Setup Manual

Malhar Mahajan, Jack McGuire, Colleen Piccolo

May 2025

Contents

1 Introduction

This project began with a thrust stand from the previous year and involved major improvements to it, such as reducing the weight of the overall structure and integrating the ACSL flightstack onto its computer. This project is important in order to provide a self-contained testbed for different control algorithms.

1.1 Project Overview

This project aimed to integrate the components necessary to run the thrust stand in the ACSL lab to iteratively test controllers utilizing a Genetic Algorithm script (GA). This project successfully redesigned, reassembled, and rewired the thrust stand. This project also verified communication between the Odroid M1S and the MATLAB. This project consists of the following three main components: the physical thrust stand; the GA; and the code onboard the Odroid M1S (referred to throughout this manual as Odroid) on the drone, which can be found at the following Github repository: acsl-thrust-stand.

1.2 Purpose of This Manual

The purpose of this manual is to comprehensively walk the user through the steps necessary to set up the integrated system. The manual covers how to physically strap the drone to the thrust stand, how to power up the system, and how to work with the onboard code to collect and save data.

2 Drone and Thrust Stand Requirements

2.1 Drone Assembly and Hardware

The drone frame used in this project used the QAV400 FPV quadcopter frame with the G10 arms for the QAV500. The detailed drone components can be found in the Bill of Materials table in Section 11.1 of the appendix. The drone was designed such that the center of mass is as low as possible and centered at the Pixhawk 6C IMU's reference point.

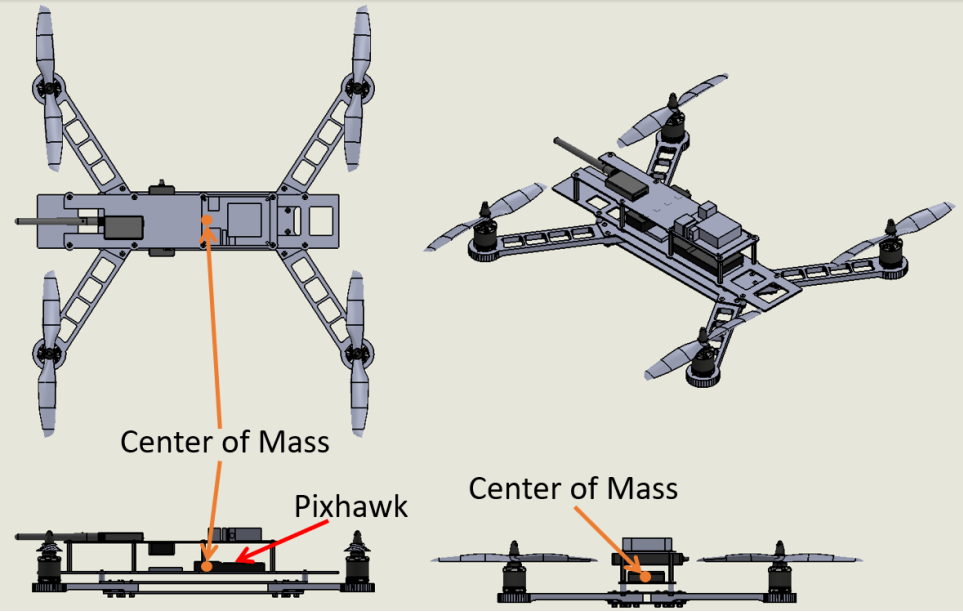

Figure 1: Drone CAD with center of mass annotated

2.2 Thrust Stand CAD Assembly

The thrust stand assembly can follow the same approach as outlined in the previous manual. However, the rings have additional cuts because their masses had to be heavily reduced for better motor performance.

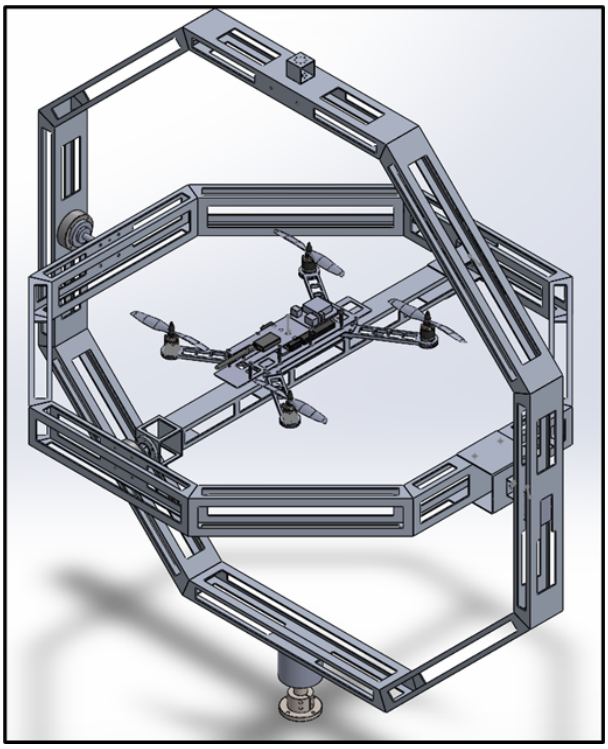

Figure 2: Thrust Stand CAD Assembly

3 Physical Setup of the Drone and Test Stand

3.1 Assembly of the Thrust Stand

The physical assembly of the thrust stand consists of fastening the three thrust stand rings together, and then fastening that structure to the outer frame of the thrust stand. First, the roll bar is fastened to the pitch ring. Then, the pitch ring is secured to the yaw ring. Finally, the yaw ring is attached to the outer structure. They should be done in this order to ensure that the axes of rotation for each degree of freedom intersect at the exact center of the thrust stand.

Figure 3: Thrust stand at the ACSL Lab with the drone on it

3.2 Securing Drone to Rig

The drones center of mass needs to be as closely aligned to the center point of the thrust stand as possible. The center point being the intersection of the 3 axes of rotation of the thrust stand. The drone must be secured to the roll bar so that the drone and roll bar act as a single rigid body. For this project the drone was secured to the roll bar of the thrust stand with two velcro straps. See the figure below for an example of attaching the drone to the thrust stand: **NOTE:** It is important to ensure that the Pixhawk on the assembled drone is mounted as close to the drone's center of mass as possible.

3.3 Power System Setup

The power system has two components: the drone battery and the lithium ion battery powering the Odroid. The lithium ion battery powering the Odroid needs to be charged

regularly, and it can be charged leveraging the power adaptor found on the Odroid's case and the USB-C wire.

3.4 Sensor and Telemetry Wiring

The schematic below shows how the electronics on the drone are assembled. The schematic includes the wiring of the thrust stand components as well as the drone components.

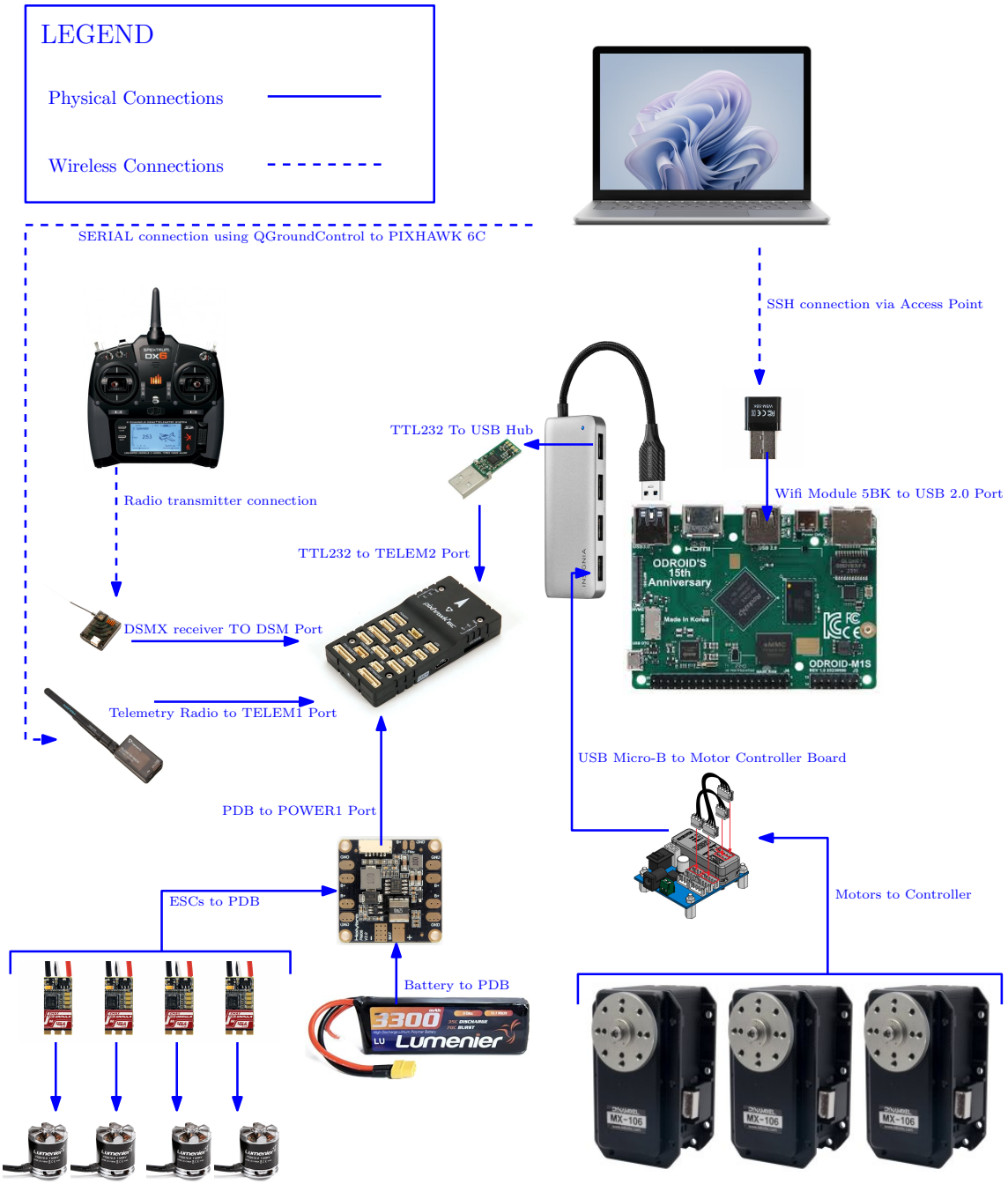

Figure 4: Thrust Stand Electrical Schematic

3.5 Odroid and Pixhawk Communication Setup

The Odroid and Pixhawk communicate over a high-speed MAVLink connection using a TELEM2-to-USB FTDI cable, enabling ROS2 nodes on the Odroid to send control commands and receive telemetry directly from the PX4 autopilot running on the Pixhawk.

3.6 Hardware Connections

- Connect the Pixhawk's `TELEM2` port to the Odroid using a **JST-GH to USB FTDI cable**.
- Ensure the wiring is correct (usually pre-wired in the FTDI cable):
  - Pixhawk TX → FTDI RX (USB side)
  - Pixhawk RX → FTDI TX (USB side)
  - Pixhawk GND → FTDI GND
- After connection, verify the Odroid detects the FTDI device as `/dev/ttyUSB0` (or similar) by running:

```
$ ls /dev/ttyUSB*
```

4 Software Requirements and Tools

4.1 On Host Computer

- PuTTY
- WinSCP
- MATLAB 2024a
  - Aerospace Toolbox
  - ROS Toolbox
- DYNAMIXEL Wizard 2.0

4.2 On Odroid

- Ubuntu 22.04 (X-bit)
- ROS2 Galactic
- PX4 Firmware and packages

4.3 Cloning and Building the Thrust–Stand Workspace

For full, up-to-date instructions on cloning, building, and cleaning the ACSL Thrust-Stand codebase (including all required submodules, C++ nodes, CAD models, and MATLAB scripts), please refer to the project's GitHub README:

```
https://github.com/andrealaffly/acsl-thrust-stand#readme
```

All workspace setup, dependency installation, and build steps are detailed there, along with one-click scripts to streamline the process.

5 Running the System

5.1 Powering On and System Checks

Before running the thrust stand system, follow these hardware and software preparation steps:

1. **Update Software:** Connect the Odroid to Ethernet and run:

   ```
   $ git pull
   ```

   in the project directory to ensure the latest code is synced.
2. **Prepare Drone and Hardware:**
   - Mount and secure the drone to the thrust stand using velcro straps.
   - Attach the Odroid to the drone.
   - Connect the blue Pixhawk TELEM2-to-USB FTDI cable to the Odroid.
   - Ensure the Odroid lithium-ion battery and the drone battery are both fully charged.
3. **Reboot Odroid:** Run:

   ```
   $ sudo reboot
   ```

   While the Odroid is connected to Ethernet. **Note:** The Odroid wifi will turn off when this command is executed and the host computer's wifi will need to be reconnected to it.
4. **Remote Controller:** Before turning on the drone, power on the remote controller (TX) to ensure proper TX-RX pairing (this enables the kill switch).

5.2 Launching the Flight Stack

Once the hardware setup is complete, follow these software launch steps:

1. **Clone the Git Repository** : Clone the Github Repository
2. **Build ROS2 Workspace (if code was updated):** Navigate to your workspace and rebuild:

   ```
   $ cd <path-to-repository>/acsl-thrust-stand
   $ source install/local_setup.bash
   $ source /opt/ros/galactic/setup.bash
   $ colcon build --packages-select drone_dynamixel_bridge
   ```

3. **Check Dynamixel Configuration:** Using Dynamixel Wizard 2.0, make sure all Dynamixel motors are:
   - Set to `current control mode`.
   - Torque-enabled on all three axes (roll, pitch, yaw).
   - Once finished, plug in the Dynamixel motor through the U2D2 power board to the Odroid USB hub.
4. **Start the micro-ROS Agent:** In a new terminal:

   ```
   $ MicroXRCEAgent serial --dev /dev/ttyUSB0 -b 921600
   ```

5. **Run the Dynamixel Bridge Node:** In another terminal:

```
$ cd <path-to-repository>/acsl-thrust-stand
$ source /opt/ros/galactic/setup.bash
$ source install/local_setup.bash
$ ros2 run drone_dynamixel_bridge drone_dynamixel_bridge
```

Note: This node computes and sends torque commands; allow some startup time.

6. **Launch the Flight Stack Node (Optional):** Only run this if you intend to execute the full control loop, including UAV rotor commands:

```
$ ros2 run flightstack flightstack
```

Warning: Running the flight stack may spin up the drone's rotors — ensure the system is secure.

7. **Run the Performance Index Node (for Mission Evaluation):** To evaluate control system performance over the mission:

```
$ ros2 run drone_dynamixel_bridge performance_index
```

5.3 Important Operational Notes

- The drone must be powered on and connected for odometry data, but rotors will not spin unless the flight stack node is running.
- The remote controller kill switch works independently but must be paired before powering on the drone.
- Always ensure the micro-ROS agent is running before launching bridge or flight stack nodes.
- After any code edit or pull, always rebuild with `colcon build` for that specific package (Flightstack or Drone Dynamixel Bridge) and re-source the environment.
- Ensure you connect the FTDI-USB that connects to the Pixhawk first before the one that connects to the Dynamixel motors to avoid having to change the USB number within the scripts.
- The performance index node will automatically shut down after the mission duration and save results to file.

6 MATLAB Genetic Algorithm

6.1 Genetic Algorithm Overview

The Genetic Algorithm script (GA), which is used to iteratively find the approximate best combination of Proportional, Integral, and Derivative gains, is powered by the MATLAB Global Optimization toolbox [2]. The purpose of the GA is to deliver the approximate best set of these gains at the end of the testing session. Communication between the Odroid and the GA has been verified. The MATLAB Global Optimization toolbox is used to calculate the next set of gains to test each iteration. The GA leverages this MATLAB toolbox to find the best set of possible gains. The GA uses the performance index to determine if the set of gains found for that iteration performed better or worse than the previous set.

There are several advantages to using a script leveraging a genetic algorithm to iterate through the possible sets of gains. In the GA, the ratio of exploration to exploitation can

be set in the `options.MutationFcn` field. The GA first performs a relatively broad search in a specific area looking for the area's minimum. This minimum indicates the particular set of gains that produces the lowest amount of error. Then, once the performance of those gains has been evaluated through simulation, the GA generates a new set of gains based on the performance metric generated from the previous iteration. After the algorithm has finished the first loop to get a set of gains, it enters a new loop to possibly find a better estimate of gains. The MATLAB Global Optimization toolbox allows customization for how many iterations it should search with to see if there is a better minimum error adjacent to the already identified error minimums. If using the simulation MATLAB, the performance metric is calculated in the `cost_fun_eval.m` file. When using the thrust stand, the `cost_fun_eval.m` file prompts the `bash_executor.m` file to run and get a file from the Odroid. The performance metric can be calculated in two locations: onboard the Odroid (if physically testing) and in the GA script (when simulating testing). The performance metric that's calculated onboard the Odroid is calculated using the L2Norm of the error between the expected and current trajectory. The last step of system integration needs to fully implement the passing of this file from the Odroid to the GA. Currently, there is verified communication between the Odroid and the GA through a test file. The GA calculates the performance metric by integration to get the error. The L2Norm of the error is then calculated.

6.2 On-board Performance–Index Node

The node `performance_index_node` runs on the Odroid as part of the ROS 2 stack and evaluates the continuous cost in real time:

- It subscribes to the actual attitude $\mathbf{q}(t) = \begin{bmatrix} w, x, y, z \end{bmatrix}^{\mathsf{T}}$ published by PX4 on `/fmu/out/vehicle_attitude` and converts the quaternion to roll–pitch–yaw $\eta(t) \in \mathbb{R}^3$.
- Simultaneously it listens to the desired set-point quaternion $\mathbf{q}_d(t)$ on `/fmu/in/vehicle_attitude_setpoint` and converts it to $\eta_d(t)$.
- Once both streams are available it time-integrates the squared Euclidean error $\|e(t)\|_2^2 = \|\eta_d(t) - \eta(t)\|_2^2$ with a 100 ms sample period, using

$$J(T) = \int_0^T \|e(t)\|_2^2 \, \mathrm{d}t \approx \sum_{k=1}^{N} \|e[k]\|_2^2 \, T_s,$$

  where $T_s = 0.1$ s.
- After the mission horizon T elapses it publishes both to the console and to a file `performance_index.txt`, which the MATLAB–GA script fetches.

6.3 Features of the Genetic Algorithm

The GA has two separate "modes" that can be used depending on the desired use case. These are stored in two different .m files. `main_GA_PID_rot_dynamics.m`, found in the `Physical` folder, executes the GA functions designed to be compatible with the thrust stand. `Main_GA_PID_rot_dynamics.m`, found in the `Simulation` folder, executes the GA with simulated results.

6.4 Using the Genetic Algorithm

The GA needs to be launched from the host computer. If running the physical thrust stand, it is run from the `main_rot_dynamics.m` file. If simulating, it is run from the `Main_GA_PID_rot_dynamics.m` file. Keep in mind that all .m files referenced in the main file need to be in the same folder so that they can all be in the same MATLAB directory. Once the MATLAB directory is setup, click "Run" on the `main_rot_dynamics.m` file. The connection between the GA and the Odroid on the thrust stand has been verified. The next step for this functionality is to implement it in testing.

Figure 5 below shows a sample output of the "Main_GA_PID_rot_dynamics.m" file after simulation:

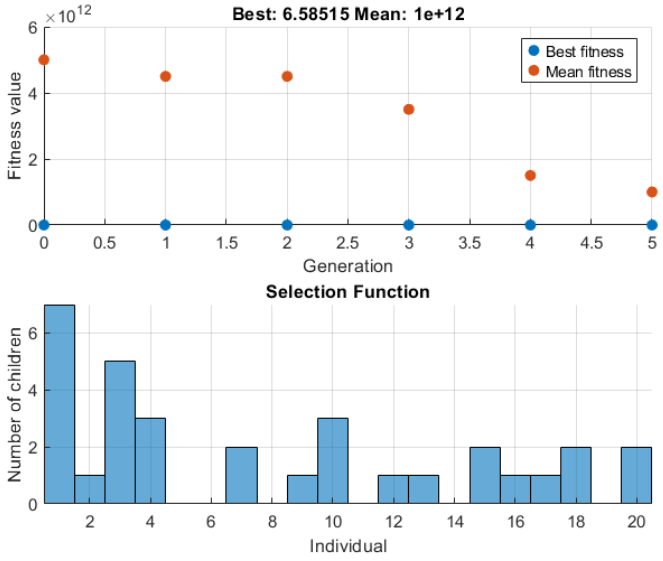

Figure 5: Output of MATLAB Simulation

Detailed instructions on how to set up the GA for either use case can be found in the README.md in the MATLAB folder at acsl-thrust-stand.

7 Data Logging and Analysis

7.1 Flight Stack Data Logging

The `drone_dynamixel_bridge` node automatically records critical system states and control signals during runtime into a time-stamped CSV file. This log includes:

- Timestamp (ROS2 time in seconds)
- Euler angles: roll (ϕ), pitch (θ), yaw (ψ)
- Angular velocities: p, q, r
- Angular accelerations (derived): $\dot{p}$, $\dot{q}$, $\dot{r}$
- Total body torque estimates (T_x, T_y, T_z)
- Gimbal torque commands
- Raw current commands sent to Dynamixel motors

Logs are written to a CSV file named using the system date-time format: `diag_YYYYMMDD_HHMMSS.csv`.

7.2 Data Export

The log files can be accessed directly from the Odroid filesystem. For export:

1. Connect to the Odroid using `scp` or `WinSCP`.
2. Navigate to the workspace directory where the CSV files are stored.
3. Transfer the desired `diag_*.csv` file to your local machine for post-processing.

These logs can be analyzed using MATLAB, Python (with `pandas`), or Excel.

7.3 Best Practices for Data Handling

- **Organize Logs:** Maintain a structured folder system to separate logs by date, mission, and experimental condition.
- **Back Up Regularly:** Copy logs from the Odroid to an external or cloud drive after each test session.
- **Synchronize Clocks:** Ensure ROS2 timestamps are consistent across sessions by regularly synchronizing the Odroid system clock.
- **Document Context:** Keep a manual or digital record describing what each log run represents (controller version, hardware setup, mission profile).
- **Validate Logs:** After each run, quickly inspect the CSV file for completeness to catch any data drops or logging failures.

The system's CSV logs provide a detailed trace of both control commands and physical responses, enabling fine-grained performance analysis and debugging of adaptive control algorithms.

8 Troubleshooting Guide

8.1 Common Issues

- **Odroid fails to connect to Pixhawk:** Ensure the Pixhawk TELEM2-to-USB FTDI cable is securely connected and appears as `/dev/ttyUSB0`. Verify that the MicroXRCEAgent is running and configured with the correct serial port and baud rate.
- **Drone does not send odometry data:** Confirm that the drone is powered on, and the MicroXRCEAgent is running. If odometry data still does not appear, reboot the Odroid, ensure the drone is properly connected, and check topic activity using `ros2 topic echo /fmu/out/vehicle_odometry`.
- **Kill switch not working:** Make sure the remote controller is powered on *before* powering on the drone to ensure TX-RX pairing. Confirm the controller is in range and functional.
- **Dynamixel motors do not respond:** Ensure all motors are set to `current control mode` and torque-enabled using Dynamixel Wizard 2.0. Check USB connections to the Odroid and verify that `drone_dynamixel_bridge` is running without errors.

8.2 ROS2 Node and Topic Debugging

- **Missing or corrupted `px4_msgs` package:**
  Make sure `px4_msgs` is correctly added to your workspace or `.gitmodules`. If it is missing, clone and build it manually:

```
$ cd <path-to-repository>/acsl-thrust-stand/src
$ git clone https://github.com/PX4/px4_msgs.git
$ cd <path-to-repository>/acsl-thrust-stand
$ source /opt/ros/galactic/setup.bash
$ source install/local_setup.bash
$ colcon build --packages-select px4_msgs # build just the msg pkg
```

- **Checking active topics:**

```
$ ros2 topic list
$ ros2 topic echo /fmu/out/vehicle_odometry
```

- **Verifying node status:**

```
$ ros2 node list
$ ros2 topic info /fmu/out/vehicle_odometry
```

- **Relevant repositories (clone into `src/` if missing OR if you run into build errors.):**
  - ACSL flightstack
  - px4_msgs
  - Dynamixel SDK
- **Building packages individually when errors appear:**
  If a full workspace build fails, isolate the faulty package:

```
# Example: build just the ACSL flightstack
$ colcon build --packages-select acsl_flightstack

# Then build the Dynamixel SDK
$ colcon build --packages-select dynamixel_sdk

# Finally rebuild everything once dependencies compile cleanly
$ colcon build
```

8.3 Odroid Boot and WiFi Problems

- **Odroid fails to connect to Wi-Fi:** Confirm Wi-Fi credentials are properly configured and the Odroid Wi-Fi adapter is functioning. Check if the hotspot or local network is stable.
- **Odroid fails to boot properly:** Ensure the lithium-ion battery is fully charged or the Odroid is connected to the charger. Monitor the boot process via HDMI to diagnose boot failures.

8.4 Flight Stack Build Failures

- **Workspace build fails:** Verify all dependencies (`px4_msgs`, `flightstack`, `dynamixel_sdk`) are present in `src`. Run:

```
$ colcon build
$ source install/local_setup.bash
```

- **Out-of-date repositories:** Always run `git pull` in `<path-to-repo>/acsl-thrust-stand/src` for each package before rebuilding.

8.5 Hardware Failures and Solutions

- **Dynamixel motor torque failures:** Use Dynamixel Wizard 2.0 to test each motor independently. Confirm correct mode and torque are set. Check for mechanical jams or misalignments.
- **Drone or thrust stand wiring issues:** Double-check all power and telemetry wiring, especially the slip rings and sensor connections. Ensure no frayed wires or loose connections.
- **Emergency shutdown:** The physical kill switch via remote controller is always active regardless of QGroundControl status, provided the TX was powered on before the drone.

9 Future Work and Recommendations

The current system attempts to cancel MOI on the thrust-stand, but several improvements have been identified during testing. This section captures the key open issues and proposes concrete next steps.

9.1 Filter and Estimator Refinement

- **Initial acceleration spikes.** The `FilterDiff` differentiator shows large transients at $t = 0$–0.1s. Recommended mitigations:
- Increase the damping ratio ζ (e.g. from 0.7 to 0.9) or lower ω_n by 20–30%;
- Possibly bypass Pixhawk gyroscope rates for the very first samples and initialise with *direct* Dynamixel encoder velocities.
- **Velocity source fusion.** Compare two pipelines:
  (a) PX4 gyro rates $\rightarrow$ filter-diff;
  (b) Dynamixel encoder rates $\rightarrow$ filter-diff;

9.2 Hardware and System Enhancements

- **Thrust-Stand Redesign for Low MOI.** Fabricate a more compact frame using carbon-fiber tubes and 7075-T6 aluminium joints, aiming for a 50 to 60 percent reduction in total stand inertia. A smaller polar MOI reduces the cancellation torque requirement, allowing the existing MX-106T actuators to remain within their allowable current range.
- **EMI refinement.** Look into possible issues with Electromagnetic Interference with the
- **On-board IMU upgrade.** Mount a BMI088 or ADIS16470 on the stand frame and fuse its output with Pixhawk gyros for better observability of stand dynamics.

10 Acknowledgements

The team would like to thank the following individuals for their contributions to this project.

Dr. Andrea L'Afflitto:

- Weekly team meetings and providing guidance for project development.
- Modeling the equations of motion of the thrust stand.
- Designing the controllers for the thrust stand.

Graduate Students of ACSL:

- Luca Nanu (CAD, torque testing, control algorithm development, "motor_characteristic.m" and "figure_plot.m" files)
- Giri Mugundan Kumar (Software and schematic)
- Mattia Gramuglia (Software)
- Giorgio Orlando (Control algorithm development)

11 Appendix

11.1 Bill of Materials (Detailed)

Bill of Materials for Thrust Stand

Quantity	Item	Price per unit	Cost	Hyperlink
1	Gearset	127.70	127.70	Dynamixel
2	Slip Rings	55.84	111.68	Amazon2
1	Aluminum Rods	14.99	14.99	Amazon3
2	Flange-Mounted Shaft Support for 1/2" Shaft, 2024 Aluminum	92.81	185.62	McMasterCarr1
1	6061 Aluminum Rectangular Tube, 1/8" Wall, $2\frac{1}{4}$"×$2\frac{1}{4}$"×$\frac{1}{4}$ ft	15.91	15.91	McMasterCarr2
1	6061 Aluminum Rectangular Tube, 1/16" Wall, $1\frac{1}{2}$"×$1\frac{1}{2}$"×1 ft	5.59	5.59	McMasterCarr3
2	Silver Corner Bracket, 1" Long for 1" Rail	7.92	15.84	McMasterCarr3
4	Silver Corner Bracket, $1\frac{1}{2}$" Long for $1\frac{1}{2}$" Rail	8.80	35.20	McMasterCarr4
1	Black-Oxide Alloy Steel Socket Head Screw 8-32×5/8"	14.89	14.89	McMasterCarr5
1	Zinc-Plated Steel Hex Nut 8-32	2.02	2.02	McMasterCarr7
2	Set-Screw Shaft Collar for 1/8" Shaft, 6061 Aluminum	41.11	82.22	McMasterCarr8
2	Rust-Oleum Stops Rust Flat Black Spray Paint (12 oz)	6.98	13.96	Lowes1
2	2X Ultra Cover Flat Black Paint & Primer (12 oz)	6.48	12.96	Lowes2
1	18-8 SS Hex Screw 5/16"-18×3/4"	8.39	8.39	McMasterCarr9
1	18-8 SS Hex Head Screw 5/16"-18×7/8"	4.48	4.48	McMasterCarr10
1	Stranded Wire 300 V AC, 22 AWG, Black	6.70	6.70	McMasterCarr11
1	Stranded Wire 300 V AC, 22 AWG, Red	6.70	6.70	McMasterCarr12

New Drone Components

Quantity	Item	Price per unit	Cost	Hyperlink
4	EMAX Formula 45A BLHeli32 ESC	34.99	139.96	ESCs
1	XT30 Male/Female Connector Pair w/ 10 mm 16 AWG Wire	9.99	9.99	Amazon
1	Insignia 4-Port USB 3.0 Hub, Gray	14.99	14.99	Best Buy
1	Spektrum DSMX 2.4 GHz Receiver SPM9745	45.35	45.35	ATModels

Legacy Drone and Thrust Stand Components (Original BOM)

Qty	Item	Hyperlink
1	ODROID-XU4 SBC	ameridroid
1	5 V / 4 A Power Supply	ameridroid
1	64 GB eMMC 5.1 Module (Blue Dot)	ameridroid
1	USB 3.0 eMMC Module Writer v2	ameridroid
1	DC Plug & Cable Assembly 5.5 mm *L*-type	ameridroid
1	QAV400 "Dirty-Frame" Plate	getFPV
1	QAV400 Power-Distribution Board	getFPV
1	QAV400 Flight-Controller Cover Plate	getFPV
1	QAV400 "Clean-Frame" Base Plate	getFPV
1	APC Prop-Adapter Ring Set	getFPV
1	APC 8×4.5MRP Propeller (CW)	getFPV
4	APC 8×4.5MR Propeller (CCW)	getFPV
6	FX2216-9 1100 kV V2 Motors	getFPV

6	30 A BLHeli-S ESC (2–4 S, DSHOT)	getFPV
2	Lumenier 3300 mAh 4 S 35 C Li-Po Battery	getFPV
6	G10 Arm for QAV500	getFPV
1	Pixhawk 6C & PM02 V3 Power Module	getFPV
1	150 mm Male–Female JR Servo Lead	Amazon
1	FS-i6 RC Transmitter & iA6B Receiver	Amazon
1	Rankie HDMI Cable (2-pack, 6 ft)	Amazon
1	Wired Keyboard & Mouse Bundle	Amazon
2	USB-to-TTL 3.3 V FTDI Cable	Amazon
1	Sabrent 4-Port USB Hub	Amazon
8	M3 Rubber Vibration Isolators 8 × 8 mm	Amazon
1	SiK Telemetry Radio V3	Holybro
1	USB Wi-Fi Adapter for ODROID-XU4	Walmart
1	M3 × 6 mm Socket-Head Screws (100 pk)	McMaster-Carr
1	M3 × 8 mm Socket-Head Screws (100 pk)	McMaster-Carr
1	M3 × 10 mm Socket-Head Screws (100 pk)	McMaster-Carr
1	M3 Female Standoff 35 mm	McMaster-Carr
1	M3 Steel Hex Nuts (100 pk)	McMaster-Carr
3	Low-Carbon Steel Rod ∅1” × 1 ft	McMaster-Carr
2	6061 Rectangular Tube 1 3/4” × 3 1/2” × 6 ft	McMaster-Carr
1	1.5” × 1.5” T-Slot Extrusion (8 ft)	McMaster-Carr

1	1.5” × 1.5” T-Slot Extrusion (10 ft)	McMaster-Carr
1	Easy-Access Flange-Mounted Shaft Support	McMaster-Carr
16	1.5” Corner Bracket (for 1.5” rail)	McMaster-Carr
16	3” Corner Bracket (for 1.5” rail)	McMaster-Carr
3	Dynamixel MX-106T Smart-Servo	Robotis
1	XT60 Li-Po Pigtail 12 AWG (5-pack)	getFPV
3	12-Circuit Through-Bore Slip Ring	ATO
1	Energizer AA Batteries (12-pack)	Amazon
2	12 V 10 A DC Power Supply (120 W)	Amazon
2	Pololu 5 V 5 A Step-Down Regulator	getFPV
1	Dynamixel Starter Set	Robotis
2	MX-106 Gear / Bearing Set	Robotis

References

[1] A. L’Afflitto, “ACSL Flightstack,” GitHub repository, 2024. [Online]. Available: `https://github.com/andrealaffly/ACSL-flightstack/tree/main`

[2] MathWorks, “Global Optimization Toolbox,” 2024. [Online]. Available: `https://www.mathworks.com/products/global-optimization.html`

[3] A. L’Afflitto, *A Mathematical Perspective on Flight Dynamics and Control*. London, UK: Springer, 2017.

[4] M. Gramuglia, G. M. Kumar, and A. L’Afflitto, “A Hybrid Model Reference Adaptive Control System for Multi-Rotor Unmanned Aerial Vehicles,” in *AIAA SciTech*, Orlando, FL, Jan. 2024.

[5] M. Gramuglia, G. M. Kumar, and A. L’Afflitto, “Two-Layer Adaptive Funnel MRAC with Applications to the Control of Multi-Rotor UAVs,” in *Int. Workshop on Robot Motion and Control*, Poznan, Poland, July 2024.

[6] M. Gramuglia, "Design and high-fidelity simulations of an adaptive control system for X8-copters employed for payload delivery," M.S. thesis, Politecnico di Milano, Milan, Italy, 2023.

[7] M. Gramuglia, G. M. Kumar, G. A. Orlando, and A. L'Afflitto, "An Open-Source Framework to Design, Tune, and Fly Nonlinear Control Systems for Autonomous UAVs," invited paper, *AIAA SciTech*, Orlando, FL, Jan. 2025.