# Table of Contents

**Purchasing Components:**

- Purchase Order 1 (10/8/2024):
    - Total procurement time: 3 Weeks
    - Aluminum 6061 Plates
        - Procurement Time: 2.5 weeks
        - Manufacturing Time: 3 weeks
            - Better to spend the extra money to get the plates machined from an external supplier to accelerate the assembly process
    - Filastruder:
        - Missing tool changer stepper motor and its corresponding wires
            - Tried to get a replacement part, but Filastruder only offers replacements within 2 week of delivery
                - Didn’t notice missing component whilst confirming the contents of all the packages
                - Realized 6 weeks post delivery once assembling the tool changer component

**Resin Printing:**

- General Settings:
    - Printer: Formlabs 3/3+
    - Material(s): Tough 1500, Grey V4
    - Print Orientation: 45º angle
    - Layer Thickness: 0.1mm
    - Removed generated supports from the inside of screw holes
- Post Processing
    - Component Cleaning:
        - First manual IPA wash for 15-30 seconds
        - Second manual IPA wash for 15-30 seconds
        - Automatic IPA wash for 15 minutes
    - Material: Tough 1500
        - Cure Settings:
            - Temperature: 70º C
            - Time: 60 minutes
    - Material: Grey Resin V4
        - Cure Settings:
            - Temperature: 60º C
            - Time: 30 minutes
- Problems:
    - Material: Tough 1500
        - Print failure while printing back and front feet
            - Both prints failed the first time and succeeded the second time with no changes to the form file
            - Reason for failure: Unknown. Should compare the slicer settings of the nslab desktop to my laptop settings
                - Heat Creep
                    - Research heat creep with formlabs or try different resin type
                - Possible debris inside of the resin tank
                    - Clean thoroughly or replace
                - Dirty/Obstructed UV light
                    - Clean thoroughly or replace
    - Material: Grey Resin V4
        - Heat inserts don't stay inside of components
        - Brittle material, lead to some components snapping/breaking

**Aluminum Top and Bottom Plates:**

- General Information:
    - Purchased 2 Aluminum Plates 6061: T651, 24 in Overall Lg, 24 in Overall Wd, 0.25 in Thick, Mill, +/-0.012 in
        - Procurement time: 2.5 weeks
    - Had the machine shop cut the Aluminum Plates with the water jet
        - Sent .stl cad files
        - Procurement time: 2.5 weeks
- Problems:
    - Aluminum Plates after receiving them from the machine shop:
        - Not all holes were cut all the way with the water jet
            - Was able to tap out some of the holes that were mostly cut with hammer
            - Rest of the holes had to be drilled out
                - 1st Attempt: Took it to FabLab, but the drill press was unable to cut through the aluminum likely due to a lack of speed
                - 2nd Attempt: Took it to the machine shop to use their drill press, but had issues with alignment and clamping due to the size of the plates
                - 3rd Attempt: Clamped the aluminum plates to a table and hand drilled the holes out with the corresponding drill bit size
                    - Finished holes are not perfectly aligned with the corresponding components
        - No finish on the aluminum plates
            - Took the deburring tool to smooth out the rough edges on the aluminum plates
        - Minor issue: plates are dirty
            - Can clean with IPA or acetone
    - Top aluminum plate changed the overall height of the Jubilee. The toolhead can crash into the back Z stopper.
        - Solution: Change the home all function to home x first and y second:
            1. Open the homex.g file and comment out the following lines:

```
if !move.axes[1].homed
  M291 R"Cannot Home X" P"Y axis must be homed before x to prevent damage to tool. Press
OK to home Y or Cancel to abort" S3
  M98 P"homey.g"
```

2. Then open the homeall.g file and switch the order of the homey.g and homex.g. Should look like this:

```
; Home y, x, z, and Toolchanger Lock axes

M98 P"homeu.g" ; X and Z require U to be homed first in case a tool is currently active
M98 P"homex.g"
M98 P"homey.g"
M98 P"homez.g"
```

**CAD Modeling:**

- Heat Inserts:
    - Create a countersink hole that extends through the entire component, allowing the heat insert to be threaded into the resin using a screw and washer
        - The following measurements are inner and outer diameters (ID and OD) depending on the insert size
            - M3:
                - ID: 5.4mm, OD: 6.4mm
            - M4:
                - ID: 5.9mm, OD: 6.9mm
    - For parts that can't have through holes, press fit inserts into component(s) using a vise

- Back/Top Aluminum Plates:
    - Design Process:
        - Create a sketch to represent the horizontal aluminum extrusions or "old frame" (top and bottom) and include all of the vertical aluminum extrusions as well as the PLA components that rest on each frame.
            - To create sketches of the solid parts, create a plane on the surface of the face being drawn and offset any planes if necessary.
                - If a plane was offset to copy a certain detail of the part, after the detail is drawn switch the sketch plane back to its original plane.
        - Create a sketch of the "new frame" that follows the dimensions of the old frame
            - Both top and bottom frames widths were extended by ½" (¼" on each side) to allow for side panels to sit flush with the frames
                - After the new frame is sketched and extruded, use the previous sketches of the vertical aluminum extrusions and PLA components to create corresponding holes for each component
                    - Ensure each hole is a through hole and the correct diameter
                    - Determine which of the holes need to be threaded and which don't
                        - All holes that align with the vertical aluminum extrusions do not need threads as they will thread directly into the extrusion
        - Add any adjustments necessary for the new frame

            - For the top frame, add a slit at the front of the frame to allow for the addition of new tools such as the Opentron Pipette
                - Divide the slit into thirds to provide more rigidity to the structure while maintaining mountability
                - Behind the slit add through holes (Number of holes and spacing is up to you) for tools that need to be mounted from the bottom
            - For the bottom frame, redesign larger feet to cover more surface area and provide more support to compensate for the increased weight
        - Spacers will need to be added to the motors to compensate for the loss in height with the new assembly
            - Calculate the difference in height between the original assembly and new assembly by subtracting the thickness of the horizontal aluminum extrusion (20mm) by the thickness of the top plate (¼” = 6.25 mm)
                - $20mm - 6.25mm = 13.75mm \approx 14mm$ spacers
            - The height of the feet will also need to increased by 14mm to ensure that the motors are suspended above the surface the Jubilee is on
        - Lastly import other solid parts to ensure all holes are properly aligned with the frames

- Panels:
    - Design Modifications:
        - Back Panel (Electronics Panel):
            - Remove 13.65mm from the top and bottom of the panel for it to fit within the dimensions of the modified Jubilee
                - Disregard screw holes removed as those can’t be used anyways
                - Will be mounted using the holes on the side of the panel
                    - Can screw into T-slots mounted to the vertical extrusions
        - Side Panels:
            - Remove 20mm from the top and bottom of both side panels (40mm total removed) for it to fit within the dimensions of the modified Jubilee
                - Disregard screw holes removed as those can’t be used anyways
                - Will be mounted using the holes on the side of the panel

            - Can screw into T-slots mounted to the vertical extrusions

- Automation Deck:
    - General Information:
        - Purpose: Modular frame to hold different beakers, tip racks, sample holders, etc.
        - Material: PLA
        - Size: 305mm x 305mm
    - Design Process:
        - Create a sketch of the build plate using the original magnetic plate as a layout for the automation deck design to get accurate dimensions
        - Extrude the sketch 6.35 mm (¼ in) to create the automation deck part
        - Sketch 4 equal squares (137.5 mm x 137.5 mm) on the automation deck with 10 mm of spacing between each square
        - Design dovetail connections on the edges of each square to create modular insertions
        - Extrude remove the 4 squares + dovetail connections for the automation deck holders to be inserted
        - Divide design into 4 parts with connectors to fit inside of Bambu X1E build volume

- Automation Deck Holders:
    - Design process:
        - Create a sketch using one of the modular holes from the automation deck as a layout to get accurate dimensions
        - Offset the sketch inwards by 0.1 mm
            - This is to ensure that the holder can slide into place without force, but still provide a snug fit
        - Extrude the sketch 6.35 mm (¼ in) to match the height of the automation deck
        - Sketch the dimensions of the object (tip rack, vial holder, wellplate, beaker, etc.) on the center surface of the holder
        - Extrude remove the sketch 4.35 mm
            - Leave 2 mm of space between your object and the magnetic build plate to keep the two from making direct contact

- Automation Deck Holder - Sample Holder:
    - Dry Sample Holder Design Process:
        - Sketch a rectangle that measures 85.5 mm x 127.75 mm
        - Extrude the rectangle 60 mm

        - Create the sample slots:
            - Create a construction line 22.5 mm away one of the 127.75 mm sides of the rectangular prism
            - Sketch a center point rectangle on the construction line that measures 27 mm x 3 mm
            - Create a 2 mm outward offset on this rectangle
            - Dimension the 2 rectangles to be 5 mm from one of the 85.5 mm sides of the prism
            - Create a linear pattern of the rectangles along the surface of the prism
                - This patter should be 10x in the vertical direction and 2x in the horizontal direction
                - Spacing between the patterns should be 12.306 mm
        - Extrude remove the slots 53.65 mm
        - Add 5 mm fillets to the corners of the base
        - On the surfaces of the front slots, sketch a center point rectangle that measures 53.65 mm x 23 mm
        - Extrude remove all the way through the object
        - Finally, face blend the base of each slot with a radius of 2.5 mm
    - Wet Sample Holder Design Process
        - Design process is the exact same as the dry sample holder, but the dimensions are slightly larger
            - These slots will be slightly bigger to allow for more space between the sample and the holder
        - Sketch a rectangle that measures 85.5 mm x 127.75 mm
        - Extrude the rectangle 60 mm
        - Create the sample slots:
            - Create a construction line 22.5 mm away one of the 127.75 mm sides of the rectangular prism
            - Sketch a center point rectangle on the construction line that measures 29 mm x 5 mm
            - Create a 2 mm outward offset on this rectangle
            - Dimension the 2 rectangles to be 4 mm from one of the 85.5 mm sides of the prism
            - Create a linear pattern of the rectangles along the surface of the prism
                - This patter should be 10x in the vertical direction and 2x in the horizontal direction
                - Spacing between the patterns should be 12.306 mm
        - Extrude remove the slots 53.65 mm

        - Add 5 mm fillets to the corners of the base
        - On the surfaces of the front slots, sketch a center point rectangle that measures  53.65 mm x 20 mm
        - Extrude remove all the way through the object
        - Finally, face blend the base of each slot with a radius of 1.65 mm

- Vacuum Tool:
    - Design Process:
        - Start by sketching a circle with a 14 mm diameter
            - This will be the part that contacts the sample
        - Extrude the circle 5 mm
        - On the surface of the cylinder, sketch 2 circles from the centerpoint. ID: 5.12 mm, OD: 7.12 mm
        - Then sketch a path perpendicular to the surface of the cylinder with a height of 20 mm and a length of 25 mm
        - Sweep the face of the circles sketched on the surface of the cylinder using the perpendicular line as the path of the sweep
        - Add a 5 mm radius fillet to the circle at the base of the sweep
            - This should create a sturdy connection between the original cylinder and sweep
        - On the bottom surface of the cylinder, sketch a 12 mm diameter circle
        - Extrude remove the 12 mm circle by 5 mm
        - Fillet the inner base of the sweep 5 mm
        - On the bottom surface of the cylinder, create a vacuum chuck design.
            - Sketch a 12 mm circle
            - Offset sketch a circle from the 12 mm circle by 2 mm
            - Offset sketch a circle from the 10 mm circle by 1.2 mm
            - Repeat this pattern of offsets (offset by 2 mm, then 1.2 mm) until you can't anymore.
            - Sketch 2 perpendicular rectangles along the horizontal and vertical centers of the base of the cylinder
            - Extrude remove the 1.2 mm spacings 5 mm to finish chuck design
        - At the opposite end of the model (opening of the sweep), create a sketch perpendicular to the drawing to make the hose barb.
            - Sketch 5 mm up from the center of the sweep
            - Sketch a 5 mm line away from the surface
            - Sketch a 1.3 mm line down
            - Sketch a 2 mm line away from the surface
            - Sketch a 0.8 mm line up and then a diagonal line 60 degrees from the vertical line.

            - The length of the diagonal line should be 1.6 mm
        - Copy and paste the barb 7 time to create 8 total barbs
        - Close the drawing then revolve the sketch
        - Fillet the base of the sweep and the hose barb 2 mm
- Offset a plane from the center of the 25 mm length of the sweep
- Create a sketch on that plane and make a rectangle that measures 33.45 mm x 10.45 mm
    - Add 2 points 2.725 mm away from the center of the short walls
- Extrude the sketch 7 mm to connect it to the vacuum tool
- Add 2 mm fillets between the extruded rectangle and the vacuum tool
- Finally, create 3.4 mm through holes using the points created in the rectangle sketch

**Jubilee Assembly:**

- Problems:
    - L shaped vertical aluminum extrusions don’t have long enough threads.
        - Solution:
            - Buy shorter M5 screws to fit within the thread length
    - Holes on the aluminum bottom plates aren’t perfectly aligned with the corresponding components
        - Solution:
            - Redrill the holes (Only works for the components that require through holes and not threaded holes)
    - Tool wires are heavy and are not well suited for traveling over the top of the Jubilee
        - Solution:
            - Reroute the wires

**Degradation Testing:**

- Keyence Analysis:
    - General:
        - 5x Lens
        - Stitched 5 images of the center of the dogbone
- Testing Procedure:
    1. Specimen fabrication
        a. Design and 3D print 40 ASTM D638 Type V dogbone specimens, consisting of 20 PLA and 20 Grey Resin V4 samples.
    2. Solvent exposure
        a. Submerge five dogbones of each material in water, isopropyl alcohol (IPA), and acetone for 24 hours.
        b. Retain five additional dogbones of each material as unexposed control specimens.
    3. Post-immersion inspection
        a. After submersion, visually inspect each specimen using a Keyence confocal laser scanning microscope.
        b. Acquire and stitch approximately five image scans of the mid-gauge section for each dogbone.
    4. Mechanical testing
        a. Test every specimen using a Shimadzu tensile tester in accordance with ASTM D638 procedures.
        b. Export the resulting force–displacement data as a .csv file.
    5. Data analysis
        a. From the recorded data, calculate tensile strength, tensile strain at break, and Young's modulus for each specimen.
- Results:
    - Water:
        - In comparison to the control samples for both PLA and Grey Resin V4, the dog bones submerged in water after visual inspection showed no significant differences in appearance.
    - IPA:
        - In comparison to the control samples for Grey Resin V4, the dog bones submerged in IPA after visual inspection showed no significant differences in appearance.
        - In comparison to the control samples for PLA, the dog bones submerged in IPA after visual inspection showed extreme disfigurement.
    - Acetone:

            - In comparison to the control samples for both PLA and Grey Resin V4, the dog bones submerged in acetone after visual inspection showed no significant differences in appearance.
- Tensile Testing (ASTM D638 Type V):
    - General Information: https://www.ssi.shimadzu.com/products/materials-testing/uni-ttm-system/tensile-test-methods-for-plastics-astm-d638/spec.html

      https://www.astm.org/d0638-22.html

      https://eddieliberato.github.io/blog/2020-05-14-dogbones/
        - Dogbone specimen dimensions (ASTM D638 Type V):
            - Overall length: 63.5 mm
            - Gauge length: 7.62 mm
            - Width of narrow section (gauge width): 3.18 mm
            - Length of narrow section: 9.53 mm
            - Distance between grips (overall length of reduced section + grip sections): 25.4 mm
            - Width of ends (grip section): 9.53 mm
            - Thickness: 3.2 mm (nominal)
            - Radius of fillet (transition between narrow and grip sections): 14 mm
        - PLA Print Settings:
            - Printer: Bambu Lab P1S
                - AMS Attachment
            - Filament Type: White PLA Basic
            - 0.4mm nozzle
            - No supports
            - 0.2mm layer height
            - 6 perimeter layers (This is especially critical for parts that have heat-set inserts)
            - 20% infill * printable in the provided orientations
            - outer perimeter layer first
            - uncheck "Detect Thin Walls" if your Slicer has this setting.
                - Otherwise, small gaps will not be filled.
        - Grey Resin V4 Print Settings:
            - Printer: Formlabs 3/3+
                - Print Orientation: 45° angle
                - Layer Thickness: 0.1mm
            - Cure Settings:

    - Temperature: 60º C
    - Time: 30 minutes
- Data analysis script description:
  1. Mount Google Drive and define the data folder.
     a. Connects to Google Drive (drive.mount) and sets a folder path containing tensile-test CSV files.
  2. Locate all CSV data files.
     a. Scans the selected folder and lists every .csv file to be analyzed.
  3. Define sample geometry (ASTM D638 Type V).
     a. Sets:
        i. Gauge length = 7.62 mm
        ii. Width = 3.18 mm
        iii. Thickness = 4 mm
        iv. Cross-sectional area A = width × thickness ($mm^2$)
  4. Loop through each CSV file.
     a. For every sample:
        i. Load the file as a DataFrame with columns Time (s), Force (N), and Stroke (mm).
        ii. Remove commas or stray characters and convert text to numeric values.
        iii. Drop any rows containing invalid or missing data.
  5. Compute stress and strain.
     a. Stress (MPa) = Force (N) ÷ Area ($mm^2$)
     b. Strain = Stroke (mm) ÷ Gauge length (mm)
  6. Estimate Young's modulus.
     a. Select the linear elastic region where strain $\leq 0.002$.
     b. Perform a linear regression of stress vs. strain.
     c. The slope of the fitted line gives Young's modulus (MPa).
  7. Determine tensile strength and elongation.
     a. Tensile strength: maximum stress value.
     b. Tensile strain: strain corresponding to that maximum stress.
  8. Store all calculated results.
     a. Saves the computed tensile strength, tensile strain, and Young's modulus for each sample into a results list.
  9. Create a summary table.
     a. Converts results into a single DataFrame and adds an "Average" row showing mean values for all samples.
  10. Display the summary.
      a. Prints a formatted summary table listing all sample files with their corresponding calculated properties and overall averages.

11. Export results to Excel.
    a. Writes the summary DataFrame to batch_summary.xlsx in the same Drive folder for record-keeping.

- Future Improvements
  - Test multiple different resins
    - Issue:
      - Grey Resin V4 is very brittle and not ideal for lots components
    - Solution:
      - Test another Formlabs resin such as Tough 1500 which is more flexible than Grey
  - Isolate dogbone samples after printing
    - Issue:
      - Material properties of the samples will change depending on the environment it is left in causing variations in Tensile Data
    - Solution:
      - Place dogbones in a dry box to prevent environmental effects

**Vibration Testing:**

- **Standalone Vibration Data Logger**
    - Components:
        - Adafruit MPU-6050 6-DoF Accel and Gyro Sensor - STEMMA QT Qwii
        - Raspberry Pi Pico WH - Pico Wireless with Headers Soldered
        - Adafruit PCA9548 8-Channel STEMMA QT / Qwiic I2C Multiplexer - TCA9548A Compatible
        - STEMMA QT / Qwiic JST SH 4-Pin Cable
        - Bread Board
        - Jumper wires
        - 3D printed MPU-6050 holders
    - Testing locations:
        - Tool changer
        - Bed
        - Frame
    - Testing procedure:
        1. Hardware setup
            a. Connect the Raspberry Pi Pico WH and one Adafruit MPU-6050 accelerometer on a breadboard using jumper wires.
            b. Connect the I²C multiplexer to the MPU-6050 via a JST 4-pin cable.
        2. Sensor mounting
            a. Mount the Raspberry Pi + MPU-6050 breadboard assembly onto the magnetic build plate.
            b. Mount the other two MPU-6050 sensors to the tool head and the aluminum vertical extrusion, respectively.
        3. Electrical connections
            a. Connect all mounted MPU-6050 boards to the multiplexer using JST connector cables.
            b. Power the Raspberry Pi Pico WH through the micro-USB connector attached to a computer.
        4. Software setup
            a. Upload the triple_accel.py data-acquisition script to the Raspberry Pi using Thonny.
            b. Upload the X-, Y-, and Z-axis simulation scripts to VS Code on the host computer.
        5. Data acquisition
            a. Run the triple_accel.py script on the Raspberry Pi.
            b. After the initial sensor calibration completes, start the corresponding axis simulation script in VS Code.

6. Test completion
    a. When the VS Code terminal indicates that the simulation has finished, stop the triple_accel.py script to end data collection.
7. Data handling and analysis
    a. Download the generated .csv data file from the Raspberry Pi.
    b. Process the file using the vibration-analysis Python script to generate FFT and time-domain plots.
8. Repetition
    a. Repeat the complete procedure for each axis (X, Y, Z) on both the modified and original Jubilee platforms.

- Data collection script description:
    1. Pick an I²C bus that actually works.
        a. Tries a few common Pico pin combos and selects the first config that returns any devices on a scan.
    2. Probe what's on the bus.
        a. Scans for a PCA9548A I²C multiplexer and for any directly connected MPU6050 (0x68).
    3. If a PCA is present, sanity-check its channels.
        a. Selects channels (0–7), scans each for an MPU6050 at 0x68, records which channels really have sensors, then disables all channels.
    4. Instantiate sensor objects.
        a. Adds a direct MPU6050 if one is found on the main bus.
        b. Adds PCA-routed MPU6050s for the channels that tested positive (e.g., Ch0, Ch1).
    5. (Optional) Calibrate all sensors.
        a. If enabled, asks you to leave the rig still, averages a bunch of samples per sensor, and sets offsets so X≈0 g, Y≈0 g, Z≈+1 g when level.
    6. Let everything settle.
        a. Waits a few seconds (warm-up), then repeatedly tests stability: collects short bursts from each sensor and checks that the standard deviation of X, Y, Z, and |accel| is below a small threshold. Retries up to a fixed limit.
    7. Choose the run mode: live or log.
        a. Live mode: continuously prints each sensor's X, Y, Z, and acceleration magnitude at the target sample rate.
        b. Logging mode: writes rows to a CSV file at the target sample rate for the specified duration (or until you stop it).
    8. Sample loop (both modes).

        a. On each tick: read all sensors, compute magnitude, mark any failed read as NaN, and keep time with millisecond ticks.
    9. Progress feedback.
        a. Once per second, prints a status line with elapsed time, total samples, and whether any read errors occurred.
    10. Clean finish.
        a. On duration end (or Ctrl+C), stops the loop; in logging mode it closes the file and reports how many samples and sensors were captured.
    11. Core assumptions & constants baked in.
        a. MPU6050 is at ±2 g scale (divides raw by 16384).
        b. Default addresses: PCA = 0x70, MPU6050 = 0x68.
        c. Defaults include sample rate (e.g., 500 Hz), settling window, stability threshold, and which PCA channels to try.

- x-axis script description:
    1. Initialize the controller.
        a. The script starts by defining the Jubilee's IP address and creating an HTTP session to communicate with the Duet 3 web control interface.
    2. Set movement parameters and limits.
        a. It defines safe X-axis limits (e.g., −3 mm to 303 mm) and a default travel speed (e.g., 3000 mm/min).
    3. Send G-code commands through HTTP.
        a. Commands like G1 X... F... are sent to the printer's /rr_gcode endpoint to move the tool head, and responses are read back as JSON.
    4. Check current position.
        a. The code queries /rr_status to get the current X coordinate and uses it as the reference "starting position."
    5. Run a defined number of back-and-forth cycles (default = 3).
        a. For each cycle:
            i. Move the tool head to the minimum X position.
            ii. Wait until motion completes and optionally pause for a few seconds.
            iii. Move the tool head to the maximum X position.
            iv. Wait again and pause.
            v. Log progress and repeat until all cycles are done.
    6. Wait for motion to finish before continuing.
        a. Uses M400 and periodic status checks to confirm the machine is idle before proceeding to the next move.

7. Pause at end points for sensor stability.
    a. Short pauses (e.g., 5–10 s) give time for accelerometer data collection or vibration settling before direction changes.
8. Return to the starting position.
    a. After completing all cycles, the script moves the X-axis back to the original position where the test began.
9. Handle errors gracefully.
    a. If anything fails (connection loss, timeout, etc.), it logs the error and tries to return the tool head to the starting position.
10. Provide detailed logging output.
    a. Throughout the run, timestamps and messages are written to the console describing every movement, pause, or error condition.

- y-axis script description:
    1. Connect to the Jubilee printer.
        a. The script begins by creating an HTTP connection to the Duet 3 controller using the printer's IP address.
    2. Define motion limits and speed.
        a. Safe travel bounds for the Y-axis (e.g., −34 mm → 280 mm or 295 mm) and a default feed rate (≈ 3000 mm/min) are set.
    3. Send motion commands via HTTP.
        a. G-code commands such as G1 Y### F### are sent to the /rr_gcode endpoint to move the toolhead along the Y-axis.
    4. Read current position.
        a. The script queries /rr_status to determine the current Y coordinate, which is saved as the "starting position."
    5. Run a multi-cycle travel routine.
        a. The head moves from minimum → maximum → minimum Y positions for a set number of cycles (default = 3).
    6. Pause at each extreme.
        a. After reaching each limit, the machine waits several seconds (e.g., 5–10 s) to allow for vibration settling or acceleration-sensor sampling.
    7. Wait until motion completes.
        a. The script issues an M400 command and repeatedly checks printer status to confirm the move has fully finished before continuing.
    8. Repeat the pattern.
        a. Each back-and-forth motion counts as one cycle; short pauses occur between cycles to ensure consistent timing.
    9. Return to the start position.
        a. Once all cycles finish, the head moves back to its original Y

location and confirms completion.

10. Handle errors safely.
    a. If communication or motion fails, the script logs the problem and attempts to return the toolhead to its starting point before exiting.
11. Log progress to the console.
    a. Every major action—move, pause, completion, or error—is time-stamped and printed for test documentation.

- z-axis script description:
  1. Connect to the Jubilee’s controller.
     a. The script starts by creating an HTTP session to communicate with the Duet 3 web control interface using the Jubilee’s IP address.
  2. Define safe motion parameters.
     a. It sets a safe vertical travel range (e.g., 10 mm → 290 mm) and a conservative default speed (around 2000 mm/min) since the Z-axis moves the entire tool head assembly.
  3. Send G-code commands over HTTP.
     a. Commands such as G1 Z### F### are sent to move the tool head up or down, and /rr_gcode responses confirm successful communication.
  4. Determine the starting Z position.
     a. The script queries the printer’s /rr_status endpoint to read the current Z coordinate, storing it to return to later.
  5. Run a three-cycle motion routine.
     a. For each of the three cycles:
        i. Move down to the minimum Z limit (closest to the bed, but safely above it).
        ii. Wait until motion finishes and pause for a few seconds.
        iii. Move up to the maximum Z limit (near the top of travel).
        iv. Wait again and pause.
        v. Record progress and continue.
  6. Confirm each motion is complete.
     a. The script issues M400 (wait for moves to finish) and periodically checks machine status until it reports idle.
  7. Include controlled pauses.
     a. Short dwell times at each endpoint allow vibration damping or sensor data capture before the next move.
  8. Repeat until all cycles finish.
     a. Each up-and-down motion pair counts as one cycle; after three cycles the sequence stops.
  9. Return to the starting height.

    a. Once the final cycle is complete, the head moves back to its original Z position and confirms motion completion.

10. Handle errors gracefully.
    a. If communication or motion fails, the script logs the issue and tries to safely return the tool head to its start position.
11. Provide continuous console feedback.
    a. Every move, pause, and status check is timestamped and printed through the logger, giving a clear record of the full test.

- Data analysis script description:
    1. Load configuration and constants.
        a. Defines which sensors are used (Direct, Ch0, Ch1), their physical placements (e.g., bed plate, vertical extrusion, tool head), and color codes for plotting.
    2. Read CSV data.
        a. Opens a provided CSV file (or all CSVs in a folder) containing acceleration measurements and timestamps.
    3. Identify and clean the time column.
        a. Detects the column containing time or timestamps and converts it into elapsed seconds from the start of recording. It handles both numeric (milliseconds) and datetime formats.
    4. Compute acceleration magnitudes.
        a. For each sensor key (Direct, Ch0, Ch1), it looks for acceleration columns (x, y, z) or precomputed magnitudes, then calculates a single magnitude vector in $m/s^2$.
    5. Determine sampling rate.
        a. Calculates the sampling frequency (fs) by taking the median spacing between time samples.
    6. Skip startup transient data.
        a. Discards the first 5 seconds of readings to avoid startup noise and focus on steady-state vibrations.
    7. Perform FFT (Fast Fourier Transform).
        a. Converts the time-domain acceleration signal for each sensor into a frequency spectrum, representing vibration intensity versus frequency.
    8. Convert units for readability.
        a. Converts accelerations from $m/s^2$ to g ($1 g = 9.80665 m/s^2$) for both numerical reporting and plotting.
    9. Generate frequency-domain plots.
        a. Plots FFT magnitude vs. frequency for all sensors.
        b. Marks and labels the dominant frequency peaks (main vibration

modes).
        c. Adds a legend, grid, axis labels, and a text box summarizing peak frequencies.
        d. Saves the result as raw_fft.png in an output folder.

10. Print numerical analysis results.
    Displays:
    a. Sampling rate and total duration
    b. Number of samples used after skipping 5 s
    c. Frequency resolution
    d. For each sensor:
        i. Top three peak frequencies and magnitudes
        ii. RMS (root-mean-square) vibration level
        iii. Maximum instantaneous acceleration
11. Handle multiple files automatically.
    a. If the input is a directory, it processes every .csv file within and generates separate FFT plots and reports for each one.
12. Command-line usage.
    Run from a terminal as:

    ```
    python vibration_test_final.py --input <path_to_csv_or_folder>
    ```

    The script will automatically produce FFT visualizations and printed summaries.

- Future Improvements
    - Start data collection and axis movement simultaneously
        - Issue:
            - Need to use 2 different programs to communicate with the Raspberry Pi and the Jubilee
        - Solution:
            - Run both scripts on one program
    - Isolate the testing environment
        - Issue:
            - Jubilee is susceptible to many environmental properties which may cause unwanted fluctuations in the data collection
        - Solution:
            - Find a more isolated location to conduct testing

**Color Mixing Demonstration:**

- Components:
    - 350ul wellplate
    - Opentron pipette tip rack
    - 3 scintillation vials
    - Red, yellow, and blue food coloring
    - OT-2 P300 pipette
    - Raspberry Pi High Quality HQ Camera - 12MP
        - 16mm 10MP Telephoto Lens for Raspberry Pi HQ Camera - 10MP
        - Arducam CSI to USB UVC Camera Adapter Board for Raspberry Pi HQ Camera, 12.3MP IMX477 Camera Board
    - 3D printed holders
- Testing procedure:
    1. Tool preparation
        a. 3D print and install all tool holders and tools on the Jubilee platform.
    2. Solution preparation
        a. Prepare and mix red, yellow, and blue food coloring solutions, and dispense them into the scintillation vials.
    3. Automation deck configuration
        a. Define all deck locations in the deck_grid.json configuration file to match the physical layout of vials, tip racks, and the 96-well plate.
    4. Software setup
        a. Upload the run_color_mixing_demo.py script to VS Code.
    5. Dry run validation
        a. Perform a dry test (without liquids) to verify that all deck coordinates are correct and that the machine operates as intended.
    6. Automated color mixing
        a. Execute the color mixing demo to carry out the fully automated pipetting and mixing sequence.
- Simulation script description:
    1. Import all modules and setup environment.
        a. Loads libraries for CSV logging, OpenCV imaging, NumPy math, and the Jubilee control modules for motion, pipetting, and deck layout (vials, plate, tips).
    2. Define machine and user parameters.
        a. Sets IP address, camera specs, vial/tip/well positions, pipette volume calibrations, approach distances, Z-axis speeds, and travel speeds for automation.
    3. Specify batch design.
        a. Defines how many samples to make and what color ratios or explicit

per-color volumes to use (e.g., 50 % R, 30 % Y, 20 % B for 272 µL total).

4. Initialize helper and movement functions.
    a. Provides small utilities to:
        i. Send raw G-code (_gcode)
        ii. Pause for dwell times
        iii. Move above vials, wells, or tips
        iv. Execute approach → pause → dive motions
        v. Handle single-press tip pickup and V-axis eject sequences
5. Setup liquid-handling routines.
    a. Implements low-level steps for:
        i. Aspirating from vials using the V-axis to draw liquid up
        ii. Dispensing into wells and performing a blowout
        iii. Mixing within wells by repeated aspirate/dispense cycles with full volume
6. Define camera functions.
    a. Handles capturing one frame from the connected camera, computing a circular-mask mean RGB value at the image center, and saving images as JPEGs.
7. Compute per-sample color volumes.
    a. For each sample, resolves exact red, yellow, and blue microliter values, using either the per-sample explicit volumes, a global override, or default ratios.
8. Run the main batch loop.
   Initializes the pipette, selects the correct tool, and iterates through target wells:
    a. For each sample:
        i. Pick up and use one dedicated tip for each color (R, Y, B).
        ii. Aspirate from each color's vial and dispense into the destination well.
        iii. Return each tip to its slot.
        iv. Pick up a fourth "mixing" tip and run mixing cycles to homogenize the color.
9. Capture and analyze the result.
    a. Switch to the camera tool and move above the mixed well.
    b. Capture an image and compute mean RGB values (either through OpenCV or the optional image-processing module).
    c. Save the photo and measured RGB values for documentation.
10. Log experiment data to CSV.
    a. Appends a line to color_mixing_log.csv with timestamp, well ID, color volumes, RGB means, camera info, and which tip positions were used.
11. Repeat for all samples.

a. Automatically moves to the next well and repeats pickup, mixing, imaging, and logging until the full batch is complete.

**Automated Dip Coating:**

- Components:
  - Coating solution: spray paint
  - 300ml beaker
  - 25 x 75 x 1mm microscope slides
  - 3D printed holders
  - Rocker 300 Vacuum pump
  - US Solid Motorized Ball Valve
  - Vacuum tubing
- Keyence Analysis:
  - General:
    - 20x lens
- Testing procedure:
  1. Tool preparation
     a. 3D print and install all tool holders and tools on the Jubilee motion platform.
     b. Vacuum setup:
        i. Plug in and place the Rocker 300 vacuum pump
           1. Ensure the pump is on a stable surface with enough ventilation.
           2. Keep the inlet hose barb accessible.
        ii. Connect the vacuum tubing from the pump to the manifold
           1. Attach the tubing securely to the manifold inlet.
        iii. Connect tubing from the manifold to the vacuum tool and US Solid Motorized Ball Valve
           1. Confirm airtight connections on all ports.
           2. Use clamps if needed to prevent leaks.
        iv. Mount the motorized ball valve to the Jubilee frame using a 3D-printed bracket
           1. Ensure the mount supports the weight of the valve and prevents strain on tubing.
           2. Route cables safely away from moving axes.
  2. Solution preparation
     a. Prepare a beaker containing the coating solution and place it on the automation deck.
  3. Automation deck configuration
     a. Define all deck locations in the deck_grid_dip.json configuration file to match the physical positions of the dry tray, wet tray, and coating bath.
  4. Software setup
     a. Upload the run_dip_coating_demo.py script to VS Code.

    5. Dry run validation
        a. Perform a dry test (without the coating solution) to confirm that all deck coordinates are correct and that the machine operates as intended.
    6. Automated dip coating
        a. Execute the automated dip coating routine to coat all designated samples under controlled submersion times and withdrawal speeds.
    7. Post-coating analysis
        a. Analyze the coated samples using a Keyence confocal laser scanning microscope to visualize and measure film thickness.
    8. Data recording and visualization
        a. Record the film thickness of each sample and plot film thickness versus withdrawal speed to evaluate coating uniformity and process performance.

- Simulation script description:
    1. Import local Jubilee + deck helpers.
        a. Loads a JUBILEE controller and deck-grid functions (safe_z, surface_z, z_hover, slides_dry, slides_wet, bath) from local modules.
    2. Define user-tunable parameters.
        a. Sets XY/Z speeds, all pause durations, dip depth, per-sample submersion times & withdrawal speeds, tray grid size, starting slot, number of samples, tool selection (T1), approach clearances, and a side-pickup Y-offset.
    3. Set vacuum valve behavior (inverted logic).
        a. vacuum_close() → hold slide (vacuum ON).
        b. vacuum_open() → release slide (vacuum OFF).
        Both include short delays to let the pneumatic state settle.
    4. Provide convenience moves.
        a. Helpers move to safe Z, and to XY targets above dry tray, wet tray, and bath, using approach heights and small timing buffers.
    5. Implement side-pickup sequence (dry tray).
        a. For a given (row, col):
            i. Close valve first (pre-vacuum).
            ii. Go to XY at Y-offset behind the slide at safe Z; pause.
            iii. Lower to pickup Z; pause.
            iv. Advance in Y to contact the slide; pause.
            v. Keep vacuum closed (holding), then Z-only lift to safe Z.
    6. Implement bath dip sequence.
        a. Move above bath → pause → approach to bath surface → dive to depth → submerge for the per-sample time → withdraw at per-sample feed → pause → return to safe Z.
    7. Implement place sequence (wet tray).

    a. Move above target wet slot → descend to approach then placement Z → pause → open valve to release → pause → back to safe Z → final safeguard: ensure valve is open.

8. Iterate tray positions in column-major order.
    a. Generator yields (row, col) pairs across the configured grid starting from START_ROW, START_COL, until N samples are processed.
9. Select tool and initialize valve.
    a. If enabled, issues T1 and opens valve (release state) before starting the batch.
10. Run the full batch loop.
    a. For each sample index and tray slot: pickup → dip → place, using per-sample overrides when provided, else fallbacks.
11. Log each run to CSV.
    a. Appends a row to dip_coating_log.csv with timestamp, index, source/destination slots, dip depth, submersion time, and withdrawal speed.
12. Honor built-in safety/time buffers.
    a. Frequent safe_z() retreats and short sleep buffers help prevent collisions and let pneumatics settle between actions.
13. Assumptions & prerequisites.
    a. Axes are already homed in Duet; deck coordinates and heights come from your local deck_grid_dip configuration; the IP/connection details live inside your local JUBILEE controller.
14. End state.
    a. After the last sample is placed and logged, prints a batch complete message.

- Future Improvements
    - Redesign automation deck/holders to make it easier for removal
        - Issue:
            - The holders fit perfectly flush into the automation deck which makes it difficult for removal without taking out the screws holding the deck to the magnetic build plate
        - Solution(s):
            - Make handles on the holders for easy removal
            - Redesign automation deck to attach via clips instead of screws
    - Sync python script and Duet mini
        - Issue:
            - The python script sends commands faster than the Duet can execute them causing the machine to fall behind the script leading to errors.

        - Current work around:
            - Add longer pauses after the dipping command is executed
                - The problem with this is that it works for slower speeds, but if you are dipping multiple samples with varying speeds the samples with higher dip speeds will just hover until the pause is finished, wasting valuable time.
    - Not enough dipping solution for all samples
        - Issue:
            - The oil based paint used can not be evenly distributed for consecutive samples
        - Solution:
            - Obvious solution is to use a more viscous solution intended for dip coating
            - Use scintillation vials with paint inside, switch tools from the vacuum to the pipette, then aspirate/disperse more paint into the dipping beaker
    - Vacuum blowing on samples
        - Issue:
            - After release the vacuum blows on the samples causing them to rattle and hit the holder potentially ruining the film
        - Solution:
            - Redesign the wet sample holder to hold the samples in a slanted orientation rather than vertical
            - Reduce the pause time after releasing the sample
    - Redesign holders to avoid ruining the film on at least one side of the sample
        - Issue:
            - Samples lean on the holder creating contact on the film and lessening the uniformity of the film on the surface of the substrate
        - Solution:
            - Redesign holders

**Software:**

- Useful links for the Duet boards
    - Getting connected to your Duet
    - GCode dictionary
- Problems:
    - The Duet 3 Mini 5+ was unable to communicate with the Duet 3 Expansion Board 3HC. The 3HC board powered on (indicated by a slow-blinking red LED), but all communication attempts such as M122 B1 or M115 B1 resulted in CAN response timeouts. This prevented X and Y motors from being enabled or homed.
        - Troubleshooting steps:
            1. Confirmed physical CAN wiring
                - Ensured proper connection from Duet 3 Mini 5+ CAN pins to 3HC CAN IN
                - Verified CANH → CANH and CANL → CANL
                - Swapped CANH/CANL to test for reversed polarity — no change
            2. Verified CAN termination
                - Checked that only two termination jumpers were present (one on Duet 3 Mini, one on 3HC)
            3. Tested with a known-good CAN cable
                - Replaced the CAN cable — no change in behavior
            4. Monitored 3HC LED behavior
                - Observed a slow red blink indicating board was powered but not communicating over CAN
            5. Attempted standard address assignment
                - Ran commands like M952 B1 A121 to assign a new address — all failed with response timeouts
                - Tried setting CAN bus speeds (250k, 500k, 1M) using M952 — still no communication
            6. Scanned for unknown assigned addresses
                - Sequentially queried possible CAN addresses using M122 B0, M122 B1, … M122 B125
                - Success: M122 B121 returned valid diagnostics, confirming the 3HC was already assigned to address 121
            7. Confirmed 3HC health
                - Valid M122 B121 output showed:
                    - Firmware 3.4.1 installed
                    - CAN communication healthy (no packet loss or timeouts)
                    - Drivers responding with no errors

8. Updated config to use correct address
    - Changed all M584, M569, and M574 entries in config.g that referenced 1.x to 121.x
9. Resolved “Failed to enable endstops” error
    - Found that M574 lines were still referencing pins on 1.ioX.in
    - Updated those to 121.ioX.in to match the actual CAN address of the 3HC
10. Successfully homed all axes
    - After fixing the endstop pin mappings and confirming motor driver mappings, all axes were able to home properly using G28

- Underperforming Z motor
    - Troubleshooting steps:
        1. Swapped motor cables
            - Swapped the motor cables between the problematic Z motor and a known-good motor to verify if the issue follows the cable (cable/motor) or stays with the driver (driver/electrical).
        2. Swapped motors
            - Installed a known-good stepper motor onto the problematic driver (0.2 on the Duet 3 Mini 5+).
        3. Checked and attempted to correct G-code configuration
            - Reviewed and attempted to update relevant G-code (config.g, bed.g) to ensure all Z motors are mapped properly.
            - Looked for issues in M584, M92, and M350 commands (driver mapping, steps/mm, and microstepping settings).
            - Result: Identified the need to ensure independent Z drivers during bed leveling (M584 Z0.2 U0.3 V0.4) and matching step/mm settings (M92 Z400 U400 V400) and microstepping (M350 Z16 U16 V16 I1).
        4. Z-axis motors were swapped with the x/y-axis motors
- Loud Idling of the Z and U (tool changer) motors:
    - Troubleshooting steps:
        1. Added stealthChop2 G-code command to the config file
            - Add D3 to the following lines of code:
                - M569 P0.2 S0 D3
                - M569 P0.3 S0 D3
                - M569 P0.4 S0 D3

                - M569 P0.1 S0 D3
    - Z probing would probe in locations off the bed
        - Troubleshooting steps:
            1. Rewire the Z motors to ensure that they match the order

**Tools:**

- Tool Prerequisites:
    - Setting tool parking locations:
        - https://science-jubilee.readthedocs.io/en/latest/building/parking_posts.html
    - Setting tool offsets:
        - https://science-jubilee.readthedocs.io/en/latest/getting_started/new_user_guide.html#setting-tool-parking-post-positions-and-offsets
- Top down camera
    - Hardware:
        - 
    - Issues:
        - Hardware:
            1. Parking post does not align with toolchanger
                - Solution:
                    i. Use original parking post 3d model for Jubilee camera tool
                    ii. Increase the height of the model by 3mm (from 22mm to 25mm)\
                    iii. Move the dowel pin holes up 3mm
- OT-2 Pipette
    - Hardware:
        - Issues:
            1. Parking post does not align with the toolchanger
                - Solution:
                    i. Use 55mm parking post 3d model
                    ii. Move dowel pins holes down 6mm
                    iii. Increase distance between dowel pin holes to 57mm
    - Electronics:
        - Issues:
            1. Miswired the stepper motor connection and fried the memory board on the OT-2 Pipette
                - Solution:
                    i. Remove the plate from the pipette to access the electronics
                    ii. Unplug and remove the fried memory board (not needed for the Jubilee)
                    iii. Verify the functionality of the pipette's internal stepper motor and limit switch

                iv. Solder wires directly onto the limit switch and stepper motor

    - Software:
        - Issues:
            1. Unable to define external pipette limit switch as a Z axis endstop
                - Solution:
                    i. Comment out the following line from the original config file:
                    "M574 Z0"

- Vacuum
    - Hardware:
    - Electronics:
        - Issues:
            1. Valve requires 12V of power
                a. Solution
                    i. Make a 2 pin connector and plug in to one of the 12V fan ports on the Duet 3 Mini 5+
    - Software: