Axolotl : Automated Chemical Synthesis Platform

Aditya Prasad

VISIT 2025, University of Washington

Mentors: Prof. Shijing Sun, Mr. Zuyang Ye

Contents

1. Introduction

1.1. The Dawn of Automated Chemical Synthesis

The field of chemical synthesis is rapidly transforming from traditional manual methods to sophisticated automated approaches. This new dawn is driven by innovative global platforms revolutionizing chemical research and development. Such platforms accelerate time-to-market and reduce R&D costs across industries like pharmaceuticals and advanced materials. Furthermore, the emergence of open-source robotic platforms democratizes access, lowering barriers and fostering broader accessibility, flexibility, and scalability in materials synthesis research. These advancements pave the way for "self-driving laboratories" that integrate machine learning and automation to expedite discovery, generate high-quality data, and enable rapid breakthroughs in clean energy and sustainability, while also reducing chemical waste.

1.2. Jubilee

At the forefront of this transformation are open-source platforms like Jubilee. Jubilee is recognized as an extensible, open-source, multi-tool motion platform designed to execute G-code for various low-force automation applications. While initially configured for multimaterial 3D printing, Jubilee has demonstrated successful adaptation for diverse laboratory tasks, including light liquid-handling.

1.3. Axolotl: A specialised Jubilee

The Axolotl setup, developed at the University of Washington's Sun Lab, is a specialized platform built upon the open-source Jubilee multi-tool motion system, specifically designed to automate complex chemical synthesis tasks. Its primary utility lies in precisely creating solutions by mixing a metal precursor, an organic precursor, and a solvent. To achieve this, Axolotl integrates a suite of four custom-tailored tools: a Dual Syringe, a Single Syringe, Spectrometer with a dip-probe,and Vacuum Gripper. This integrated system leverages Jubilee's capabilities to streamline multi-step chemical processes, enhancing efficiency, reproducibility, and safety in the laboratory.

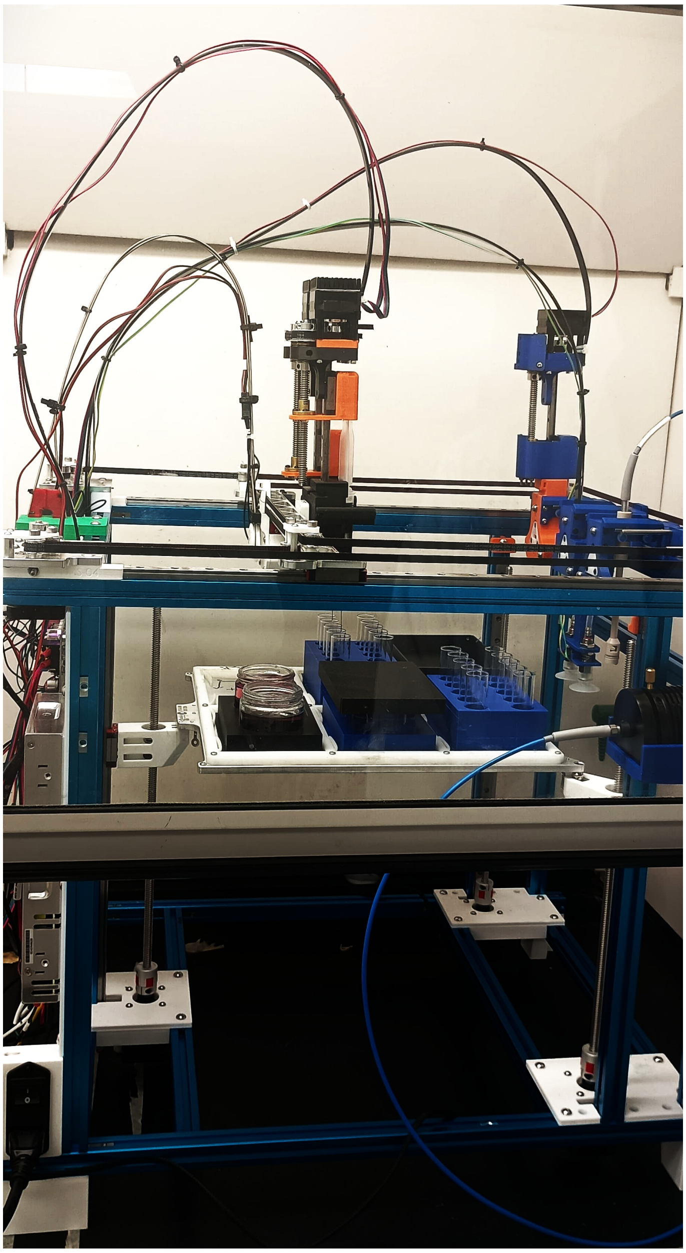

Figure 1. Axolotl: automated chemical synthesis setup

1.4. Metal Organic Frameworks

Metal-Organic Frameworks (MOFs) are a class of crystalline porous materials constructed from metal ions and organic linkers, renowned for their exceptionally high surface areas and tunable pore structures. A prominent example is Zeolitic Imidazolate Framework-67 (ZIF-67), which is formed from cobalt (Co) cations and 2-methylimidazole ligands. ZIF-67 exhibits remarkable thermal and chemical stability, high catalytic activity, and a well-defined porous structure, making it highly versatile. The material can be readily synthesized in solution at room temperature, and its properties can be optimized by adjusting reaction parameters like reactant concentrations. These characteristics make ZIF-67 a prime candidate for a wide array of applications, including catalysis, chemical sensing, gas separation, and energy storage. The Axolotl platform leverages its precision automation to synthesize and explore ZIF-67 for these advanced applications.

2. Hardware Commissioning

Before the Axolotl platform could be trusted to run multi-hour, autonomous experiments, it first had to undergo a rigorous phase of hardware assembly, debugging, and calibration. While the Jubilee kit provides the necessary components, transforming them into a reliable, precision motion system is a non-trivial engineering task.

2.1. U-Axis Assembly and Tuning

The U-axis is a critical component of the Jubilee's multi-tool architecture; it is the dedicated stepper motor that drives the flexible cables to actuate the twist-lock mechanism on the carriage, allowing tools to be picked up and parked. During initial assembly, we faced significant challenges in achieving a reliable lock. The tool plates would either fail to engage securely or would not release cleanly from the carriage. This required a complete disassembly and re-tuning of the U-axis assembly.

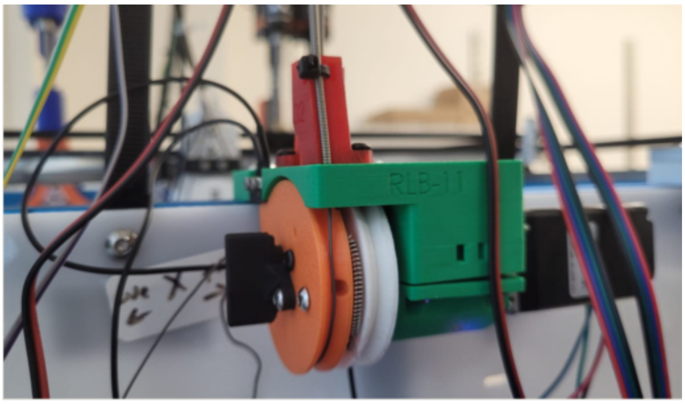

Figure 2. U-axis assembly

2.2. X & Z Limit Switch Repair

Limit switches are fundamental to the machine's operation, as they provide the physical reference points for the homing sequence, which establishes the machine's coordinate system. During initial testing, the X and Z axes exhibited intermittent homing failures. The switches would either fail to trigger or, more problematically, trigger prematurely, causing the homing sequence to fail. The root cause was traced to faulty wiring and loose connections at the switch terminals, similar to issues encountered on the custom tool heads. The solution involved systematically re-soldering the connections for both the X and Z limit switches and rigorously testing the homing cycle dozens of times to confirm that the electrical contact was stable and reliable under all conditions.

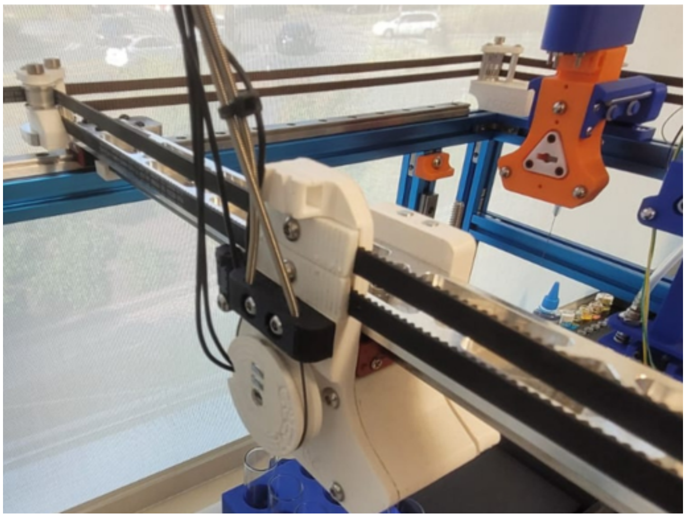

Figure 3. Tool changer assembly

2.3. Syringe Pump Dispense Calibration

A critical part of commissioning the hardware was to determine the operational limits and precision of the custom-built syringe pump tools. To establish the lowest volume that could be dispensed accurately and repeatably, we conducted a series of systematic tests. This involved evaluating different combinations of syringe sizes and needle gauges (0.5mm, 1.2mm, and 1.6mm) to find the optimal setup for low-volume liquid handling.

The performance of each configuration was assessed by measuring the Mean Absolute Error percentage (MAE%) and the standard deviation of the dispensed volumes across a range of target volumes. As shown in the results (Figures 4 and 5), while precision was generally high for volumes above 0.5 ml, the accuracy degraded significantly as the target volume decreased.

The data clearly indicated that the combination of a **10cc syringe with a 0.5mm needle** provided the best performance. This setup achieved an acceptable MAE percentage of approximately 4% at a target volume of 0.1 ml. Below this point, the error for all configurations increased dramatically. Based on this empirical evidence, we established **0.1 ml as the lowest accurate dispense volume** for the Axolotl platform and adopted the 0.5mm needle as the standard for all subsequent chemical synthesis experiments.

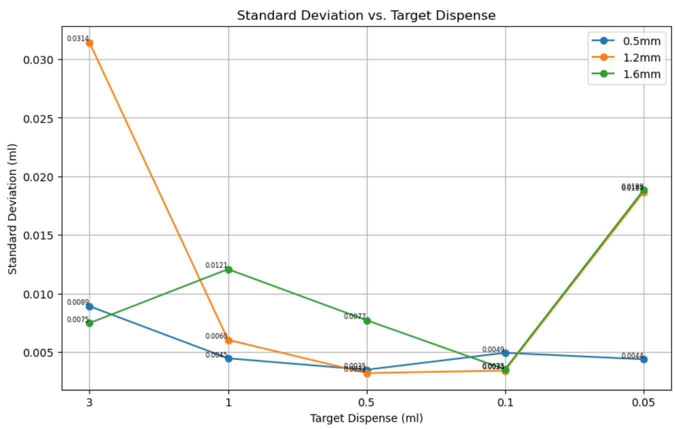

Figure 4. Standard deviation of dispensed volumes for different needle gauges. The 0.5mm needle (blue line) demonstrates consistently high precision across the tested range.

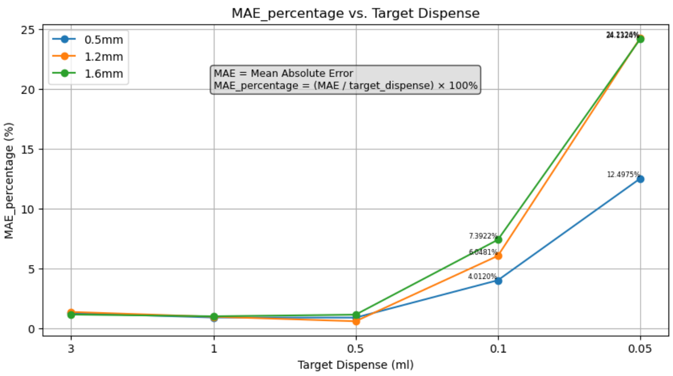

Figure 5. Mean Absolute Error (MAE) percentage versus target dispense volume. The 0.5mm needle configuration maintains an acceptable error of approximately 4% at 0.1 ml, which was determined to be the lowest reliable dispense volume.

3. Axolotl Tools

To make Axolotl capbale of conducting automated chemical synthesis tasks, a suite of four custom designed tools were developed. Each tool addresses a fundamental unit operation required for the synthesis and analysis of materials like MOFs: precision liquid handling, active mixing, labware manipulation, and in-situ characterization. This section details the design of each tool and explores the significant hardware and software challenges encountered during their development and integration.

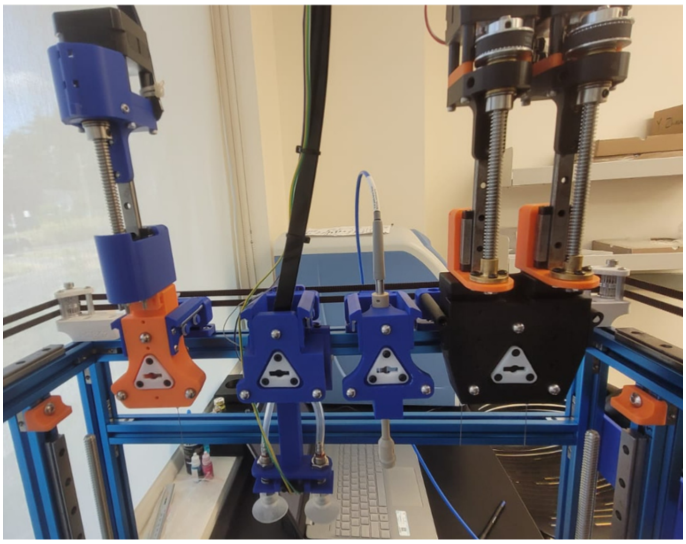

Figure 6. Axolotl Tools

3.1. Double Syringe

The Double Syringe tool is the primary workhorse for liquid handling, designed for the precise and independent dispensing of two different chemical precursors. The design is based on open-source syringe pump CAD models.

3.1.1. Tool Parking

Due to its heavy weight, the dual syringe was slightly inclining towards the side of its tool holder assembly. This incline position created problems with dual syringe parking and posting operations.

To perfect the parking and posting of the dual syringe tool, we perfectly calibrated the positions of the tool holder by making changes to Gcode files on Jubilee Web interface. We printed new tool wings for the dual syringe to prevent error in tool positioning. To furthur smoothen out the parking and posting operations, we applied Spray lubricant on the aluminium rods of the tool holder assembly.

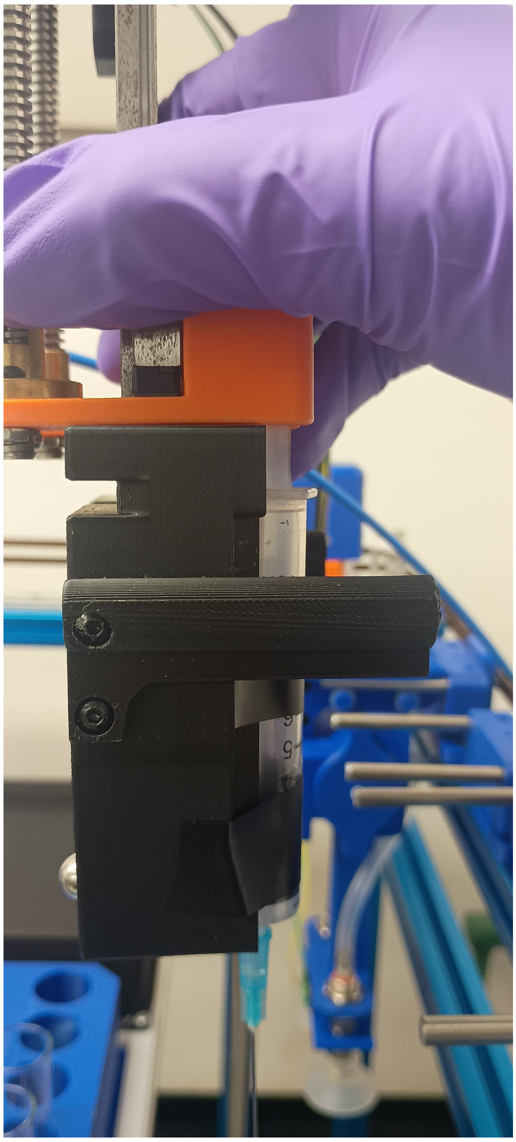

Figure 7. Dual Syringe tool wings

3.1.2. API Improvements

There were many problems with the API code for the dual syringe tool. We faced an error in the dispense call of the dual syringe tool and subsequently corrected it. We added the code for priming the dual syringe whenever a refill call is initiated. The priming of the syringes ensures that there is no loose liquid volume inside the syringe, which in turn makes the dispense liquid volume more precise.

3.2. Mixing Syringe

The mixing syringe or single syringe is used for dispensing solvent into the sample vials and is also used for mixing the contents of the vials. The mixing of the vials is done as a result of repeated aspirate and dispense calls on the single syringe tool.

3.2.1. Hardware Assembly

For the Single syringe tool, we ordered the necessary parts using the assembly guide uploaded on the official Jubilee website. We 3D printed the rest of its parts and subsequently assembled it all-together.

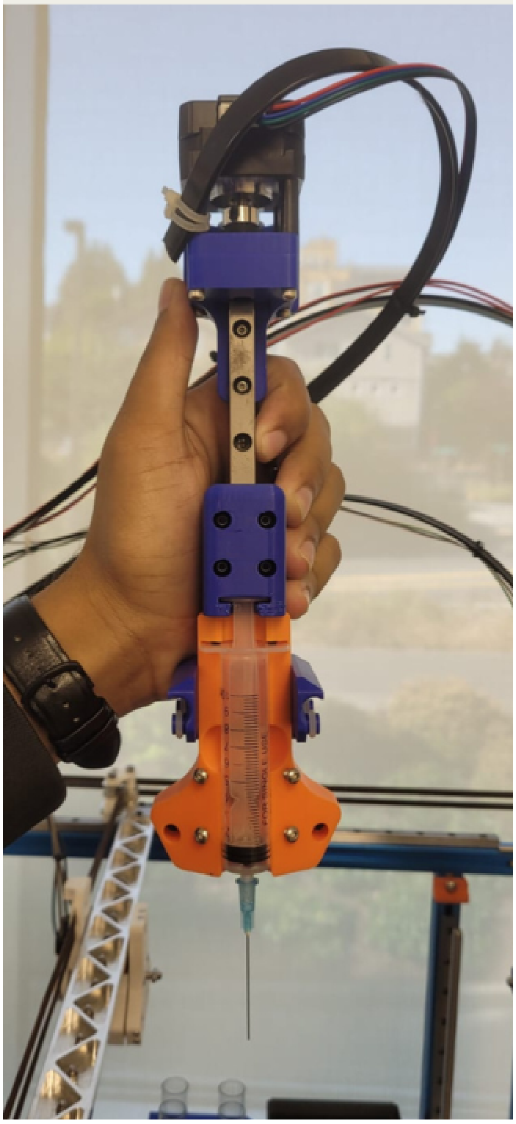

Figure 8. Single Syringe tool

3.2.2. API Code

We wrote the full API code for the single syringe tool. The single syringe API had functions for dispensing, aspirating, resetting position, and mixing operations. We rigorously tested every functionality of the tool before finalizing the tool API. We even added a priming code whenever a refill call is initiated; priming ensures that there is no loose air left in the syringe after every refill.

3.2.3. Calibrating positions

We perfected the tool holder position of the single syringe tool. This ensured smooth parking and posting operations of the tool.

Since this was a new tool, we had to make new config .json files for the labwares wrt to the single syringe tool. After making all the necessary config files we precisely calibrating their manual offset positions.

3.3. Vacuum Gripper

Vacuum Gripper tool is used for picking up and placing lids from one labware to another.

3.3.1. Hardware

We faced problems with the depth limit switch attached on the vacuum gripper tool. Due to loose wiring, the limit switches were prematurely activated. Hence we had to re-solder the wiring connections and test the tool till it worked flawlessly.

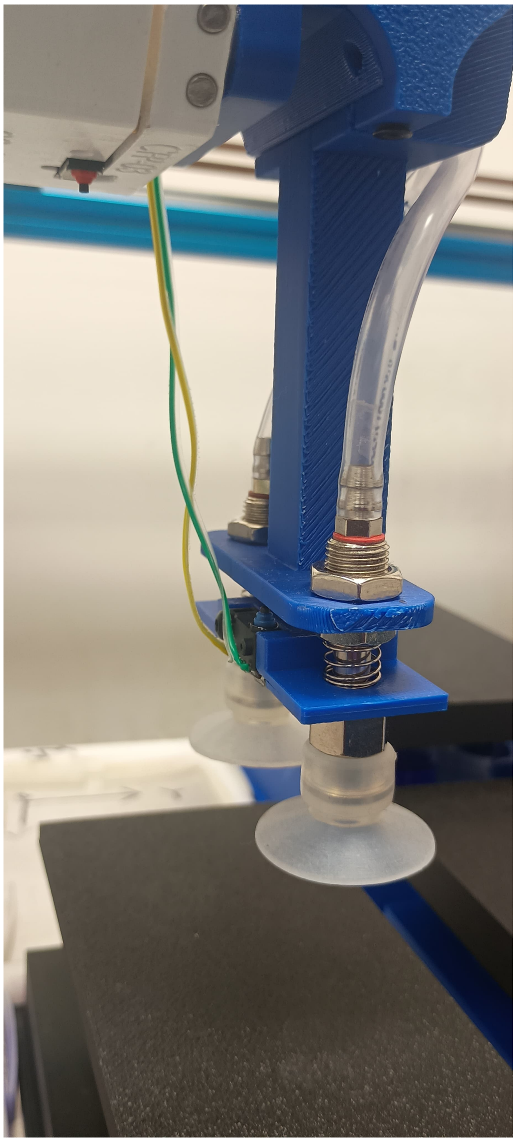

Figure 9. Vacuum Gripper wirings

3.3.2. Calibrating positions

We perfected the position of the tool holder of the vacuum gripper tool. This ensured smooth parking and positioning operations with the tool.

3.4. Spectrometer

We used a dip-probe spectrometer to collected absorbance data of the samples. The dip-probe spectrometer had a mirror attached to its end, which reflects light to the probe. The void space between the probe and the mirror had some water retained into it after every data collection. In order to avoid contamination of samples, we wash the probe in methanol solution after every data collection.

3.4.1. Hardware

Since we earlier worked a thinner probe, the vials were thin. But now, due to the new spectrometer probe, we had to order wider vials which can fit the probe into it. We redesigned the Well-plates and 3D printed those 10-vial well plates. Since these were new labwares, we had to make new config .json files for each of those well-plates and precisely calibrate the postions of the well plates wrt to the spectrometer probe.

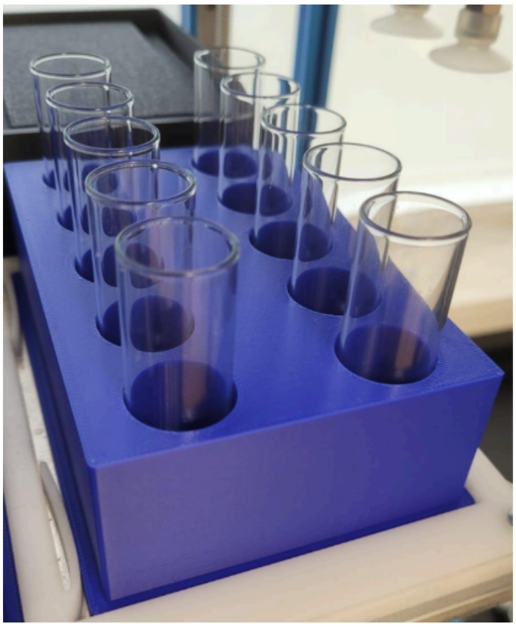

Figure 10. 10 Vial Well Plates

The new wider vials had a total liquid capacity of 14ml. Since during MOF synthesis we make MOF samples in total liquid volume of usually around 6-7ml, thus the depth of the solution level was too low for the probe to reach till that depth. Hence we redesigned and 3D printed a new probe holder for the spectrometer.

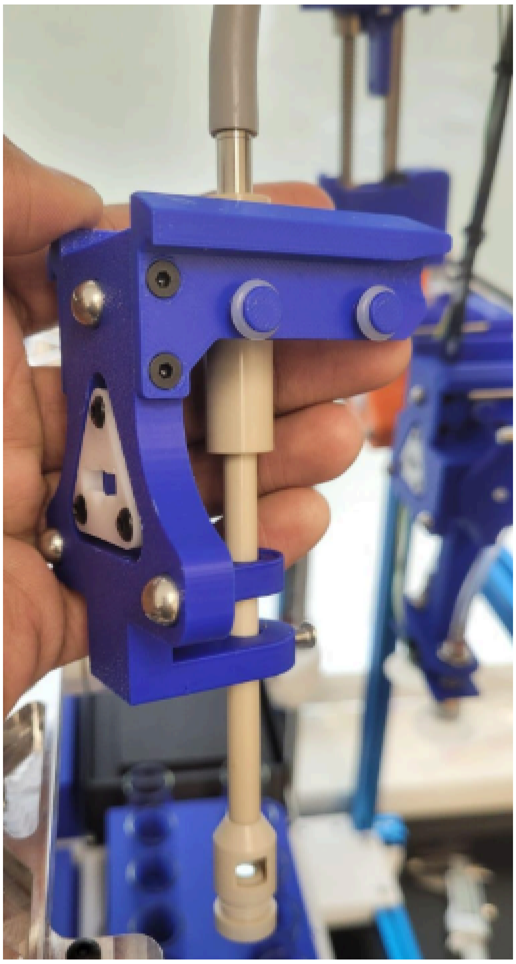

Figure 11. Spectrometer Probe Holder

3.4.2. Calibrating positions

We perfected the tool holder position of the spectrometer tool. This ensured smooth parking and posting operations of the tool. To prevent the probe from hitting the vials, we precisely calibrated the positions of the labwares wrt to the spectrometer tool.

3.4.3. API Improvements

We made many changes to the OceanDirect spectrometer API. We corrected the errors in the collect_spectrum and compute_absorbance functions of the OceanDirect API. We even added a plot_spectrum which can be used for directly plotting the absorbance spectra in the output cell.

We added additional metadata inside the .csv file of the absorbance spectrum data of the vial samples. This additional meta-data included the names and volumes of the indivisual chemicals inside the vial as well as the timestaps at which the readings were recorded.

We added code to implement a data management plan to store the recorded spectrum data files in an organised manner. In this data management plan, the spectrum data files will be stored in differing folders depending on the Operator name, experiment name, experiment date and the vial name in the follwing manner.

```
Spectrum_data
└─25-07-2025
  └─Aditya
    ├─expt1
    │ ├─experiement_metadata.json
    │ ├─mof_recipes.json
    │ ├─operations_log.txt
    │ ├─references
    │ │ ├─dark.npy
    │ │ └─white.npy
    │ └─spectra
    │   ├─A1.csv
    │   ├─A1_T0_absorbance_spectrum.png
    │   ├─A1_T10_absorbance_spectrum.png
    │   ├─A1_T20_absorbance_spectrum.png
    │   ├─A2.csv
    │   ├─A2_T0_absorbance_spectrum.png
    │   └─A2_T15_absorbance_spectrum.png
    └─expt2
      ├─experiement_metadata.json
      ├─mof_recipes.json
      ├─references
      │ ├─dark.npy
      │ └─white.npy
      ├─spectra
      │ ├─A1.csv
      │ ├─A1_T0_absorbance_spectrum.png
      │ └─A1_T10_absorbance_spectrum.png
      └─⋮
```

We added a wash probe functionality to the spectrometer. Whenever the wash probe function is instantiated, the spectrometer probe will go to the methanol beaker and dip the probe multiple times. Washing probe ensures no cross-contamination between the vials solutions.

4. Error Handling

For a complex, multi-step automated platform like Axolotl to operate reliably, a robust error-handling framework is not just beneficial, it is essential. Without intelligent checks and balances integrated into the software API, simple logical errors in an experimental script could lead to failed experiments, wasted reagents, and potential damage to the hardware. To mitigate these risks and ensure the integrity of autonomous operations, several critical error-handling conditions were implemented at the core of the Jubilee API. These checks function by maintaining a virtual state of the labware that is continuously updated, allowing the API to validate commands against this state before they are executed.

4.1. Lid-on-Top State Tracking

A common source of error in automated workflows is managing the state of labware lids. An attempt to pick up a lid from a vial that has no lid, or to dispense liquid into a vial that is still capped, would cause the entire process to fail. To counter this, we implemented a state-tracking mechanism within the software.

A boolean attribute called `lid_on_top` was added to the `Labware` class. This attribute serves as a digital twin for the physical state of the labware, and its value is automatically updated by the API whenever the vacuum gripper tool is used to either pick up or place a lid.

The critical component is the error-handling function, which explicitly checks if the current state of the `lid_on_top` attribute matches the expected condition for the requested action. If the check fails, the API raises an error and halts execution, preventing the erroneous physical action and providing a clear message to the user.

Pseudocode for Lid Status Check

```
FUNCTION check_lid_status(location, expt_cond.):
  labware = GET_LABWARE_AT(location)

  IF labware.has_lid_on_top IS EQUAL TO expt_cond.:
    PROCEED_WITH_ACTION
  ELSE:
    RAISE_ERROR("Lid status does not match expt_cond.")
  END IF
END FUNCTION
```

4.2. Liquid Volume and Level Tracking

To prevent the overflow or underflow of liquid from vials and beakers, it is crucial for the system to track the volume of liquid in each container. Dispensing more liquid than a vial can hold or attempting to aspirate more liquid than is available are common failure points in automated liquid handling.

To address this, we implemented an attribute called `currentLiquidVolume` within the `Well` class. This attribute acts as a real-time record of the liquid volume in every container on the deck. The API diligently updates this value after every single aspirate or dispense operation.

The corresponding error check is invoked before any liquid handling operation. It performs two primary checks: confirming that a target well has enough capacity before dispensing (Overflow Check) and verifying that a source well has sufficient liquid before aspirating (Underflow Check). If either condition is not met, a specific, informative error is raised.

Pseudocode for Liquid Volume Update and Check

```
PROCEDURE update_liquid_volume(well, volume, is_dispense):
  IF is_dispensing IS TRUE:
    well.current_volume += volume
  ELSE:
    well.current_volume -= volume
  END IF
END PROCEDURE
```

Pseudocode for Liquid Volume Check

```
FUNCTION check_liquid_level(well, tgt_vol, is_dispense):
  IF is_dispense IS TRUE:
    IF (well.curr_vol + tgt_vol) > well.total_cap:
      RAISE_ERROR("Overflow: well cannot acc vol.")
    END IF
  ELSE:
    IF well.curr_vol < tgt_vol:
      RAISE_ERROR("Underflow: Not enough liq in well.")
    END IF
  END IF
END FUNCTION
```

5. Experiment Orchestration

To transition from controlling individual tool actions to executing a complete, multi-step scientific experiment, a higher-level orchestration layer is required. This layer acts as the "brain" of the operation, translating a scientist's experimental plan into a precise sequence of robotic actions. In the Axolotl software architecture, this is handled by a dedicated `Experiment` class, which takes a declarative synthesis plan as input and manages the entire end-to-end workflow of vial synthesis and spectral data acquisition in a fully automated manner.

5.1. The Synthesis Plan: A Declarative JSON Approach

The foundation of any automated experiment is the `synthesis_plan.json` file. This file is a human-readable, structured document that defines all parameters of the experiment. Rather than writing a complex script for every new experiment, the user simply modifies this JSON file to describe the desired outcomes. This declarative approach separates the "what" (the experimental design) from the "how" (the robotic execution), making the system more flexible and accessible.

A typical `synthesis_plan.json` contains key information such as:

- Experiment Metadata: Details like the experiment name, date, and operator.
- Synthesis Recipes: A list of samples to be created, specifying which precursors and what volumes should be dispensed into each target vial.
- Spectrum record interval: The interval time gap between successive recordings on a sample vial.

This file serves as the single source of truth for the experiment, ensuring that every run is precisely documented and reproducible.

5.2. The `Experiment` Class and Orchestration Routine

The `Experiment` class is the core Python object that orchestrates the entire workflow. Upon initialization, it takes the path to the `synthesis_plan.json` as its primary argument. Its main responsibility is to parse this plan and manage a complex, time-sensitive sequence of operations to ensure both efficient synthesis and accurate, timely data collection.

The orchestration routine is not a simple linear execution of tasks. It is an intelligent, event-driven process designed to handle concurrent synthesis and characterization schedules. The workflow proceeds as follows:

1. **Initialization and Setup:** The `Experiment` object is created. It reads and validates the `synthesis_plan.json`. Based on the metadata, it automatically creates a structured directory on the file system to store all outputs from the run. This ensures that all data is saved in a consistent, organized manner from the very start.
2. **Pre-run Preparations:** Before any synthesis begins, the orchestrator performs critical setup tasks. First, it commands the spectrometer tool to collect reference readings (e.g., dark and white references) to ensure accurate absorbance measurements later. Following this, it directs the dual syringe tool to aspirate and fully prime the syringes with the necessary precursor and solvent solutions, ensuring they are ready for precise dispensing.
3. **Batch Precursor Dispensing:** To maximize efficiency, the routine begins by dispensing the common reagents in batches. The robot first dispenses the metal precursor into all vials specified in the synthesis plan, one by one. After completing this, it proceeds to dispense the solvent into the same set of vials. This batching method minimizes unnecessary tool changes and movements.

4. **Dynamic Synthesis and Characterization Loop:** This is the core of the orchestration logic. The system initiates a master loop that is not merely sequential but is governed by a scheduler that prioritizes time-sensitive tasks.
   - The orchestrator begins synthesizing the first vial in the plan by dispensing the final reagent (e.g., the organic precursor), which initiates the chemical reaction. The moment this is done, a timer is started specifically for this vial, and its second spectrum reading (T=10) is scheduled.
   - The system then consults its task queue. The queue contains two types of tasks: "synthesize the next vial" and "record spectrum for vial X at time Y".
   - The orchestrator continuously checks this queue and executes the task with the nearest deadline. For example, while it is in the process of synthesizing Vial 2, it constantly monitors the clock for Vial 1's next scheduled spectrum reading. If Vial 1's reading is due before the synthesis of Vial 2 is complete, the system will pause the synthesis task, switch to the spectrometer tool, perform the reading on Vial 1, and then resume the synthesis of Vial 2.
   - This dynamic, priority-based approach ensures that no scheduled data collection is ever missed. The system will always prioritize recording a spectrum for a previously synthesized vial over starting a new synthesis if there is a time conflict.
5. **Completion:** This intelligent loop continues until all vials have been synthesized and all scheduled spectrum readings for every vial have been collected. The experiment concludes, leaving a complete and organized record of the run, including the initial plan, metadata, and all time-resolved spectral data. This orchestration model is crucial for running kinetic studies where the timing of measurements is critical to the scientific outcome.

6. Results and Discussion

To test Axolotl platform, a single-vial experiment was designed and executed. The objective was to synthesize ZIF-67 and track its formation in real-time using the integrated spectrometer tool. The fully automated experiment ran for a total of 3 hours and 20 minutes, with spectral data collected at regular 15-minute intervals.

The time-resolved absorbance spectra collected during the automated synthesis are presented in Figure 12. The data clearly shows the evolution of the characteristic ZIF-67 absorbance profile over a period of 180 minutes.

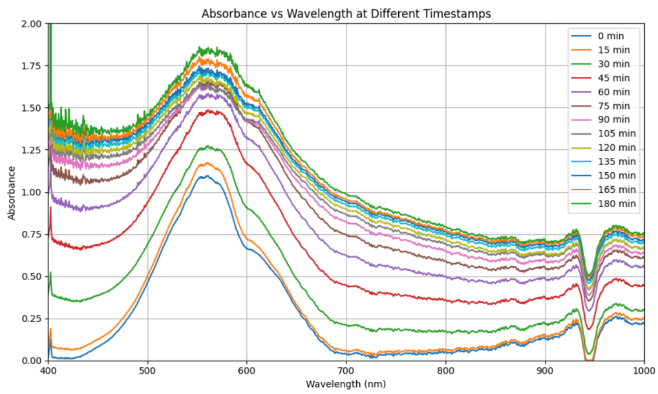

Figure 12. Absorbance spectra of a single ZIF-67 vial, synthesized and monitored automatically by the Axolotl platform. Spectra were recorded every 15 minutes for a total of 180 minutes.

7. Future Prospects

The successful development and validation of the Axolotl platform establishes a robust foundation for automated chemical synthesis. However, the ultimate vision extends beyond pre-programmed automation to the creation of a truly autonomous, self-driving laboratory capable of intelligent decision-making and accelerated discovery.

7.1. Bayesian Optimization

The next major phase of this project is to integrate an AI agent that can intelligently navigate the vast and complex parameter space of MOF synthesis. We will implement a Bayesian optimization (BO) algorithm to guide the experimental process. The BO agent will use the results from previous experiments to build a probabilistic model of the synthesis landscape, mapping input parameters (like precursor volumes and concentrations) to material outcomes.

7.2. Motion and Task Planning AI Agent

We also plan to integrate an AI agent to enhance the physical operation of the Axolotl setup. Current workflows rely on a fixed sequence of actions. A more advanced AI agent could be developed to perform dynamic motion planning, optimizing the robot's movements to minimize travel time and avoid potential collisions in more complex deck layouts.This would make the entire orchestration process more efficient and robust, especially in high-throughput screening scenarios with dozens of parallel experiments.

7.3. Early Stopping Conditions

To make the data collection process more efficient, we plan to implement an intelligent "early stopping" mechanism for the spectrometer readings. The current system collects data at fixed intervals for the entire duration of the experiment. The future system will analyze the kinetic data in real-time by fitting it to Gaultieri model and then stop the recordings after the stopping condition is staisfied for that vial.

8. Acknowledgements

I extend my sincere gratitude to Professor Shijing Sun and Mr. Zuyang Ye for their invaluable support and mentorship throughout this project. Their guidance, expertise, and encouragement have been instrumental in the successful completion of this work. I am deeply appreciative of their commitment to my academic and professional growth.

References

[1] The Science Jubilee Team, *Science Jubilee Documentation*. Read the Docs, 2025. [Online]. Available: https://science-jubilee.readthedocs.io/en/latest/getting_started/index.html

[2] The Jubilee Community, *Jubilee Wiki Main Page*. jubilee3d.com, 2025. [Online]. Available: https://jubilee3d.com/index.php?title=Main_Page

[3] UW Sun Lab, *Axolotl_Jubilee GitHub Repository*. GitHub, 2025. [Online]. Available: https://github.com/uwsunlab/Axolotl_Jubilee/tree/main