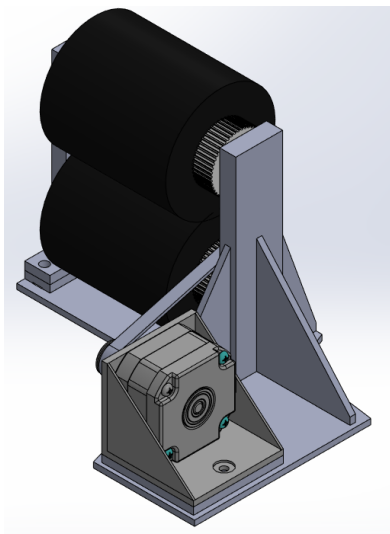


Figure 10. Shows the second iteration of the tensioning device, left being the CAD model, right being the realised model. The design used TPU rollers, timing belt, and pulley. This design functioned well, however, still had room for improvements.

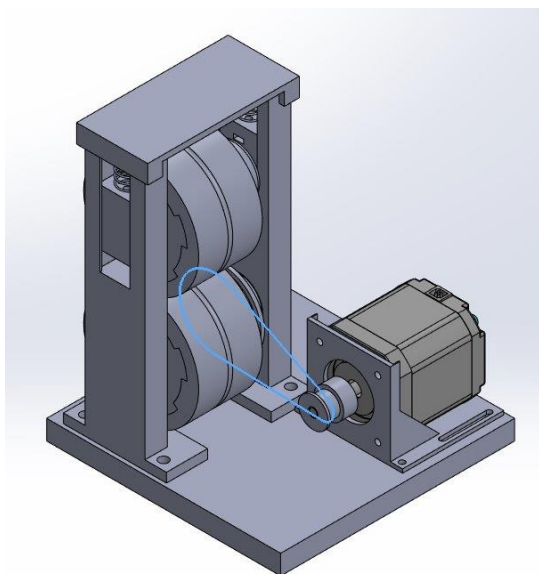


Figure 11. Shows the third iteration of the tensioning device, left being the CAD model, right being the realised model. This was the final iteration that had minimised the main functioning issues, though further improvements can still be made.

5.3.2. Progress:

The third iteration of the design performs well and with a high degree of control. Please refer to the “Machine Testing Experiments and Results” document for further details.

To summarise current progress, an excel document (see Appendix) was made which correlates extrusion machines (Desktop SJ35) gauge reading to its auger RPM (see Figure 12.). A speed of 410, correlating to approximately 5.5 RPM was chosen for most experiments.

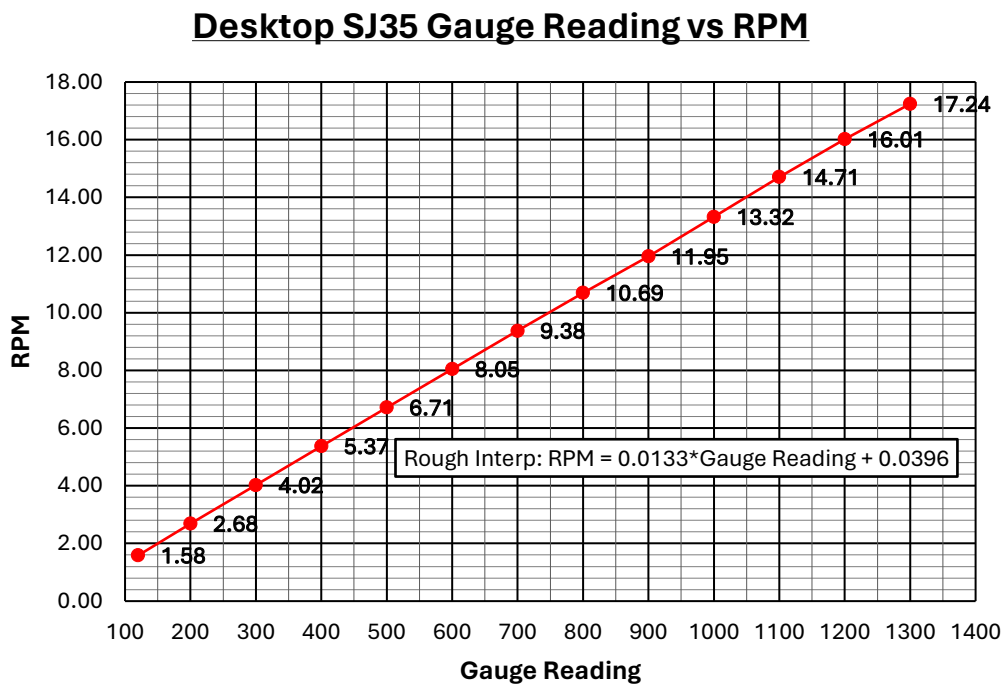


Figure 12. Shows the gauge reading of the Desktop SJ35 extruder’s auger to its actual RPM.

The material experimented within this session was labelled as 1588ATL, though, beyond this no further information could be found. To test the ability of the tensioner to alter filament diameter, different speeding settings (refer to the Excel document in the appendix) was performed, results of which can be viewed in Figure 13.

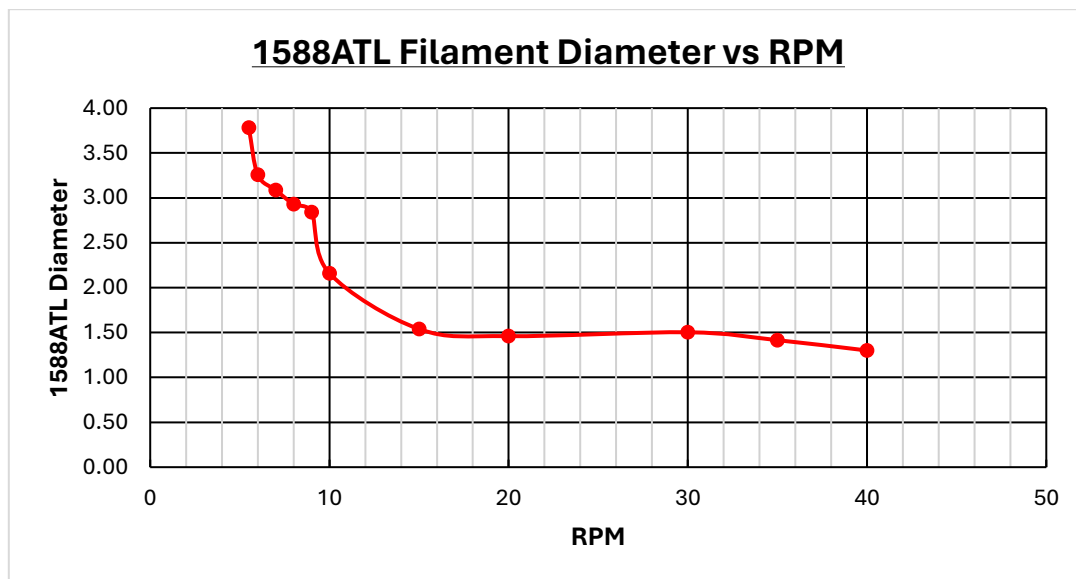


Figure 13. Shows experimental results correlating RPM of tensioner roller speed to 1588ATL diameter.

5.4. Spooling Subsystem

5.4.1. Status:

Level Winding

The level winding system was not considered to be updated until the very end of the project. Marc presented the idea of an x-end stopper for calibration purposes. The need for improving this system did not present itself due sense of urgency to get the level winder to accurately adjust for the whole duration of the test.

5.4.2. Progress:

Spooling

The original system comprised of a spooler fixed on an aluminium shaft. This shaft controlled the rotation of the empty filament spool. It was determined that if the speed is not matching the tensioning speed due to human input, it can create the risk of ripping the filament apart from the system. This discussed the possibility of adding a sensor to communicate the speed of the filament, however, was disregarded due to over-engineering and still due to present risk of breaking the filament if the speeds are not exactly matched. Therefore, a new design was proposed that allows the spooler to rotate at a fixed speed without the need to accommodate to the increase in spool circumference or if the tensioning system changes speed. The idea was to control the speed of the spool by putting in on top of the rotating pulleys. If the speed of the filament is too slow in comparison to the spooling rotation, then tension will occur causing the spooler to slip on top of the pulley and the given bearings. At the time, there was no other way around. If the rotation is too slow, too much filament will build up between the

spooling system and the tensioning system leading to some fault caused by entanglement. We thus settled for an angular velocity that will always be faster than the tensioning system. This will cause slipping to occur, however no entanglement problems.

5.5. Overall System

Based on the status of the project, producing filament of consistent diameter was very viable. However, there were issues with two systems, the Measurement system and the Spooling system. These systems could not be completed due to time constraints and equipment failure. Improvement of each system could include:

Cooling system

There is sufficient cooling for the filament to the point that it maintained its shape within 0.2-0.5mm of its desired diameter when moving through the tensioner which is a great success. However, air cooling was not fully effective in cooling down the filament completely as at higher temperatures (>150) passing through the systems, heat can still be detected from within the filament which may reduce the quality of the filament. Therefore, recommendations are to either focus purely on improving the existing system or determine if water cooling could increase the cooling efficiency. At the time of the status, water cooling was not an option due to two reasons: One, ProtoSpace does not permit water technology in the ProtoLab and two, most options were very expensive, almost to the industry level.

Measurement system

Detecting the filament lux was found possible but failed to produce the correct filament diameter. This data incompatibility resulted from the detection which causes the conversion equation from lux to millimetres to be inaccurate and produce undesirable data. The measurement system can control the speed of the tensioning system; however, this feedback loop is not executed until one rotation of the tensioning system has completed. By the time one revolution is completed, a few centimetres of filament have already been extruded. Thus, proving the time is too long. As a result, it is recommended a calliper is used to measure the filament correctly and using threading in the coding block to complete multiple tasks at the same time.

Tensioning system

It was successful to draw and feed filament to the next system consistently, however, the system needed the filament to be manually placed through and occasionally, the filament would skip and miss the tensioner. So, it is recommended creating a guiding system to make the process more reliable.

Spooling System

Spooling the filament was successful and the system would slip whenever the filament tension was too great. This was executed well as the slipping could occur without breaking the filament. However, due to time constraints and 3D printing parts failure, fully testing the system to determine its effectiveness in winding and correctly guiding the filament through was not conducted. In addition, the spool sometimes would skip and stall until manual intervention. This may be due to the weight issue or the bearing connecting with the spool. Therefore, recommendations are changing the connection between the spool and the motor turning to increase friction.

6. Known Issues and Challenges

6.1. Initial Assumptions

For the assumption of this project, there were three aspects to consider:

- Environmental conditions
- Operation parameters
- Raw material quality

When looking at environmental conditions, the machine will only be used indoors with maintained standard temperatures (20-25 degrees) and humidity (30-50%). For the operational parameters, the machine will work a maximum of 8 hours a day since ProtoSpace does not permit the machine to work unattended or beyond hours. It is also assumed that the extrusion rate is constant. Lastly, for the raw material quality, it is assumed that the pellets used is correctly processed and free of contaminants. This is to ensure every product produced is uniform and the data collected is correct.

6.2. Risk Analysis

Based on the current setup of the project, a contingency plan is necessary to address potential issues. A table has been made to show what problems can occur, what level of risk they have and the countermeasures in place to ensure that issues can be dealt with.

Table 3. Risk Analysis				
Problems	Rate Problem (1-25)	Likelihood (1-5)	Consequence (1-5)	Countermeasure
Equipment Failure	6	3	2	maintenance, high quality parts
Raw Material Contamination	8	2	4	Checking material before usage, buy material from reliable source
Health and safety hazards	15	3	5	Ensure all safety protocol are followed, wear PPE, ensure proper ventilation for certain materials

		Consequence				
		Negligible 1	Minor 2	Moderate 3	Major 4	Catastrophic 5
Likelihood	5 Almost certain	Moderate 5	High 10	Extreme 15	Extreme 20	Extreme 25
	4 Likely	Moderate 4	High 8	High 12	Extreme 16	Extreme 20
	3 Possible	Low 3	Moderate 6	High 9	High 12	Extreme 15
	2 Unlikely	Low 2	Moderate 4	Moderate 6	High 8	High 10
	1 Rare	Low 1	Low 2	Low 3	Moderate 4	Moderate 5

Figure 14. Risk Scale (Guthrie, 2020)

7. Recommendations

Parts orders occurred between Sprints 2 to 3; While personal parts were incorporated for substitution during testing, it is highly recommended to source and confirm purchases to prevent delay in project progress. Despite ordering “relatively” early on 02/05/24, our first parts only started arriving on 03/05/24 despite ordering from Australian distributors.

External stakeholders encountered was the Design, Architecture and Building (DAB) Department where their shredder may be used for the shredding of 3D prints for recycling. A problem encountered with their machine is their mesh filtering component which would separate finer usable shredded particles from larger unusable ones. Currently, their mesh size is larger than what would be required for this project’s application. Please see the attached document “DAB – Tony Meeting – 08.03.24” for more information.

7.1. *Cooling Subsystem*

The next steps for the cooling subsystem are designing advanced closed-loop feedback architecture. This can be done in two suggested paths, however, innovative approach outside of these paths is also open to implementation if the overall methodology maintains efficiency and modularity that will sustain the filament material quality.

The first suggestion is having the cooling subsystem change based on temperature. This can be done through temperature sensors contacting directly with the filament to transmit real-time data for self-adjustment. The other suggestion is working alongside the other subsystem to adjust according to the tension levels of the filament and the output reading of the measuring subsystem for size correction. Both methods are intended to be autonomous and differ by being either independent or, co-dependent on the subsequent systems.

As forementioned in the “Machine Testing Experiments and Results” document, testing the system will validate how optimal the system is functioning. As a suggestion from previous sprints, determine the best forms of heat dissipation whether it is from material, improving the cooling of the extruder itself or re-design. Revision of knowledge such as heat transfer analysis is recommended to evidently provide accuracy and having known hypothetical results before evaluating the experimental analysis.

7.2. *Measuring Subsystem*

The system is well designed in its ability to measure the correct lux associated with the filament diameters; however, it contains multiple flaws.

7.2.1. *Changing for different filament material*

For the current system, it can only measure one material at a time. This is due to the conversion equation from lux to millimetre of the filament (not yet correct). Since each material has different properties which reflect different light level, additional instructions

will be required should another material be spooled. The authors recommend using a different type of measuring device such as a time-of-flight sensor to measure the filament diameter or a four pin output data calliper to measure but make sure it can correctly produce the data output. Further research into measuring systems is encouraged.

7.2.2. *Laser sensors data change if move filament through after a period*

For the current setup in figure M, it is shown that the laser is consistently on which the light from the laser is shining onto the sensor. This has caused the measurement data of the laser to decrease overtime and reduce the accuracy of the measurement. This is due to many reasons such as thermal effects, saturation of light photo-bleaching and degradation of the sensor due to too much exposure or long-time exposure to the laser. Also, the laser has some issue with connection as the soldering part is not good and may require regular maintenance. Using a different type of laser or turning off the laser when not using is a suggestion. In addition, using other methods of measuring filament like calliper or time-of-flight sensor to measure the diameters is also a recommendation.

7.2.3. *Coding cannot take consistence measurement*

For the current Arduino coding for the measurement of lux, it can take the lux measurement consistently, however if it is connected to other systems, there will be a delay. This is due to the structure of the software of the *loop*. Since it must complete the whole code to repeat the process of measuring the filament again. This creates error and reduces the ability to control filament accuracy immediately. It is recommended to use the thread function to run multiple tasks at the same time such as the TaskScheduler library (see Figure 15.) or use a different controlling system other than the Arduino to manage the systems like Raspberry Pi.

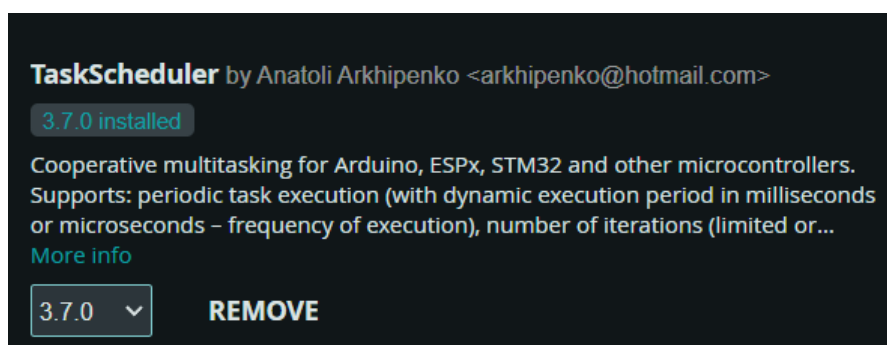


Figure 15. Shows the TaskScheduler library in Arduino.

7.3. Tensioning Subsystem

Recommendations for the tensioning subsystem will be listed in dot points as follows:

- **Spring constant** – Acquire a spring with a known or determine the spring constant of the current spring to perform relevant calculations which may impact slippage and compressive forces acting on the filament tensioned.
- **Adjustable Spring** – The spring force causing the gripping action could have a mechanism which allows for its adjustability.
- **Rollers** – Experiment with varying roller diameters and materials. Additionally, experiment with different filament slots sizes and shapes (the indentation at the centre of the roller).
- **Torque increasers** – currently a timing belt is used to increase the torque of the rollers. An alternative method of achieving this may be the use of a geared-stepper motor.
- **Belt and pulley placement** – currently the belt installed, though possess some tension, still has slack which is problematic should one wish to increase speeds. This problem may be remediated if the stepper motor bracket positioning slots could be easily adjusted then tightened on its base.

7.4. Spooling Subsystem – Winding

7.4.1. Accommodate different sized spools.

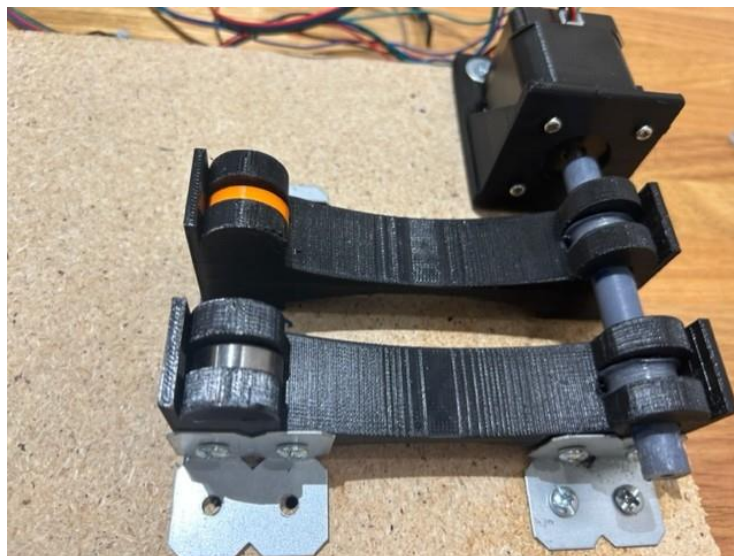


Figure 16. Winding control sub-system.

The winding system is drawn by two pulleys connected to a shaft (Figure 16). To stop the pulleys from slipping the pulleys were superglued to the shaft to ensure translation of the motor to the pulleys. Thus meaning, each pulley was fixed at that position on the shaft and only once spooler was suitable to this system.

7.4.2. *Better Winding gears/pulley system.*

Building on from 6.4.1, the pulleys that were used to drive the system were slipping due to the contact area and a low coefficient of static friction between the two materials. To fix this issue we used double sided tape on the pulleys to help drive the spooling system. Overtime the tape would lose its stickiness and the spooling system would constantly slip. In the future a completely new winding system will be necessary with a new gearing system that will promise winding at both low and high speeds. This system will also need bearing that cause the spooler to slip in the case of tension.

7.5. **Spooling Subsystem – Leveling**

7.5.1. *x-end stopper to home leveller.*



Figure 17. End-Stopper limit switch.

Just like in a 3D printer that adjusts its position every time it turns on, the same principle can be applied to the levelling system to ensure that the location of the level winder is proportional to the amount of filament that is coming out of it. This can also prevent the motor from stalling if it reaches to far the right or too far to the left.

7.6. **Overall recommendations**

- The system (excluding the extruder) should ideally run on one power supply (currently power needs to be supplied individually to: the cooling system, measuring (not functioning), tensioning, and spooling subsystems).
- The system should ideally run on one computer (currently 2-3 Arduino's are required).
- The system should have a user-interface and allow the user to change the settings in accordance with the extruder
- Create an accurate CAD model of the Desktop SJ35 extruder in SolidWorks.
- The current extruding machine (Desktop SJ35) is tall which makes it inconvenient to align parts as a stand is needed. Create a long-term solution stand which is easily adaptable such that this height dilemma is not an issue.
- Should one wish to modify the entire system this will be surprisingly useful.

- As not all CAD files are available, are of differing versions, or have design faults with them, it is recommended that the next team reference our sketches and recreate the CAD models.
- Coding solutions should be found to integrate all four systems into one Arduino. Alternatively, use a different, more appropriate solution.
- Coding solutions for improving feedback loop for measuring-tensioning system. (The use of Threading). (Note Arduino does not support threading).
- Coding solution for level winding system + the integration of an x-stopper to home system.
- Improved spool-winding system to accommodate large spool sizes.
- Water cooling system instead of fans.
- Use standard fastener sizes for the designs.
- Redesign measuring subassembly since the current required a lot of calibration to work correctly.
- System should be fixed to a wooden base to improve portability of the system.
- Tensioning system should have guider to ensure the filament stays on the pulley
- Other filament types should be considered (Velostat, PVA, ABS etc).

8. References

Carmichael, M. (2017, March 18). *V2.1 Overview of the Design Process* [Video]. YouTube
<https://www.youtube.com/watch?v=SY0uivnmB0s>

Electronoobs. (2019, June 9). 3D filament meter caliper Arduino i2c tutorial.
https://electronoobs.com/eng_arduino_tut93.php

Guthrie, G. (2020, November 4). Qualitative risk analysis vs quantitative risk analysis: What's the difference? Nulab. <https://nulab.com/learn/project-management/qualitative-risk-analysis-vs-quantitative-risk-analysis-whats-difference/>

Random Nerd Tutorials. (2022, March 10). Arduino with BH1750 Ambient Light Sensor | Random Nerd Tutorials. Random Nerd Tutorials.
<https://randomnerdtutorials.com/arduino-bh1750-ambient-light-sensor/>

Hendra P Syahputra. (2022, April 16). Desktop Filament Extruder Pt 2 : Diameter Sensor #filamentextruder #3dprinting #3dprinter #arduino. YouTube.
<https://www.youtube.com/watch?v=vE6tmdFhxbk>

9. Appendix

9.1. *Microsoft Teams Planning - Jeremy*

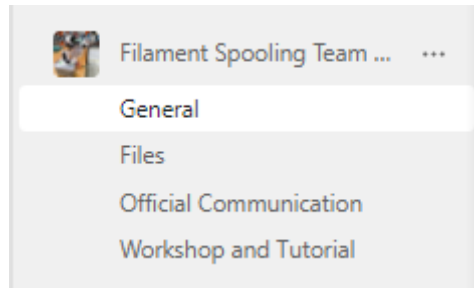


Figure 18. Filament Teams channel

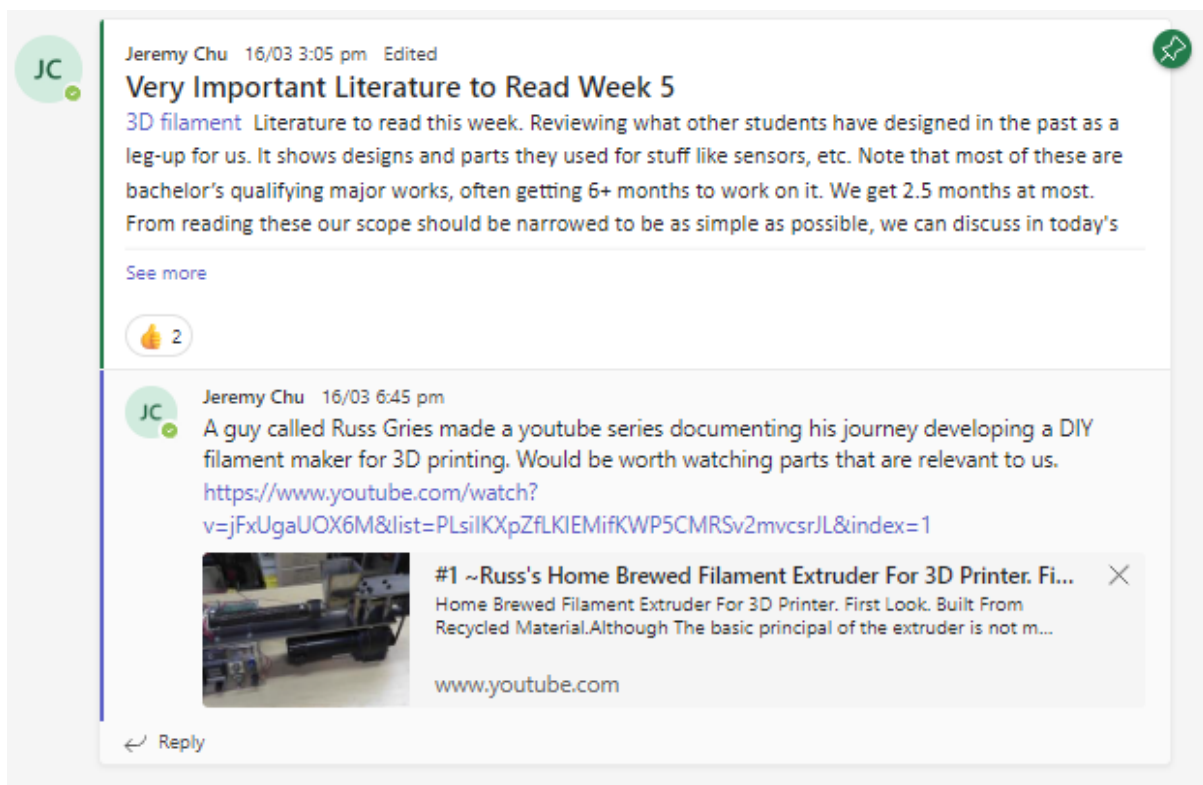


Figure 19. Teams Channel – Research

9.2. *Meeting Minutes*

- See attached document, “Meeting Minutes”

9.3. *Assembly Instructions*

- See 3D CAD Files

9.4. *Technical Drawings – Jeremy and Jonathan*

- See attached Engineering Drawings

9.5. Wiring Schematics – Mark

NOTE: The wiring schematics shown are similar examples to the existing system. Due to time constraints, high-level detailed diagrams could not be made. However, the overall system is relatively intact thereby, should provide ease of understanding the wiring structure of each system in real observation.

9.5.1. Measurer Wire Schematic

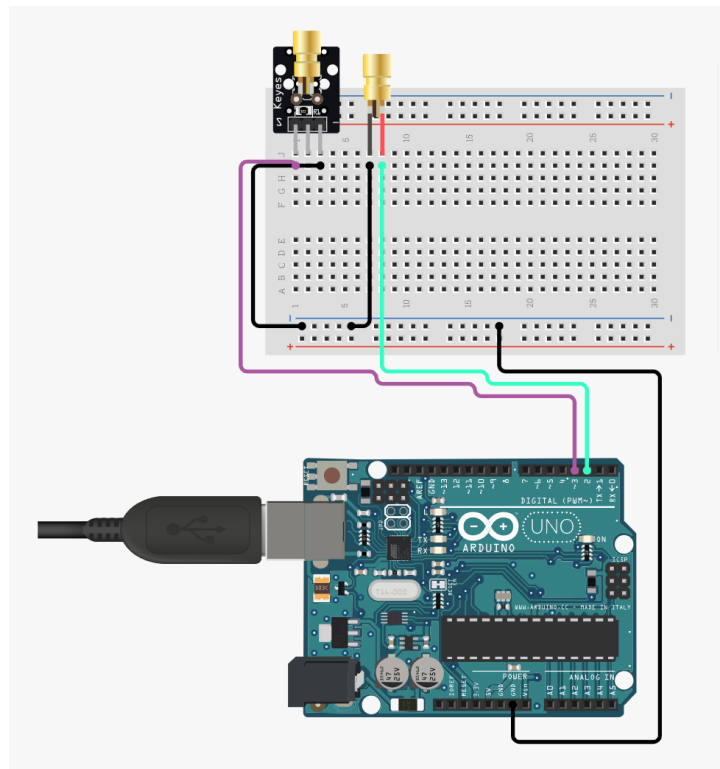


Figure 20 (a). Measure Wire Schematic

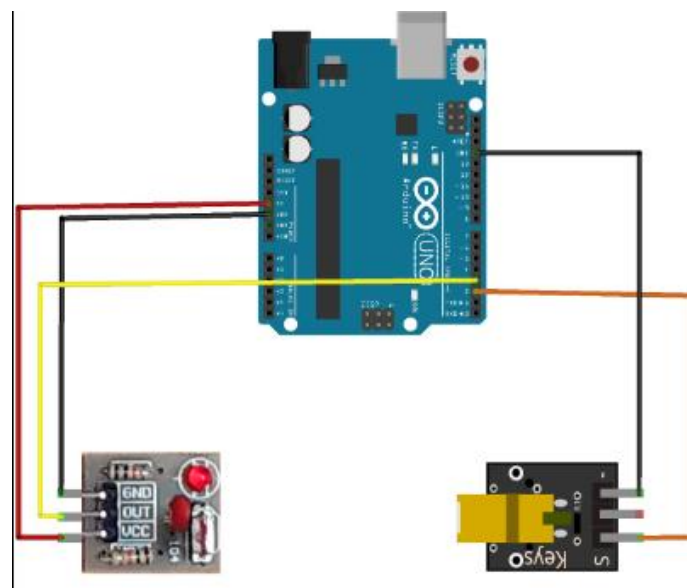


Figure 20 (b). Measure Wire Schematic

9.5.2. *Light Sensor Wiring Schematic (Random Nerd Tutorials, 2022)*

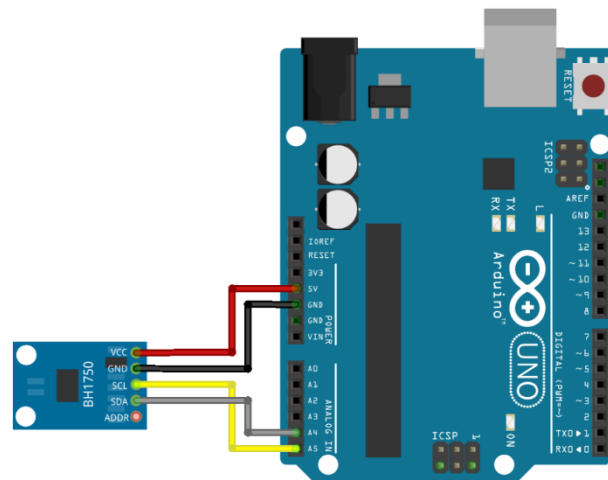


Figure 21. Light sensor wiring schematic. (Random Nerd tutorials, 2022).

9.5.3. *Calliper Sensor Measuring Wire Schematic (Potential) (Electronoobs, 2019)*

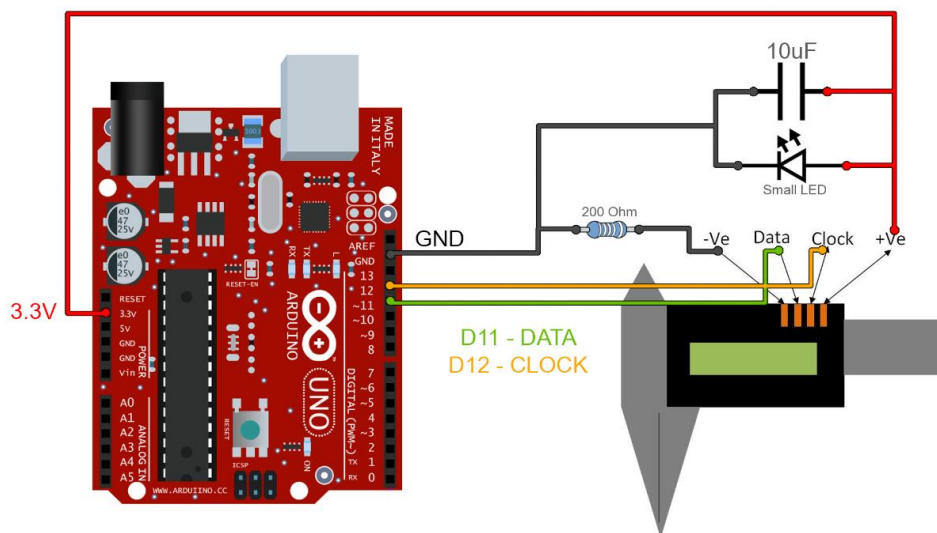


Figure 22. Caliper sensor Measuring wire schematic. (Electronoobs, 2019).

9.5.4. Cooling Wire Schematic

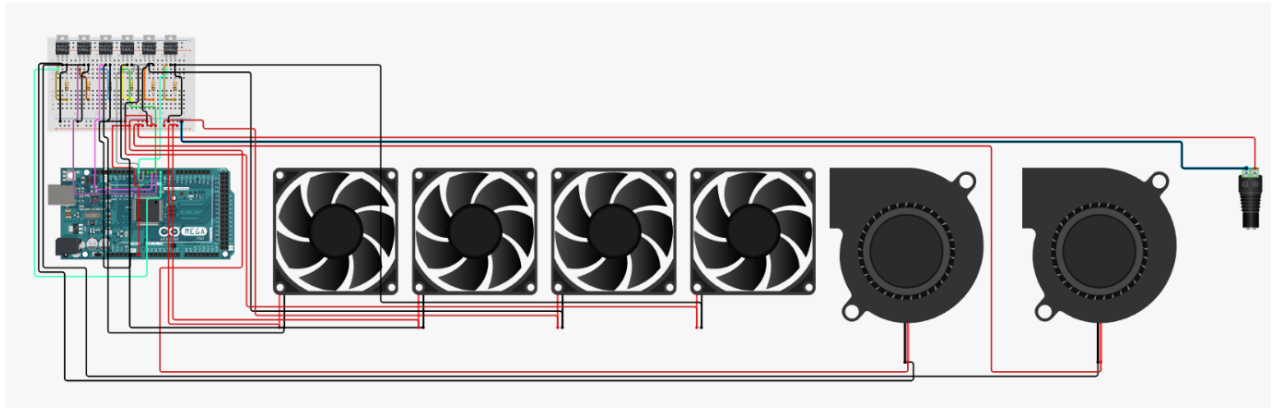


Figure 23. Cooling Wire Schematic

9.5.5. Stepper Motor Control for Winding/Tensioning Wire Schematic

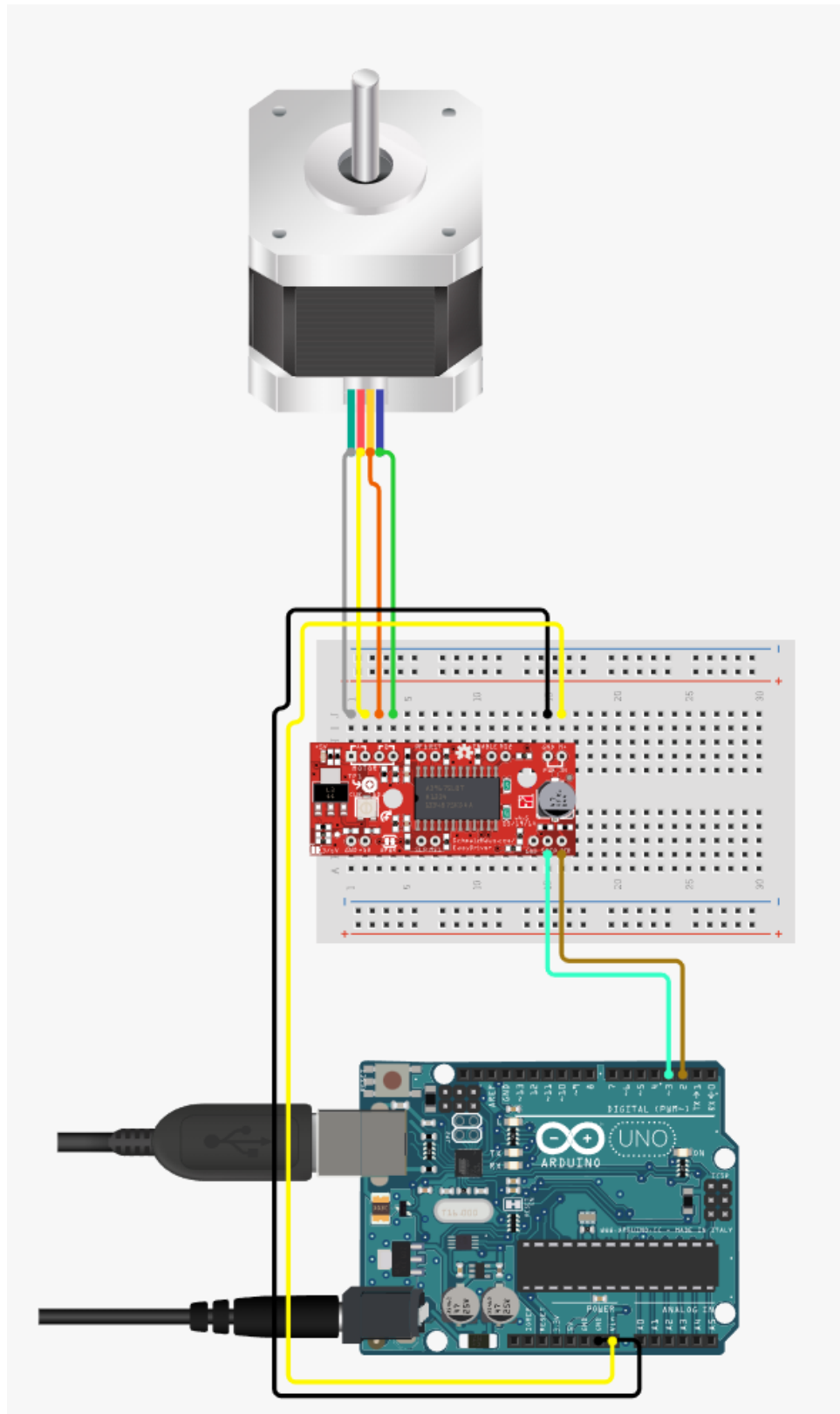


Figure 24. Stepper Motor Control for Winding/Tensioning Wire Schematic

9.6. Coding

9.6.1. Cooling systems

```

////////////////////////////////////
/***** Closed Loop Control Code *****/
////////////////////////////////////

// Define tachometer input pin for fan speed feedback
const int tachPin = 2; // Digital input pin connected to fan's tach wire

// Define desired fan speed setpoint (e.g., 50% of maximum speed)
const int setpoint = 128; // 0 to 255 (0% to 100% duty cycle)

// Variables
int fanSpeed = 0; // Current fan speed (0 to 255)
int error = 0; // Error between setpoint and actual fan speed

void setup() {
  // Initialize PWM and tach pins
  pinMode(pwmPin, OUTPUT);
  pinMode(tachPin, INPUT);

  // Start serial communication
  Serial.begin(9600);
}

void loop() {
  // Read current fan speed from tach wire
  fanSpeed = analogRead(tachPin);

  // Calculate error
  error = setpoint - fanSpeed;

  // Adjust PWM signal based on error
  analogWrite(pwmPin, setpoint);

  // Print fan speed and error to serial monitor
  Serial.print("Fan Speed: ");
  Serial.print(fanSpeed);
  Serial.print(", Error: ");
  Serial.println(error);

  // Delay for stability (adjust as needed)
  delay(100);
}

```

Figure 25. Closed-loop control code - Fan

```

//***** MOSFET Code *****/
//*****

int fanPin = 9;          // PWM pin connected to the transistor or MOSFET
int tachometerPin = 2;   // Digital input pin connected to the fan's tachometer/pulse output
int rpm;                // Variable to store the fan speed in RPM

void setup() {
  // put your setup code here, to run once:
  pinMode(fanPin, OUTPUT);
  pinMode(tachometerPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  rpm = readRPM();        // Function to read the fan speed from the tachometer/pulse output
  Serial.print("Fan Speed (RPM): ");
  Serial.println(rpm);

  // Control fan speed based on the measured RPM value
  if (rpm < desiredRPM) {
    analogWrite(fanPin, 255); // Full speed
  } else {
    analogWrite(fanPin, 0);    // Stop or reduce speed
  }

  delay(1000); // Delay for 1 second
}

int readRPM() {
  // Function to read and calculate the fan speed from the tachometer/pulse output
  // Implement the logic to measure the pulse frequency and calculate the RPM
}

```

Figure 26. MOSFET Code - Fan

```

//***** Temperature and Humidity Sensor Code *****/
//*****

//DS18B20 Temperature and humidity sensor
int temp_sen = 8;
byte dat [5];

void setup () {
  Serial.begin (9600);
  pinMode (temp_sen, OUTPUT);
}

void loop () {
  start_test ();
  Serial.print ("current humidity =");
  Serial.print (dat [0], DEC); // display the humidity-bit integer;
  Serial.print ('.');
  Serial.print (dat [1], DEC); // display the humidity decimal places;
  Serial.println ("%");
  Serial.print ("current temperature =");
  Serial.print (dat [2], DEC); // display the temperature of integer bits;
  Serial.print ('.');
  Serial.print (dat [3], DEC); // display the temperature of decimal places;
  Serial.println ("C");
  delay (700);
}

byte read_data () {
  byte data;
  for (int i = 0; i < 8; i++) {
    if (digitalRead (temp_sen) == LOW) {
      while (digitalRead (temp_sen) == LOW); // wait for 50us
      delayMicroseconds (30); // determine the duration of the high level to determine the data is '0' or '1'
      if (digitalRead (temp_sen) == HIGH)
        data |= (1 << (7-i)); // high front and low in the post
      while (digitalRead (temp_sen) == HIGH); // data '1 ', wait for the next one receiver
    }
  }
  return data;
}

void start_test () {
  digitalWrite (temp_sen, LOW); // bus down, send start signal
  delay (30); // delay greater than 18ms, so DS18B20 start signal can be detected
}

```

Figure 27. Temperature and humidity sensor code

9.6.2. Tensioning systems

Note: With this code, if the step count exceeds that of 1 full revolution with any intermediate step afterwards (e.g., a delay(X) function), abnormalities occur. As such, if you wanted this stepper motor to rotate 2 revolutions only and pause for 1 second, you will need to repeat the myStepper.step(X) function 2 times.

```
TensionerCode.ino
1 // N.B. THE STEPPER MOTORS SHOULD BE SET TO FULL MICROSTEPPING MODE: 1, 1, 1
2 #include <Stepper.h>
3 #define BASESPEED 33600 // 33600 = roller RPM of 7, refer to the "Stepper Motor Calculations for Code.xlsx" sheet
4 #define STEP 28800 // 1 full revolution for the specific roller diameter and its speed ratio
5 Stepper myStepper(6, 5, 4); // Arduino pin allocations for the stepper motor driver
6
7 void setup()
8 {
9   pinMode(6, OUTPUT); //Enable
10  pinMode(5, OUTPUT); //Step
11  pinMode(4, OUTPUT); //Direction
12  digitalWrite(6, LOW); //Stepper is On? LOW = Yes, HIGH = Off
13 }
14
15 void loop(){
16
17   myStepper.setSpeed(BASESPEED); // speed of stepper motor
18   myStepper.step(STEP); // steps of the stepper motor (refer to Excel sheet,
19   //n.b. DO NOT exceed above 1 revolution per step)
20 // IF YOU WANT TO TEST MOTOR PERFORMANCE BY PERFORMING 2 REVOLUTIONS
21 // myStepper.step(STEP); // Repeated myStepper.Steps to program the stepper to move more than one revolution
22 // myStepper.step(STEP);
23 // delay(1000); // delay required in order to see the stepper move particular steps, else infinite rolling
24 }
```

Figure 28. Tensioning system Code.

9.6.3. Measuring systems

Note: Before running the code, some libraries are required to run the sensor

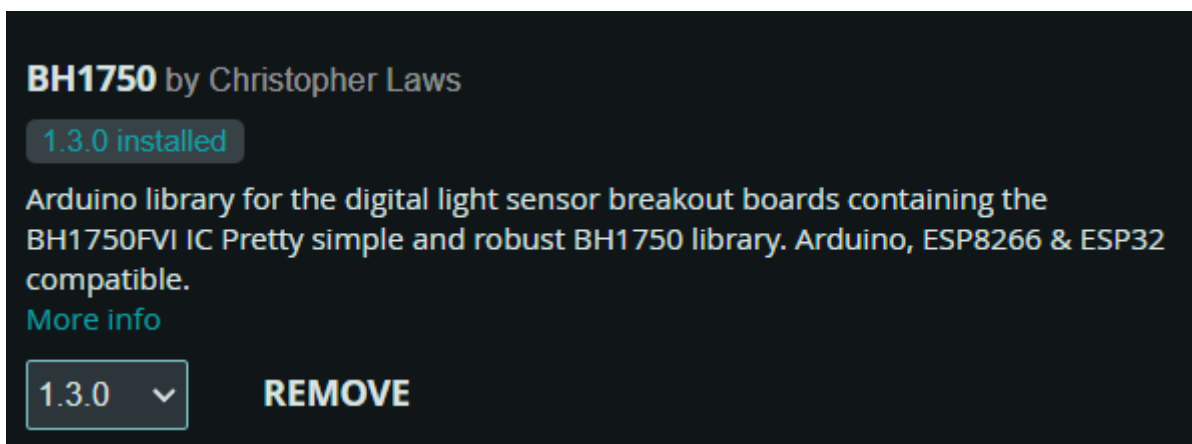


Figure 29. Library for Sensor Control

```

#include <BH1750.h>
#include <Wire.h>
#include <Stepper.h>
#include <math.h>

//Stepper.myStepper(6,5,4);
float diameter;
float max_lux = 70;
float min_lux = 20;
float lux_limit = 9000; // Corrected variable name from lux_limt to lux_limit
const float baseSpeed = 100800;
const float step = 86400;
float speed = baseSpeed; // Initialize with default speed
BH1750 lightMeter;

void setup() {
  pinMode(2, OUTPUT); // Laser
  Wire.begin();
  lightMeter.begin();
  Serial.begin(9600);
  // pinMode(6, OUTPUT); // Enable
  // pinMode(5, OUTPUT); // Step
  // pinMode(4, OUTPUT); // Direction
  // digitalWrite(6, LOW); // Stepper is On? LOW = Yes, HIGH = Off
}

void loop() {
  digitalWrite(2, HIGH); // Laser on
  float lux = lightMeter.readLightLevel();
  Serial.print(lux); Serial.println(" lux");
  diameter = -2.257 * log(lux / 160.28); // this need to be calibrated to match which material is being used
  Serial.print(diameter); Serial.println(" mm");
  // if (lux <= min_lux) {
  //   Serial.println("too big");
  //   speed = baseSpeed * 2;
  // }

void setup() {
  pinMode(2, OUTPUT); // Laser
  Wire.begin();
  lightMeter.begin();
  Serial.begin(9600);
  // pinMode(6, OUTPUT); // Enable
  // pinMode(5, OUTPUT); // Step
  // pinMode(4, OUTPUT); // Direction
  // digitalWrite(6, LOW); // Stepper is On? LOW = Yes, HIGH = Off
}

void loop() {
  digitalWrite(2, HIGH); // Laser on
  float lux = lightMeter.readLightLevel();
  Serial.print(lux); Serial.println(" lux");
  diameter = -2.257 * log(lux / 160.28); // this need to be calibrated to match which material is being used
  Serial.print(diameter); Serial.println(" mm");
  // if (lux <= min_lux) {
  //   Serial.println("too big");
  //   speed = baseSpeed * 2;
  // } else if (lux >= max_lux){
  //   Serial.println("too small");
  //   speed = baseSpeed / 2;
  // }
  // Serial.print(speed); Serial.println(" speed");
  // }
  // for (int i = 0; i < 3; i++) { // 3 is the number of revolution for it to turn before repeating the whole loop
  //   myStepper.setSpeed(speed);
  //   myStepper.step(step);
  //   // Adding a delay between each run, adjust as needed
  // }
  delay(1000); // I put the delay here to show it workes (you can remove it when fully tested)
}

```

Figure 30. Code for Measuring system.

9.6.4. Level - Winding system

```

#include <Stepper.h>

const int stepsPerRevolutionwindingmotor = 200; // Number of steps per revolution
for the winding motor
const int stepsPerRevolutionlevelmotor = 2000; // Number of steps per revolution
for the leveling motor
#define dirPinwind 43 // Direction pin winding
#define enablePinwind 46 // Enable pin winding
#define dirPinleveler 9 // Direction pin leveler
#define enablePinleveler 12 // Enable pin leveler

// Initialize the Stepper library with steps per revolution and pins
Stepper myStepper1(stepsPerRevolutionwindingmotor, 44, 45);
Stepper myStepper2(stepsPerRevolutionlevelmotor, 10, 11);

int speedWinding = 200; // Speed in RPM for winding motor
int speedLeveling = 300; // Speed in RPM for leveling motor

void setup() {
  pinMode(enablePinwind, OUTPUT); // Enable pin winder
  pinMode(enablePinleveler, OUTPUT); // Enable pin leveler
  pinMode(45, OUTPUT); // Step pin wind
  pinMode(11, OUTPUT); // Step pin leveler
  pinMode(dirPinwind, OUTPUT); // Additional direction control pin winder
  pinMode(dirPinleveler, OUTPUT); // Additional direction control pin leveler
  digitalWrite(enablePinwind, LOW); // Enable the stepper driver winder
  digitalWrite(enablePinleveler, LOW); // Enable the stepper driver leveler

  myStepper1.setSpeed(speedWinding); // Set speed for winding motor
  myStepper2.setSpeed(speedLeveling); // Set speed for leveling motor

  digitalWrite(dirPinwind, HIGH); // Set direction to forward winding
  digitalWrite(dirPinleveler, HIGH); // Set direction to forward leveling
}

void loop() {
  static long stepCounter1 = 0;
  static long stepCounter2 = 0;
  static unsigned long lastStepTime1 = 0;
  static unsigned long lastStepTime2 = 0;
  static bool direction1 = true;
  static bool direction2 = true;

  unsigned long currentTime = millis();

  // Calculate the interval between steps based on the speed set
  double interval1 = 60000.0 / (stepsPerRevolutionwindingmotor * speedWinding); //
interval in milliseconds for stepper1
  double interval2 = 60000.0 / (stepsPerRevolutionlevelmotor * speedLeveling); //
interval in milliseconds for stepper2

```

```

// Check if it's time for stepper1 to step
if (currentTime - lastStepTime1 >= interval1) {
    myStepper1.step(1);
    stepCounter1++;
    lastStepTime1 = currentTime;
}

// Check if it's time for stepper2 to step
if (currentTime - lastStepTime2 >= interval2) {
    myStepper2.step(1);
    stepCounter2++;
    lastStepTime2 = currentTime;
}

// Reverse direction for both steppers after a certain number of steps
if (stepCounter1 >= stepsPerRevolutionwindingmotor * 10) { // Example condition
to reverse after 10 revolutions
    direction1 = !direction1;
    digitalWrite(dirPinwind, direction1 ? HIGH : LOW);
    stepCounter1 = 0;
}

if (stepCounter2 >= stepsPerRevolutionlevelmotor * 10) { // Example condition to
reverse after 10 revolutions
    direction2 = !direction2;
    digitalWrite(dirPinleveler, direction2 ? HIGH : LOW);
    stepCounter2 = 0;
}

// Small delay to avoid overloading the loop
delay(1);
}

```

Figure 31. Level – Winding system code

9.7. *Measuring System – Pictures of what was attempted*



Figure 32.

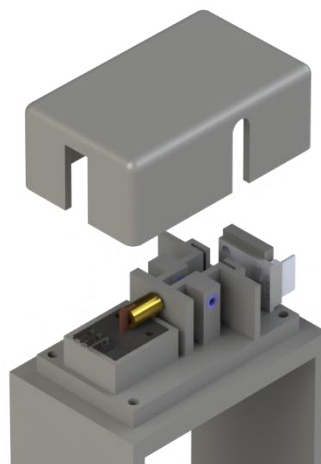


Figure 33.

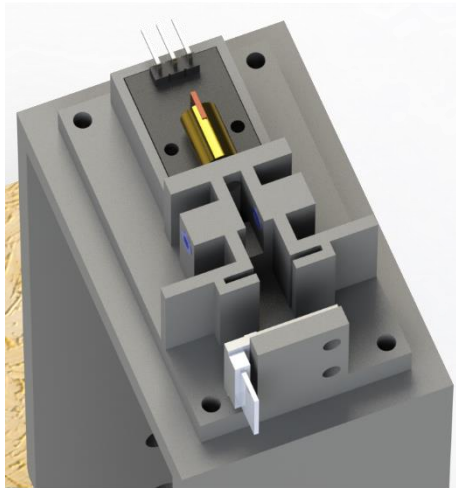


Figure 34.