

Rapid Adaptation Rover

Learned Dynamics Compensation for a Differential Drive Robot

Design Document

Team 39

Devansh Agarwal — da30

Srish Vijay — svija4

Bhavya Agrawal — bhavyaa2

ECE 445: Senior Design

Fall 2026

University of Illinois Urbana-Champaign

October 2026

Professor: Viktor Gruev

Teaching Assistant: Aniket Chatterjee

Rapid Adaptation Rover	1
Learned Dynamics Compensation for a Differential Drive Robot	1
Design Document	1
1 Introduction	4
1.1 Problem and Solution	4
1.2 Visual Aid	4
2 Design	7
2.1 System Block Diagram	7
2.2 Physical Design	8
2.3 S1 Mechanical Platform and Actuation	8
2.3.1 Mechanical Layout	9
2.3.2 Drivetrain Design	10
2.3.3 Mechanical Stability and Interfaces	11
2.3.4 Requirements and Verification	11
2.4 S2 Real Time Control and Communication	12
2.4.1 Motor Command Execution	12
2.4.2 STM32-Raspberry Pi Communication	12
2.4.3 Command Freshness and Failure Handling	13
2.4.4 Requirements and Verification	13
2.5 S3 Proprioceptive Sensing	14
2.5.1 Rear Wheel-Speed Sensing	15
2.5.2 Motor Current Sensing	15
2.5.3 IMU Sensing	16
2.5.4 Battery Voltage Sensing	16
2.5.5 Requirements and Verification	16
2.6 S4 Power and Safety	17
2.6.1 Power Distribution	18
2.6.2 Independent Hardware Safety	19
2.6.3 Power and Safety Interfaces	19
2.6.4 Requirements and Verification	20
2.7 S5 Evaluation Sensing	21
2.7.1 Floor-Motion Reference	21
2.7.2 Drag-Force Measurement	22
2.7.3 Payload, Incline, and Yaw Measurements	22
2.7.4 Evaluation Data Isolation	23
2.7.5 Requirements and Verification	23
2.8 S6 Simulation and Learned Control	24
2.8.1 Simulation and Training	25
2.8.2 History-Based Adaptation	26
2.8.3 Raspberry Pi Inference	26
2.8.4 Frozen Model Configuration	26
2.8.5 Requirements and Verification	27
2.9 S7 Baselines and Evaluation	28
2.9.1 Controller Baselines	29

2.9.2 Controlled Test Conditions	29
2.9.3 Motion-Tracking Metrics	30
2.9.4 Adaptive-Controller Improvement	30
2.9.5 Drag-Recovery Measurement	31
2.9.6 Data Retention and Experimental Validity	31
2.9.7 Requirements and Verification	31
2.10 Tolerance Analysis	33
2.10.1 Critical Design Parameter	33
2.10.2 Mathematical Analysis	33
2.10.3 Expected Worst-Case Performance	35
3 Cost and Schedule	35
3.1 Cost Analysis	36
3.1.1 Labor	36
3.1.2 Parts and Materials	36
3.1.3 Total Project Cost	36
3.2 Schedule	36
3.2.1 Weekly Project Schedule and Team Responsibility Distribution Table	36
4 Discussion of Societal Impact, Engineering Standards, Ethics, and Safety Considerations	36
4.1 Societal Impact	36
4.2 Engineering Standards	36
4.3 Ethics	36
4.4 Electrical and Mechanical Safety	36
4.5 Safety Mitigation Procedures	36
5 References	36

1 Introduction

1.1 Problem and Solution

A differential-drive rover can respond differently to the same motor commands when its operating conditions change. Added payload changes the mass that must be accelerated, an incline introduces additional gravitational load, and mechanical resistance increases the effort required from the motors. These changes can affect acceleration, speed, and turning performance, particularly during transient motion, even when a fixed-gain controller can eventually restore the desired steady-state speed. This problem is relevant to small indoor transport robots, which may experience changing loads and travel conditions while still requiring consistent and predictable motion. Our project will investigate whether recent onboard measurements can provide useful information about these changing dynamics and improve motion tracking.

To address this problem, we will build a four-wheel-supported rover with two driven wheels, two passive casters, a custom STM32 control and sensing PCB, and a Raspberry Pi 4 [12]. A compact network, inspired by rapid motor adaptation [1], will infer a representation of the rover's current dynamics using a short history of onboard sensor measurements and applied motor commands. The rover's IMU will measure yaw rate, while a downward-facing optical-flow sensor will provide an evaluation-only reference for surface motion. The same rover will be tested using tuned conventional control, a learned policy with a fixed dynamics representation, and a learned policy with history-based adaptation. The expected improvement of our approach is better short-duration speed and turning tracking when payload, incline, or mechanical resistance changes, without requiring the controller to be manually retuned for each condition.

1.2 Visual Aid

The Rapid Adaptation Rover is intended to evaluate how well a learned controller can maintain motion tracking when the rover's physical operating conditions change. The figure below shows the three primary disturbance scenarios used during testing: payload, incline, and external drag.

In the payload trial, a known mass equal to 25% or 50% of the unloaded rover mass is weighed before installation and secured in the payload tray near the center of the chassis. The rover then follows the same commanded motion profile so that the effect of increased mass can be measured. In the incline trial, the rover travels uphill on a repeatable ramp with a nominal angle of approximately 5° , allowing the controller to be evaluated under increased gravitational loading. In the drag trial, an external tether applies approximately 0 to 1 N of measured opposing force near axle height while the rover is moving on a level surface. The drag force is measured independently so that the onset and magnitude of the disturbance can be identified during analysis.

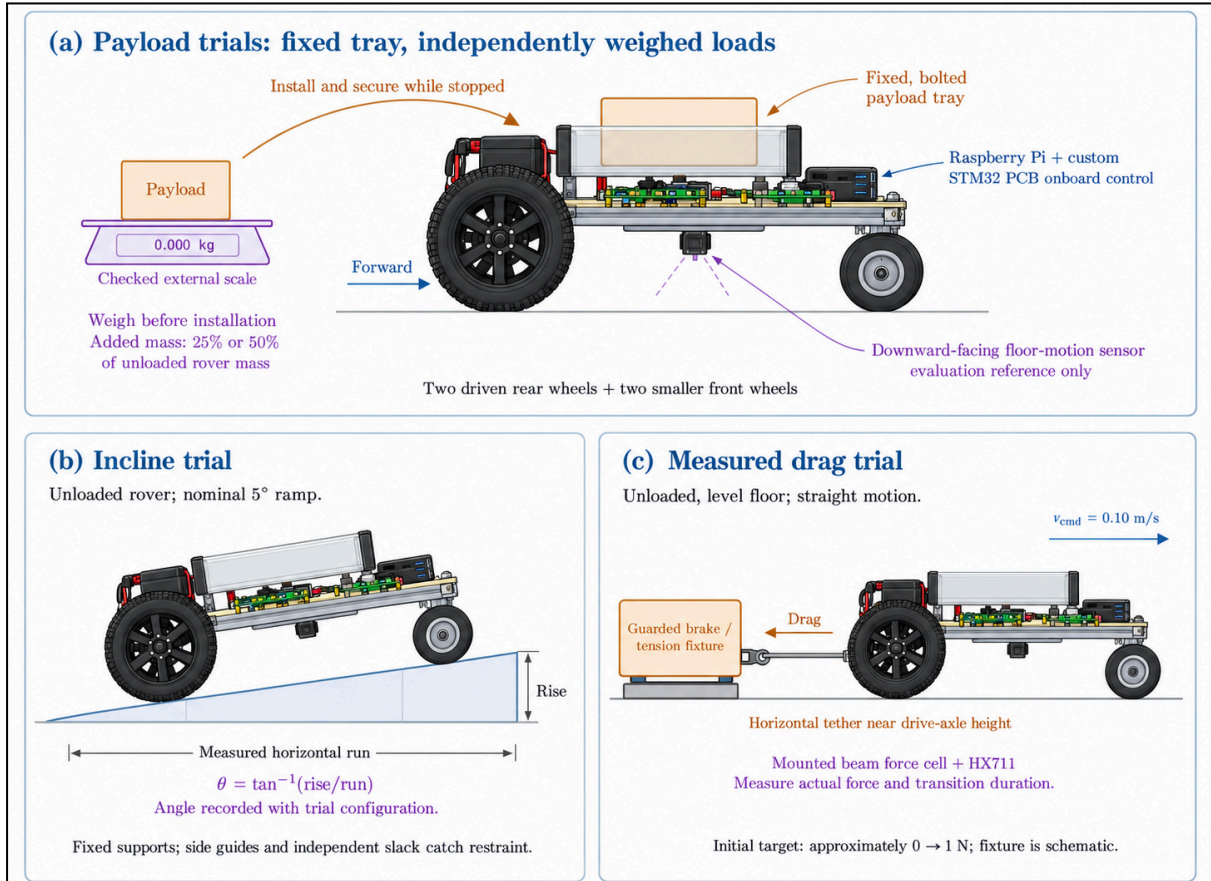
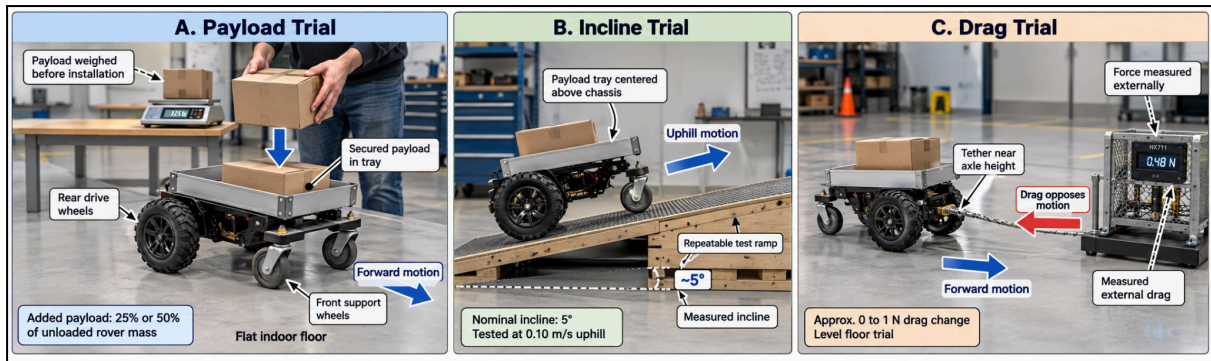


Figure 1: Visual Aid Figure Showing the Trials

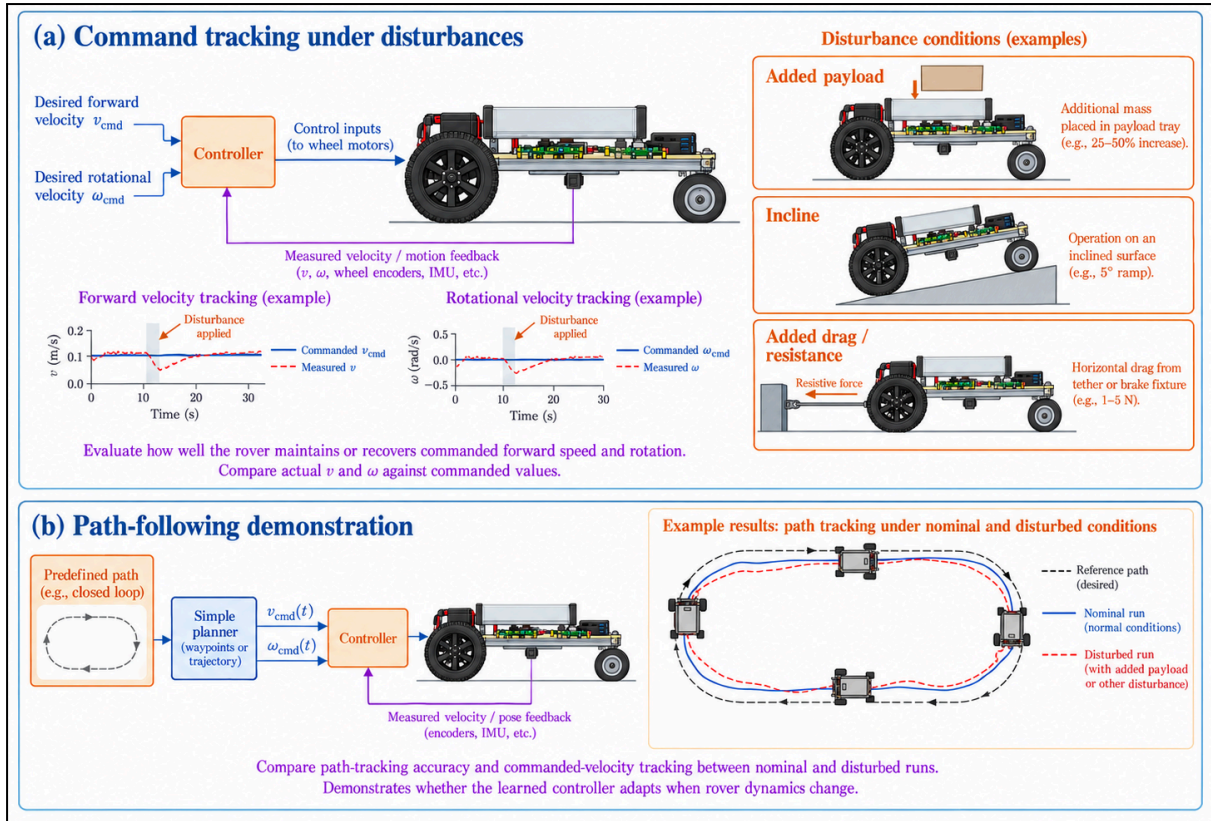


Figure 2: Rover Control Performance and Disturbance Evaluation Framework

1.3 High-Level Requirements

- **The rover shall accurately track commanded forward speed and turning rate under changes in payload, incline, and mechanical resistance.** For the defined test conditions, the rover shall achieve a forward-speed RMSE of no more than 0.02 m/s and a yaw-rate RMSE of no more than 0.10 rad/s in at least four of five valid trials.
- **The history-adaptive controller shall demonstrate improved motion tracking compared with the fixed learned controller under changing dynamics.** With identical test conditions and frozen network weights, the adaptive policy shall reduce mean forward-speed RMSE by at least 10% relative to the fixed-latent policy in the predefined primary disturbance condition.
- **The rover shall respond to changes in its dynamics within a defined recovery period.** Following a measured in-motion change in mechanical resistance, the adaptive controller shall return forward-speed and yaw-rate errors to their specified acceptable ranges within 1.5 seconds and maintain those ranges for at least 0.5 seconds in at least four of five valid trials.
- **The complete rover control system shall operate in real time and provide a safe method of disabling the motors.** The rover shall maintain a 50 Hz onboard policy loop for at least ten minutes without missing a 20 ms control deadline while logging telemetry at 100 Hz, disable both motor drivers within 101 ms after loss of valid commands, and include a hardware emergency stop capable of disabling the drive independently of Raspberry Pi software.

2 Design

2.1 System Block Diagram

The Rapid Adaptation Rover is divided into seven modular subsystems that separate the mechanical platform, real-time control, onboard sensing, power and safety, evaluation sensing, learned control, and experimental comparison functions. The Mechanical Platform and Actuation subsystem (S1) provides the four-wheel rover structure and rear differential-drive actuation. The Real-Time Control and Communication subsystem (S2) uses the STM32 to generate motor commands, enforce timing and command limits, and exchange telemetry with the Raspberry Pi. The Proprioceptive Sensing subsystem (S3) measures rear wheel speed, motor current, IMU acceleration and angular velocity, and battery voltage for use by the controller. The Power and Safety subsystem (S4) distributes battery power, generates the required voltage rails, and provides independent drive-disable mechanisms. The Evaluation Sensing subsystem (S5) measures quantities such as ground-reference velocity and externally applied drag that are used only for testing and are excluded from the learned-controller inputs. The Simulation and Learned Control subsystem (S6) runs on the Raspberry Pi and produces motor-duty commands using recent onboard measurement history. Finally, the Baselines and Evaluation subsystem (S7) compares the adaptive controller against the fixed-latent learned controller and conventional PI baseline using common test conditions and performance metrics. Together, these subsystems form a modular architecture in which sensing and control data flow between the STM32 and Raspberry Pi while evaluation-only signals remain isolated from the deployed controller.

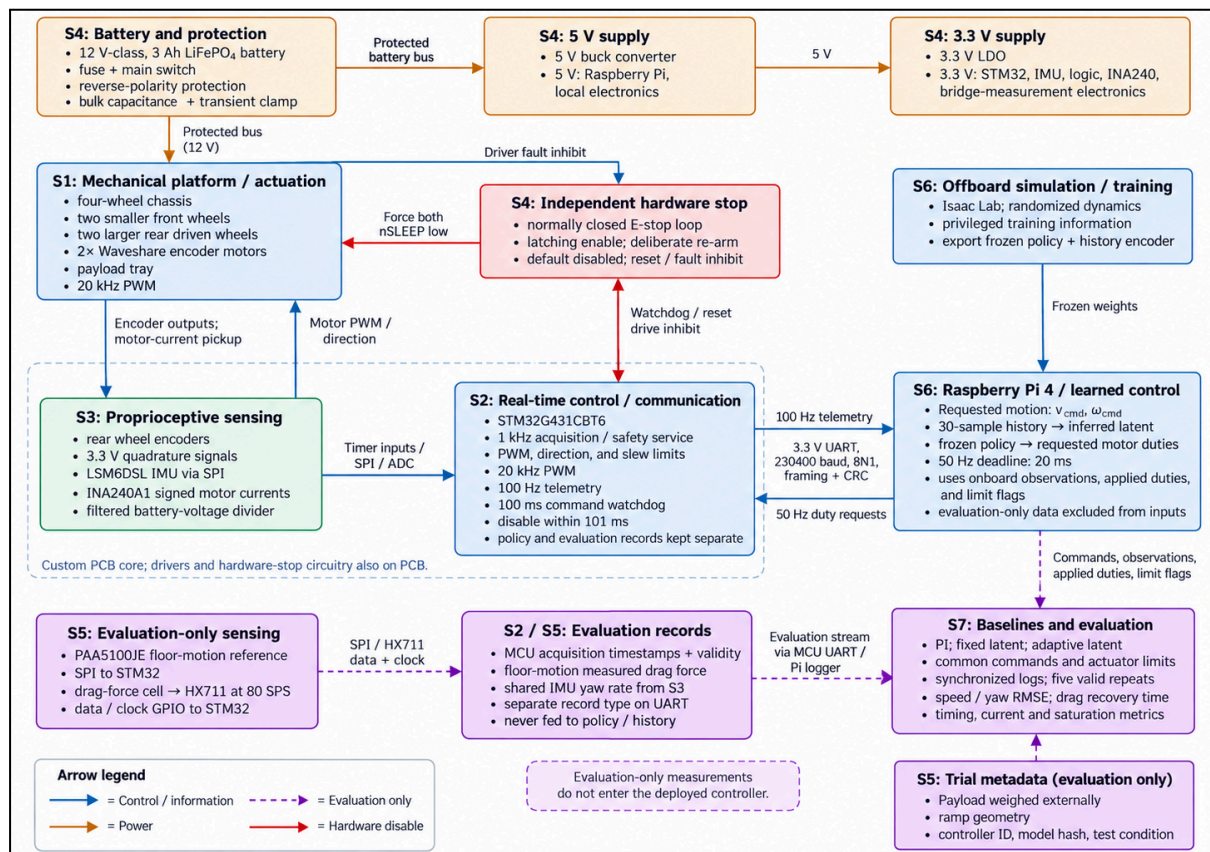


Figure 3: Overall System Architecture and Subsystem Interfaces

2.2 Physical Design

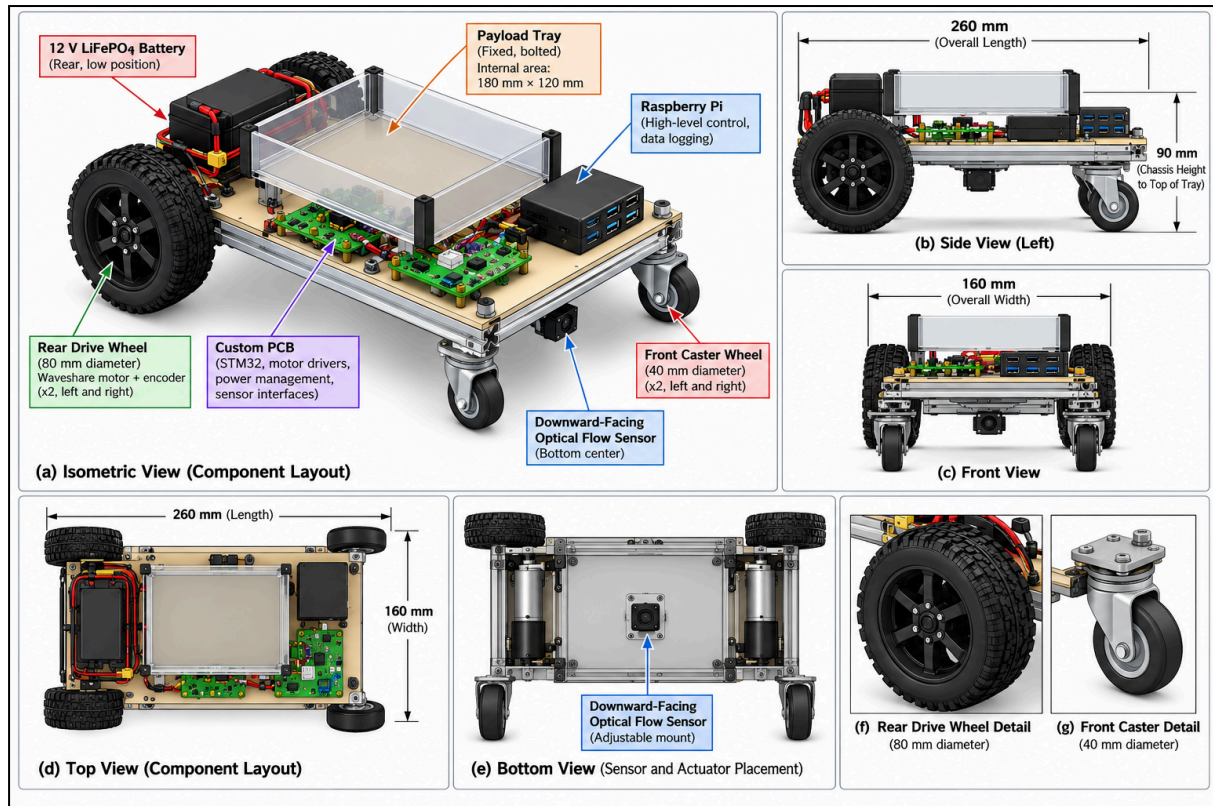


Figure 4: Rover Physical Design and Component Layout

The physical design of the rover is organized around a compact 260 mm × 160 mm chassis with a maximum height of approximately 90 mm to the top of the payload tray. Two 80 mm rear drive wheels provide differential-drive propulsion and are connected to encoder-equipped motors for measuring wheel motion, while two smaller 40 mm front caster wheels support the chassis and allow free turning. A fixed payload tray is mounted near the center of the rover so that added mass remains close to the vehicle's center of gravity. The battery is mounted low on the rear portion of the chassis to improve stability, while the Raspberry Pi and custom STM32 control PCB are positioned on the upper chassis for accessible computation, motor control, power management, and sensor interfacing. A downward-facing optical-flow sensor is mounted near the center of the underside of the rover to measure motion relative to the floor and provide an independent reference for evaluating velocity tracking. This component placement keeps the sensing, actuation, power, and computation hardware mechanically integrated while leaving the central tray accessible for controlled payload experiments.

2.3 S1 Mechanical Platform and Actuation

The Mechanical Platform and Actuation subsystem provides the physical structure and drivetrain required for the rover to execute commanded forward and turning motions while carrying the payloads used during evaluation. The rover uses a four wheel configuration

consisting of two smaller front wheels and two larger rear wheels. The rear wheels form the differential drive pair and are independently driven by two geared DC motors. The smaller front wheels provide additional chassis support while allowing the rover to maintain stable ground contact during straight line motion, turning, payload testing, incline traversal, and drag testing.

This subsystem contributes primarily to High-Level Requirement 1 (HL1) because the rover must physically produce repeatable forward and rotational motion under nominal and disturbed operating conditions. The mechanical platform must also maintain sufficient traction and stability so that measured tracking errors represent controller behavior rather than mechanical instability, loss of wheel contact, or payload movement.

The current drivetrain uses two Waveshare DCGM-3865-12V-EN-240RPM geared motors for the left and right rear drive wheels [6]. The driven wheels use Pololu #3690 80 mm diameter wheels with Pololu #1999 6 mm aluminum hubs [7], [8]. The chassis is currently planned around an approximately 260 mm × 160 mm footprint, although final dimensions and mounting locations will be determined during mechanical integration.

A rigid payload tray will be mounted near the center of the chassis. Payloads will be measured externally before installation and secured at a repeatable marked location. The battery, custom PCB, Raspberry Pi, and other electronics will also be mechanically restrained. Heavy components will be mounted as low and as centrally as practical in order to reduce center-of-mass movement and preserve wheel loading during acceleration and ramp traversal.

2.3.1 Mechanical Layout

The rover will use two smaller wheels at the front and two larger wheels at the rear. The two rear wheels are independently actuated and provide the forces required for both forward motion and differential turning. The front wheels provide mechanical support and help maintain a stable four point chassis configuration.

The physical layout will place the payload tray near the center of the rover so that the 25% and 50% added payload conditions do not significantly bias the load toward either rear drive wheel. The battery will also be mounted low on the chassis to reduce the center of mass height. The Raspberry Pi, PCB, wiring, and sensor assemblies will be positioned so that they do not interfere with wheel motion or the payload area.

The downward facing optical flow sensor will be mounted beneath the chassis near the driven wheel region and will include vertical adjustment so that its required operating distance can be established experimentally. The drag tether used during disturbance testing will attach near the height of the rear drive axle to reduce unwanted pitching moments.

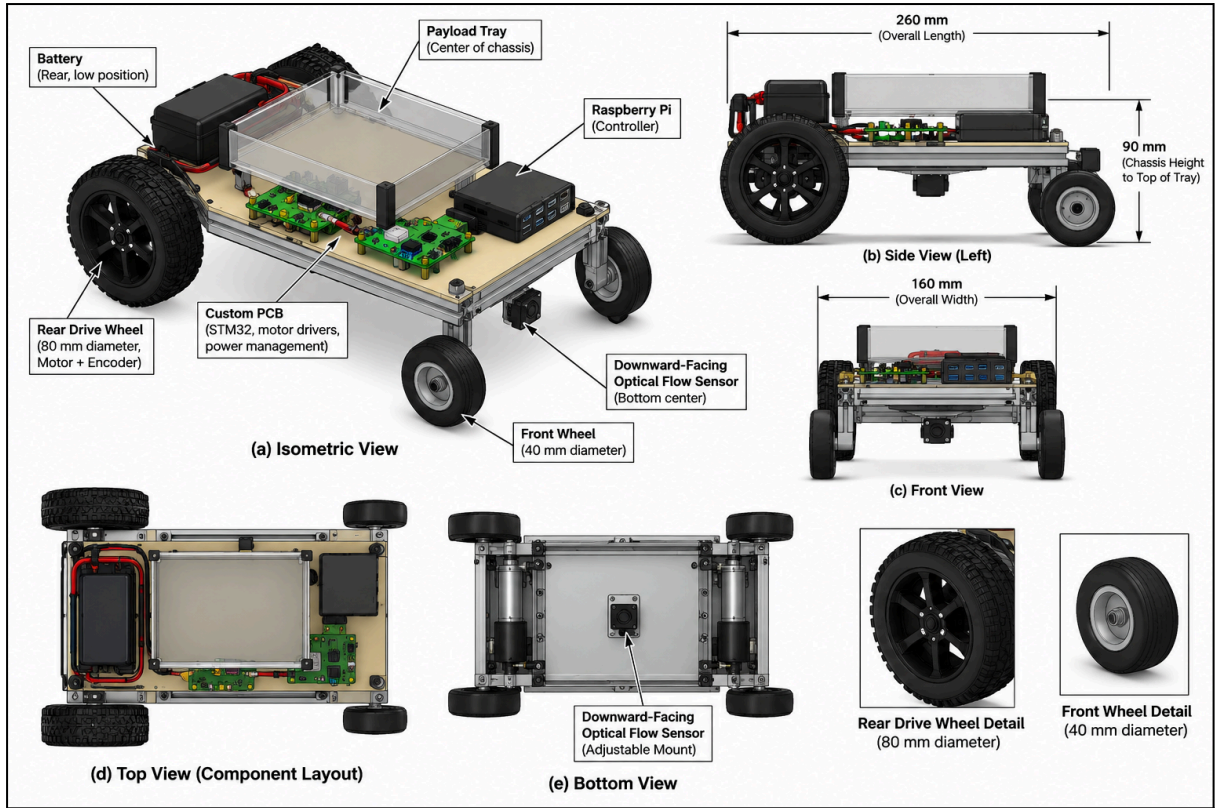


Figure 5: Mechanical Layout

2.3.2 Drivetrain Design

The drivetrain must generate sufficient force to move the rover throughout the defined operating envelope without requiring sustained actuator saturation. The primary physical disturbances include added payload equal to 25% and 50% of the unloaded rover mass, a nominal 5° incline, and approximately 1 N of externally applied drag. The proposal defines forward commands up to approximately 0.20 m/s and yaw-rate commands up to ± 1 rad/s, subject to the experimentally validated wheel-speed and motor-current envelope [28].

A conservative first order force analysis can be used to estimate the drivetrain capability:

$$F_{req} = ma + mg \sin \theta + C_{rr} mg \cos \theta + F_{drag}$$

Using the proposal sizing assumptions of a 3.0 kg loaded rover, 0.10 m/s^2 acceleration, a 5° incline, rolling-resistance coefficient of 0.03, and 1.0 N external drag,

$$F_{req} = 0.300 + 2.565 + 0.880 + 1.000 = 4.745 \text{ N}$$

$$r = 0.040 \text{ m}$$

$$\tau_{wheel} = \frac{F_{req} r}{2} = 0.0949 \text{ Nm}$$

Applying a 20% design allowance gives:

$$\tau_{design} = 0.114 \text{ Nm per rear motor}$$

2.3.3 Mechanical Stability and Interfaces

Mechanical stability is important because wheel unloading or chassis motion could distort the measurements used to evaluate the controller. The rover must therefore keep all components secured and maintain consistent contact between the rear drive wheels and the test surface throughout the validated operating conditions.

The loaded center of mass must remain within the four wheel support region during straight acceleration, turning, and ramp traversal. Particular attention will be given to the 50% payload condition because it produces the largest planned change in total rover mass. Payload placement will remain fixed between matched controller trials.

The subsystem interfaces with the Real Time Control and Communication subsystem through the two rear motor drivers and motor encoders. It interfaces with the Evaluation Sensing subsystem through the payload tray, ramp, drag tether, and optical flow sensor mount.

2.3.4 Requirements and Verification

Requirement	Verification
M1. With 0%, 25%, and 50% added payload relative to measured unloaded rover mass, the mechanical platform shall retain the battery, payload, PCB, Raspberry Pi, and sensor assemblies during straight motion, turning, and traversal of the nominal 5° ramp. No component shall detach or shift from its designated mounting position.	Equipment: external scale, angle measurement device, camera, and assembled rover. Procedure: Measure unloaded rover mass and prepare 25% and 50% payloads. Secure each payload at the designated tray location. Run straight, turning, and low-speed ramp tests. Inspect the rover and video after each test for movement or loosening. Results: Record payload, ramp angle, and pass/fail observations in a table.
M2. The rear drivetrain shall provide sufficient torque to operate within the validated disturbance envelope without sustained actuator saturation. The mechanical sizing target is at least 0.114 N·m per driven wheel under the conservative combined load calculation.	Equipment: rover telemetry, reference velocity measurement, current sensing, payloads, ramp, and drag fixture. Procedure: Increase commanded speed and loading through the intended operating envelope. Record motor current, wheel velocity, reference rover velocity, requested duty, and applied duty. Repeat for payload, incline, and drag conditions. Results: Plot current, duty, and velocity versus time and document the final usable operating envelope.
M3. Both rear drive wheels shall maintain continuous surface contact throughout the required nominal, payload, incline, and	Equipment: video, encoder telemetry, and optical flow/reference ground speed measurement.

turning tests, with no persistent wheel slip that prevents commanded motion within the validated operating envelope.

Procedure: Run the rover through the required conditions while recording encoder-derived wheel speed and ground-reference velocity. Inspect for wheel lift and compare rear wheel velocity with vehicle velocity for sustained slip.

Results: Plot wheel speed and surface speed for the most demanding condition and record any observed loss of contact or persistent slip.

2.4 S2 Real Time Control and Communication

The Real Time Control and Communication subsystem provides deterministic low level motor control and transfers commands and telemetry between the STM32 microcontroller and Raspberry Pi. This subsystem contributes to HL1 by ensuring that controller outputs are applied consistently to the two rear drive motors and to HL3 by enforcing timing, communication integrity, command-freshness, and fail-safe requirements.

The STM32G431CBT6 serves as the rover's real-time controller [2]. It independently controls the left and right rear motor drivers, collects onboard sensor data, constructs telemetry packets, receives controller outputs from the Raspberry Pi, and applies motor current, duty cycle, slew rate, and safety limits.

The STM32 generates separate 20 kHz PWM signals for the two DRV8874 motor drivers [3]. A 1 kHz service task updates sensor estimates, services acquisition, monitors faults, and applies command limits. The Raspberry Pi learned controller operates at 50 Hz, while STM32 telemetry is produced at 100 Hz.

2.4.1 Motor Command Execution

The left and right rear motors are controlled independently so that equal wheel commands produce approximately straight motion while differences between the two commands generate turning motion.

Each DRV8874 motor driver receives PWM and direction control from the STM32 [3]. Motor commands generated by either the learned controller or PI baseline are represented as normalized duty requests. Before reaching the motor drivers, these requests pass through a common command limiting layer.

The command limiting layer applies maximum duty, motor current, and command slew restrictions. Using the same limiting layer for all controller modes ensures that controller comparisons are performed with equivalent actuator constraints.

The initial short duration motor current regulation target is approximately 1.2 A per motor. The final sustained current limit and stall timeout will be determined through thermal characterization of the assembled drivetrain rather than being selected solely from motor ratings.

2.4.2 STM32-Raspberry Pi Communication

The STM32 and Raspberry Pi communicate using a full-duplex 3.3 V UART interface operating at 230400 baud using 8N1 framing. The STM32 transmits telemetry at 100 Hz, while the Raspberry Pi produces new policy commands at 50 Hz.

The proposed maximum telemetry-frame size is 128 bytes. With 8N1 framing,

$$128 \frac{\text{bytes}}{\text{packet}} \times 10 \frac{\text{bits}}{\text{byte}} \times 100 \frac{\text{packets}}{\text{s}} = 128000 \text{ bits/s}$$

$128 \text{ kbit/s} < 230.4 \text{ kbit/s}$, therefore, leaving sufficient nominal bandwidth for the maximum planned telemetry packet. The proposal limits command packets to 32 bytes.

Communication frames contain information needed to determine integrity and freshness, including frame type/version, length, sequence number, timestamp, validity information, and CRC. Telemetry additionally records requested motor duties, actual applied duties, and active limiting conditions.

2.4.3 Command Freshness and Failure Handling

The STM32 accepts a motor command only when its communication frame satisfies the required integrity and freshness checks. CRC invalid, duplicate, malformed, or stale commands will not update the rear motor duties.

Rejected packets also do not refresh the command watchdog. Similarly, nonfinite outputs from the learned controller are rejected rather than being forwarded to the motor drivers.

The command watchdog expires after 100 ms without a valid command. Because the safety/control service executes every 1 ms, the maximum specified disable delay is

$$100 \text{ ms} + 1 \text{ ms} = 101 \text{ ms}$$

This ensures that loss of Raspberry Pi communication cannot leave an old motor command active indefinitely. The independent hardware emergency-stop mechanism belongs to the Power and Safety subsystem and does not depend on successful Raspberry Pi execution.

2.4.4 Requirements and Verification

Requirement	Verification
C1. The STM32 shall generate independent left- and right-rear-motor PWM outputs at 20 kHz, and its real-time service loop shall execute at 1 kHz with a worst-case service interval no greater than 1 ms.	Equipment: oscilloscope or logic analyzer and assembled PCB. Procedure: Probe both PWM outputs and a firmware timing/debug output. Operate both motors while sensing and telemetry are active. Measure PWM frequency and service-task intervals for at least 60 s.

	Results: Report minimum, maximum, and mean PWM frequency and maximum observed service interval.
C2. STM32-Raspberry Pi communication shall use 230400 baud, 8N1, 3.3 V UART, with telemetry generated at 100 Hz and controller commands at 50 Hz. Telemetry packets shall not exceed 128 bytes, and command packets shall not exceed 32 bytes.	Equipment: logic analyzer and UART logs. Procedure: Capture UART communication during normal integrated operation. Decode baud rate, framing, packet length, and packet timestamps. Results: Provide representative decoded frames and a table of measured packet rates and sizes.
C3. Invalid, corrupted, repeated, or stale command packets shall not change the applied rear-motor duties.	Equipment: Raspberry Pi communication-test program and STM32 telemetry logger. Procedure: Send valid commands followed by CRC-corrupted, duplicate, out-of-order, and stale commands. Record requested and applied duty and packet validity. Results: Present a table showing each injected fault and the resulting motor command.
C4. Following loss of valid Raspberry Pi commands, both rear motor drivers shall disable within 101 ms of the last accepted command.	Equipment: oscilloscope or logic analyzer. Procedure: Operate the rover with the wheels safely restrained. Probe command traffic and both driver-enable signals. Stop transmission of valid commands and measure the disable delay. Repeat at least five times. Results: Report each measured delay and the maximum observed delay.
C5. Motor control, sensing, and telemetry shall execute concurrently without corrupting timestamps, commanded direction, requested duty, or applied-duty reporting during maximum intended processing and motor activity.	Equipment: oscilloscope or logic analyzer and system telemetry. Procedure: Run both rear motors while all sensing and communication functions are active. Compare commands, electrical outputs, and telemetry and check sequence-number continuity. Results: Provide timing traces and record any missing, duplicated, or inconsistent packets.

2.5 S3 Proprioceptive Sensing

The Proprioceptive Sensing subsystem measures the rover's onboard mechanical and electrical state using sensors available during normal controller operation. These measurements include left and right rear wheel velocity, left and right motor current, linear acceleration, angular velocity, and battery bus voltage.

This subsystem contributes to HL1 by providing the state information required for motion tracking and contributes to HL2 by supplying the recent observation history from which the adaptive controller estimates changes in rover dynamics.

Unlike the optical flow and drag-force measurements used for evaluation, all proprioceptive sensing signals are available to the deployed controller. S3 observations shall update at least at 100 Hz, include timestamps and validity information, and be no more than 30 ms old when used by the learned policy.

2.5.1 Rear Wheel-Speed Sensing

Each driven rear motor contains a quadrature encoder used to estimate wheel rotational velocity and direction [6]. Both encoder channels are provided to the STM32 so that forward and reverse rotation can be distinguished.

Encoder signals are conditioned before being captured by STM32 timer peripherals. Wheel-speed estimates will use a filtered count window, edge-timing technique, or combination of the two rather than relying only on encoder count differences over a single 1 ms service period. The proposal notes that a 1 ms count difference alone is too coarse for reliable low speed measurements [28].

The required wheel speed sensing accuracy is based on the needs of the controller. The left and right rear wheel speed estimates shall each achieve steady state error of no more than $\pm 0.005 \text{ m/s}$ over an equivalent wheel speed range of 0.05 to 0.30 m/s.

The encoder counts per revolution value in the original proposal should not be reused because it corresponded to the previous motor selection. The final value will be calculated from the encoder and gearbox configuration of the selected Waveshare motor.

2.5.2 Motor Current Sensing

The current drawn by each rear drive motor is measured independently using a 33 mΩ shunt resistor and an INA240A1 current-sense amplifier [31], [4]. Kelvin connections are used across each shunt to reduce measurement error caused by voltage drop through the high-current PCB traces.

Motor winding current is measured rather than only battery current because winding current may differ significantly from total battery current during PWM operation and current recirculation.

The INA240A1 provides a nominal gain of 20 V/V [4]. Therefore, with $R_s = 0.033\Omega$, the nominal sensitivity is $GR_s = 20(0.033) = 0.66 \text{ V/A}$.

$$V_{out} = 1.65 + 0.66I$$

At the two endpoints,

$$I = -1.5 \text{ A} \Rightarrow V_{out} = 0.66 \text{ V}$$

$$I = +1.5\text{ A} \Rightarrow V_{out} = 2.64\text{ V}$$

Both values remain within the STM32's nominal 3.3 V ADC range.

The proposal's worst-case error allocation estimates approximately 0.0424 A of current measurement error at $|I| = 1.5\text{ A}$ [28]. Therefore, the design requirement is a calibrated measurement error of no more than $\pm 0.05\text{ A}$.

2.5.3 IMU Sensing

The custom PCB uses an LSM6DSLTR inertial measurement unit [5]. The IMU provides acceleration and angular-velocity measurements that are included in the rover's onboard observation vector.

Angular velocity about the vertical axis provides the rover's yaw-rate measurement during turning. Accelerometer signals can also provide indirect information about changes in loading and orientation, including the effect of gravity while traveling on an incline.

The IMU does not provide an absolute, drift-free heading measurement. Consequently, the system does not rely on absolute orientation as an input to the adaptive controller.

Samples will be timestamped and aligned with the other onboard observations before being sent to the Raspberry Pi.

2.5.4 Battery Voltage Sensing

The protected battery bus is measured by the STM32 using a resistor-divider and ADC channel. Input filtering and protection prevent switching noise and expected bus transients from exceeding the ADC input range.

Battery voltage is included in the learned-controller observation history because motor response can change as battery voltage varies.

Following calibration, the measured bus voltage shall agree with an external reference measurement within $\pm 0.10\text{ V}$. The proposal defines this as an S3 acceptance target [28].

2.5.5 Requirements and Verification

Requirement	Verification
S3-1. Each rear wheel-speed estimate shall have steady-state error no greater than $\pm 0.005\text{ m/s}$ over the equivalent wheel-speed range of 0.05–0.30 m/s.	Equipment: calibrated wheel-motion or ground-motion reference and STM32 telemetry. Procedure: operate each rear wheel at multiple speeds spanning the required range. Record reference velocity and encoder-derived velocity after steady state is reached. Repeat in forward and reverse directions. Results: Tabulate reference velocity, measured velocity, and error and plot measured versus reference velocity.

S3-2. Each calibrated rear motor-current channel shall measure current over -1.5 A to $+1.5\text{ A}$ with error no greater than $\pm 0.05\text{ A}$.	Equipment: calibrated current source/reference, DMM or current meter, oscilloscope, and STM32 telemetry. Procedure: apply known positive and negative currents including zero and values near $\pm 1.5\text{ A}$. Repeat representative measurements during PWM operation. Results: Produce a calibration plot and table containing reference current, measured current, and error.
S3-3. The protected battery-bus voltage measurement shall agree with a reference DMM within $\pm 0.10\text{ V}$ throughout the validated rover operating-voltage range.	Equipment: adjustable current-limited supply and calibrated DMM. Procedure: apply several bus voltages throughout the intended battery range. At each point record DMM voltage and STM32 telemetry voltage. Repeat representative measurements under compute and motor load. Results: Tabulate reference voltage, reported voltage, and error and report maximum absolute error.
S3-4. Proprioceptive observations supplied to the controller shall update at $\geq 100\text{ Hz}$, contain timestamps and validity information, and be no more than 30 ms old when consumed by the 50 Hz policy loop.	Equipment: synchronized STM32 and Raspberry Pi logs. Procedure: operate the complete system for at least 10 min and record acquisition and policy-consumption timestamps. Calculate update intervals and sensor age for every consumed observation. Results: Report minimum update rate and maximum observation age and plot observation age versus time.
S3-5. Only permitted onboard measurements shall enter the learned-controller history. Payload mass, ramp angle, drag force, optical-flow reference velocity, and trial labels shall not appear in the policy input.	Equipment: controller-input schema and recorded policy-input tensors. Procedure: record the complete model-input feature set during operation. Disable the evaluation-only data stream and confirm continued controller execution. Inspect recorded tensors for prohibited fields. Results: Include the final policy-input feature list and a pass/fail audit of evaluation-signal isolation.

2.6 S4 Power and Safety

The Power and Safety subsystem provides regulated electrical power to the rover electronics and implements the independent hardware mechanisms required to place the drivetrain into a safe disabled state. The subsystem distributes energy from the rover battery to the motor drivers, Raspberry Pi, STM32, sensors, and supporting circuitry while maintaining the voltage rails required for reliable operation.

This subsystem contributes primarily to High-Level Requirement 3 (HL3). The integrated rover must maintain valid power during simultaneous computation and motor operation, and

the drivetrain must disable predictably following communication loss, firmware failure, or activation of the hardware emergency stop. The safety path is designed so that stopping the motors does not depend solely on the Raspberry Pi or learned controller.

The rover uses a Bioenno BLF-1203AB 12 V-class, 3 Ah LiFePO₄ battery as its primary energy source [9]. The battery feeds the protected system bus through a DC-rated fuse, main power switch, and reverse-polarity protection. The current design uses a DFRobot DFR0831 external buck converter to generate the 5 V supply used by the Raspberry Pi and associated 5 V loads [11]. A TLV75533 regulator on the custom PCB generates the 3.3 V logic rail used by the STM32, IMU, current-sensing circuitry, and digital logic [32]. The intended assembled-system acceptance ranges are 4.85–5.15 V for the 5 V rail at loads up to 3.2 A and $\pm 3\%$ for the 3.3 V rail at loads up to 150 mA. These values are subsystem acceptance targets rather than component ratings.

2.6.1 Power Distribution

The battery bus supplies the motor-driver power stage directly through the protected high-current path. The 5 V buck converter is connected to the protected battery bus and supplies the Raspberry Pi through a dedicated power connection. The 3.3 V regulator supplies the STM32 and low-voltage sensing and logic circuitry.

Power wiring will be kept short and sized for the expected current. High-current motor and braking paths will be routed separately from sensitive sensor wiring where practical. Bulk capacitance and transient-protection circuitry on the PCB will reduce disturbances produced by motor startup, reversal, and braking.

The 5 V supply must remain within 4.85–5.15 V under loads up to 3.2 A. The 3.3 V rail must remain within $\pm 3\%$ of nominal at loads up to 150 mA. These rails will be tested under simultaneous Raspberry Pi processing and motor activity because regulator ratings alone do not establish that the required voltage will be maintained at the installed loads.

The preliminary power architecture also includes two external TE Connectivity HSA5010RJ 10 Ω , 50 W braking resistors connected in parallel for a nominal 5 Ω braking load [16]. These resistors require an appropriate heatsink or metal mounting plate and will be thermally characterized before unrestricted use.

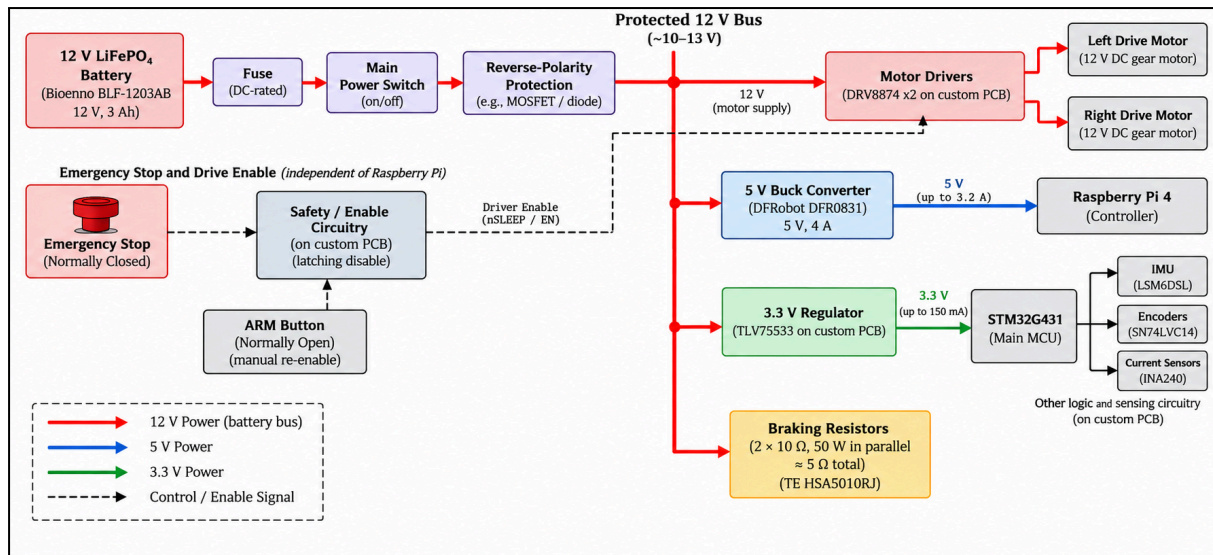


Figure 6: Power Distribution and Hardware Safety Architecture

2.6.2 Independent Hardware Safety

A normally closed emergency-stop loop provides a hardware shutdown path independent of Raspberry Pi software. The emergency-stop switch connects to the custom PCB and controls the latching drive-enable circuitry. When the emergency stop is activated, the hardware circuit forces both DRV8874 motor drivers into their disabled state [3].

The disabled state is also the default during invalid startup conditions. Pull-down circuitry prevents the drivers from becoming enabled before the system has intentionally entered an armed state. Releasing the emergency stop or clearing a fault does not automatically restart the motors. A separate normally open ARM pushbutton must be intentionally activated to restore drive capability.

The MCU additionally implements a command watchdog. If valid commands stop arriving, the command watchdog expires after 100 ms, with no more than an additional 1 ms safety-service delay. This produces the system-level requirement that both drivers disable within 101 ms of the last valid command. A separate MCU watchdog provides protection against firmware lockup.

The initial hardware emergency-stop electrical response target is ≤ 10 ms. This refers to the electrical removal of motor drive enable, not the time required for the rover to physically come to rest. Disabling motor torque does not prevent the rover from coasting or rolling downhill.

2.6.3 Power and Safety Interfaces

The Power and Safety subsystem interfaces directly with the Mechanical Platform and Actuation subsystem through the motor-driver supply and hardware-enable signals. It interfaces with the Real-Time Control subsystem through the watchdog, driver-fault, arm, and enable logic.

It also supplies the regulated 5 V and 3.3 V rails required by the Raspberry Pi, STM32, encoders, sensors, and evaluation hardware. The emergency-stop path remains electrically independent of Raspberry Pi policy execution so that failure of the learned-control computer cannot prevent the drivetrain from being disabled.

2.6.4 Requirements and Verification

Requirement	Verification
P1. The power subsystem shall maintain the 5 V rail between 4.85 V and 5.15 V at loads up to 3.2 A and the 3.3 V rail within $\pm 3\%$ of 3.3 V at loads up to 150 mA during normal operation, motor startup, motor reversal, and simultaneous Raspberry Pi processing. No unintended MCU or Raspberry Pi reset shall occur during these tests.	Equipment: Oscilloscope, digital multimeter, rover battery, assembled rover, and temperature measurement device. Procedure: Operate the Raspberry Pi and STM32 while exercising both motors through startup, acceleration, reversal, and representative high-load operation. Measure the 5 V, 3.3 V, and battery-bus voltages and monitor the system for brownouts or resets. Record regulator temperatures during the test. Results: Plot or tabulate minimum and maximum rail voltage and temperature for each operating condition. Pass if the 5 V and 3.3 V rails remain within their specified ranges with no unintended resets.
P2. Activation of the normally closed hardware emergency stop shall electrically disable both motor drivers within 10 ms, independently of Raspberry Pi software. Releasing the emergency stop shall not automatically restore motor operation; deliberate re-arming shall be required.	Equipment: Oscilloscope or logic analyzer, emergency-stop switch, ARM switch, custom PCB, and restrained rover/drivetrain. Procedure: Enable the drivetrain and command both motors to operate. Activate the emergency stop while monitoring the stop signal and both motor-driver enable/nSLEEP signals. Repeat while the Raspberry Pi continues transmitting valid commands. Release the emergency stop without pressing ARM and verify that drive remains disabled. Results: Record electrical shutdown time for both drivers and the post-stop enable state. Pass if both drivers disable within 10 ms and remain disabled until deliberate re-arming.

P3. Loss of valid commands shall disable both motor drivers within 101 ms of the last valid command. Invalid, stale, repeated, or CRC-failed commands shall not prevent the safety timeout from occurring.

Equipment: Raspberry Pi, STM32, custom PCB, oscilloscope or logic analyzer, and restrained drivetrain.

Procedure: Operate the motors normally and then stop transmission of valid commands. Repeat using stale or invalid command packets where applicable. Measure the time between the final accepted command and motor-driver disable.

Results: Record shutdown time for each test case. Pass if both motor drivers disable within 101 ms of the final valid command for every tested communication-loss condition.

2.7 S5 Evaluation Sensing

2.7.1 Floor-Motion Reference

The Evaluation Sensing subsystem provides independent measurements used to quantify the rover's test conditions and evaluate controller performance. Unlike the proprioceptive measurements used by the deployed controller, evaluation-only measurements are not permitted to enter the learned policy or its observation history.

This subsystem contributes to verification of High-Level Requirements 1 and 2 by measuring the rover's actual surface velocity, applied drag, payload condition, ramp angle, and yaw rate. These measurements allow controller performance to be evaluated independently of encoder-derived wheel motion. This distinction is important because wheel rotation alone does not establish ground speed when wheel slip occurs.

The evaluation system consists of a downward-facing Pimoroni PIM573 optical-flow breakout using the PAA5100JE sensor, a SparkFun SEN-14728 TAL221 500 g load cell and SEN-13879 HX711 amplifier for drag-force measurement, an external scale for payload measurement, direct ramp-angle measurement, and the rover IMU for yaw-rate measurement [13]-[15]. The selected HX711 is configured for 80 samples/s, while the optical-flow sensor is mounted below the chassis and calibrated at its installed operating height [14].

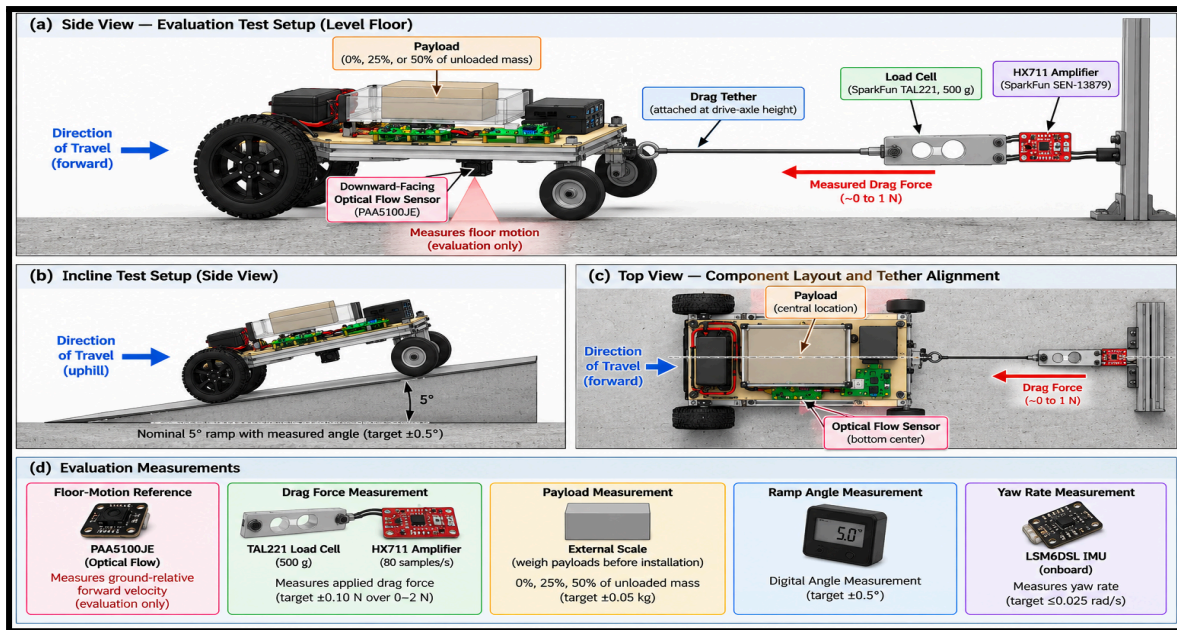


Figure 7: Evaluation Sensing and Experimental Test Configuration

2.7.2 Drag-Force Measurement

The primary drag test uses one SparkFun SEN-14728 TAL221 500 g load cell connected to a SparkFun SEN-13879 HX711 amplifier [14], [15]. The sensor is used only for the external drag fixture and is not an onboard payload scale.

The load cell will be mechanically mounted as a bending beam with force applied along its intended sensing direction. The HX711 will operate at 80 samples/s and the acquired samples will be timestamped. Calibration with known loads will convert raw HX711 readings to force.

The planned drag disturbance is a measured change from approximately 0 N to 1 N while the rover travels at approximately 0.10 m/s. Actual measured force, rather than the commanded or nominal fixture setting, defines the disturbance.

The initial calibrated force-error target is ± 0.10 N over the 0–2 N measurement range. Force repeatability, hysteresis, transition time, and acquisition/filter delay will be characterized before final controller trials.

2.7.3 Payload, Incline, and Yaw Measurements

Payload mass will be determined using an external scale checked against known reference masses. The unloaded rover mass includes its installed electronics and fixed payload tray. Test payloads equal to 25% and 50% of the measured unloaded mass will then be prepared and mounted at the designated central tray location. The initial payload measurement uncertainty target is ± 0.05 kg.

The nominal ramp angle is 5°. Ramp geometry will be measured directly using rise and horizontal run or another checked angle-measurement method. The initial angle uncertainty target is $\pm 0.5^\circ$.

Yaw rate will be obtained from the calibrated onboard IMU. Because the same IMU measurement is available to the controller, it is a shared measurement rather than an independent evaluation-only signal. Gyroscope zero bias and scale will be characterized before testing. Absolute heading and global position are not acceptance metrics.

2.7.4 Evaluation Data Isolation

Evaluation-only measurements must remain separate from the deployed learned controller. Measured payload, ramp angle, drag force, optical-flow velocity, and trial-condition labels shall not enter the learned policy or history encoder.

Only permitted onboard observations such as encoder-derived wheel velocity, acceleration, angular velocity, motor current, battery voltage, previous applied commands, requested motion, and limiting indicators may be supplied to the learned controller.

Evaluation records will retain timestamps and validity information so that force, surface motion, IMU data, controller commands, and onboard telemetry can be synchronized during analysis.

2.7.5 Requirements and Verification

Requirement	Verification
E1. The drag-force measurement system shall measure force over the selected 0–2 N range with calibrated error $\leq \pm 0.10$ N. Force repeatability, hysteresis, transition time, and measurement delay shall be characterized before final controller testing.	Equipment: SparkFun TAL221 500 g load cell, HX711 amplifier, known reference masses/forces, drag fixture, and data logger. Procedure: Apply several known forces throughout the intended 0–2 N measurement range, including repeated increasing and decreasing loads. Determine calibration, repeatability, hysteresis, acquisition delay, and force-step response. Results: Plot measured force versus reference force and calculate maximum error. Record repeatability, hysteresis, and delay. Pass if calibrated error is $\leq \pm 0.10$ N throughout the qualified range.

E2. The downward-facing optical-flow system shall provide a calibrated ground-relative forward-velocity measurement with an initial uncertainty allocation of ≤ 0.005 m/s over the validated test range. Invalid measurements shall be identified rather than replaced by encoder-derived wheel velocity.	Equipment: PAA5100JE optical-flow sensor, measured test lane, timing reference, rover, floor and ramp test surfaces, and data logger. Procedure: Establish the final sensor mounting height and orientation. Perform repeated traverses over known distances and speeds on the intended floor and ramp surfaces. Compare optical-flow-derived surface velocity against the independent reference motion and characterize measurement latency. Results: Plot measured versus reference velocity and calculate error/uncertainty over the operating range. Record invalid samples and latency. Pass if the qualified measurement uncertainty is ≤ 0.005 m/s.
E3. The evaluation system shall quantify the physical test conditions with initial uncertainty targets of ± 0.05 kg for payload mass, $\pm 0.5^\circ$ for ramp angle, and ≤ 0.025 rad/s for yaw rate. Added payloads shall equal 25% and 50% of the measured unloaded rover mass. .	Equipment: External scale, reference masses, angle-measurement device, ramp, onboard IMU, and known angle/time rotation reference. Procedure: Check the external scale using reference masses and measure the unloaded rover and added payloads. Measure the ramp angle using the selected geometric or angle-reference method. Characterize IMU gyro bias at rest and compare yaw-rate measurements against known angle/time motion. Results: Record calibration error and uncertainty for payload mass, ramp angle, and yaw rate. Document the actual masses and ramp angle used during final tests.
E4. Evaluation-only information, including measured payload mass, explicit ramp angle, measured drag force, optical-flow ground velocity, and trial-condition labels, shall not enter the learned policy or history-encoder inputs. Evaluation data shall retain acquisition timestamps and validity information for synchronized post-test analysis.	Equipment: Controller input schema, recorded model-input tensors, evaluation logs, Raspberry Pi, STM32, and analysis software. Procedure: Inspect the deployed controller input schema and recorded model inputs while the evaluation sensors are operating. Verify that evaluation measurements are stored separately from policy/history inputs. Compare timestamps and validity information in the evaluation logs. Results: Document the final policy-input schema and evaluation-data schema. Pass if no evaluation-only measurement or condition label appears in the deployed controller inputs and the evaluation records contain the required timing/validity information.

2.8 S6 Simulation and Learned Control

The Simulation and Learned Control subsystem implements the history-based adaptive controller that forms the central research component of the rover. The subsystem runs primarily on the Raspberry Pi 4 and converts requested rover motion and recent onboard measurements into left and right motor commands.

This subsystem contributes to HL1 by generating commands intended to maintain accurate motion tracking, to HL2 by adapting its behavior using a history-derived representation of rover dynamics, and to HL3 by executing the learned policy within the required real-time deadline.

The learned-control architecture contains a dynamics representation and a motor-control policy. During simulation training, a privileged encoder maps randomized physical parameters into a compact latent representation. The initial design uses eight latent elements. During deployment, those privileged parameters are unavailable. Instead, a history encoder estimates the latent representation using the rover's recent onboard observation and command history [1].

The initial history window contains 30 samples at 50 Hz, corresponding to approximately 0.60 s of recent behavior. Architecture and history length may be adjusted during development and validation, but the selected architecture and trained weights will be frozen before final evaluation.

2.8.1 Simulation and Training

Simulation will represent the differential-drive rover and vary physical properties that affect its dynamics, such as payload-related mass properties, motor response, resistance, and other modeled disturbances. The purpose of this randomization is to expose the policy to a range of dynamic behaviors before deployment.

The privileged encoder used during training may access simulated physical parameters. These privileged values are used to form the training representation but are not available to the deployed controller.

The policy receives the latent representation together with current rover observations and requested motion and generates left and right motor-duty commands. Simulation must preserve the physical actuator and observation constraints of the real rover so that the learned controller is not trained using capabilities unavailable on the hardware.

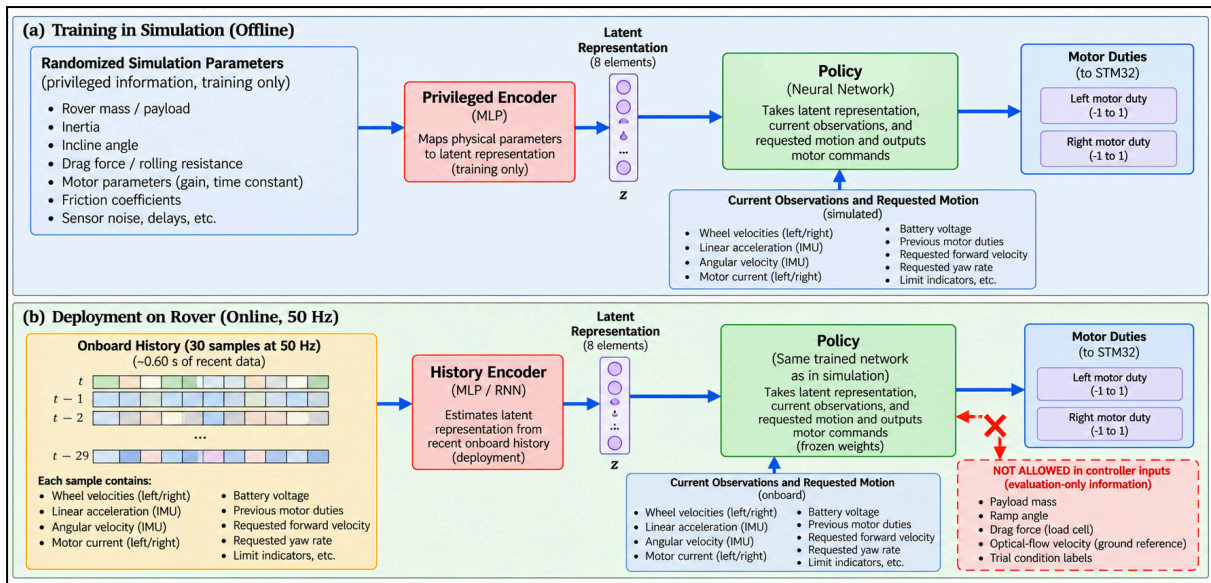


Figure 8: Learned Control Training and Deployment Architecture

2.8.2 History-Based Adaptation

The deployed history encoder receives recent permitted onboard measurements rather than explicit disturbance information. Inputs include left and right wheel velocity, linear acceleration, angular velocity, left and right motor current, battery voltage, previous applied motor duties, requested motion, and controller limiting indicators.

The representation does not need to explicitly estimate quantities such as rover mass, incline angle, or friction coefficient. Instead, it provides a compact description of recent dynamic behavior that the policy can use to modify its motor commands.

Measured payload, explicit ramp angle, drag force, optical-flow velocity, and test-condition labels are excluded from both the history encoder and policy inputs. No disturbance-specific controller switching and no online neural-network weight updates are used.

2.8.3 Raspberry Pi Inference

The learned model executes on the Raspberry Pi 4 at 50 Hz, providing one policy update every 20 ms. The initial inference-time target is <10 ms, leaving time within the 20 ms period for observation preparation, communication, validation, and command transmission.

The complete policy loop, rather than neural-network execution alone, must meet the real-time requirement. Each cycle therefore includes acquisition of the current observation, update of the history buffer, history-encoder inference, policy inference, command validation, and communication of the requested duties to the STM32.

Model outputs containing invalid or nonfinite values will not be applied to the motors. Commands remain subject to the same duty, current, slew-rate, and safety limits used by the other controller modes.

2.8.4 Frozen Model Configuration

The history encoder and policy weights will be frozen before final controller evaluation. Adaptation during a trial occurs through the changing history-derived latent representation rather than through changes to network weights.

Model hashes, architecture information, history length, normalization parameters, input schema, and policy checkpoint will be recorded before final testing. The same learned-policy checkpoint will be used for the fixed-latent and adaptive learned-controller tests so that the contribution of history-based adaptation can be isolated. This follows the proposal's explicit design that the adaptive and constant-latent modes use the same policy checkpoint [28]

2.8.5 Requirements and Verification

Requirement	Verification
L1. The deployed adaptive controller shall infer its dynamics representation from permitted onboard history while using unchanged model weights across all final test conditions. The initial architecture shall use a 30-sample history at 50 Hz and an eight-element latent representation, unless a revised configuration is selected during validation and frozen before final testing.	Equipment: Raspberry Pi 4, trained model, controller software, model files, recorded observation stream, and logging software. Procedure: Record the final model hash, architecture, history length, latent dimension, and input schema before final testing. During representative operation, record history-buffer and latent-representation data. Repeat across nominal and disturbed conditions without changing model weights. Results: Document the frozen model configuration and hashes. Verify that the history encoder receives the configured observation history and generates the configured latent representation while the model weights remain unchanged across final conditions.
C2. The learned-control implementation shall achieve an initial neural-network inference-time target of <10 ms, and the complete onboard policy loop shall execute within the 20 ms period required for 50 Hz operation.	Equipment: Raspberry Pi 4, deployed learned model, high-resolution software timing instrumentation, and representative observation/telemetry load. Procedure: Measure history-encoder inference time, policy inference time, and complete-loop execution time while operating under representative compute and communication load. Perform repeated cycles sufficient to characterize timing variation. Results: Report mean and maximum inference time, mean and maximum complete-loop time, and number of missed 20 ms deadlines. Verify

	whether the <10 ms inference target and ≤ 20 ms complete-loop requirement are satisfied.
L3. The learned controller shall use only permitted onboard observations. Measured payload, explicit ramp angle, measured drag force, optical-flow ground velocity, and test-condition labels shall not be supplied to either the history encoder or policy. No disturbance-specific controller switching or online neural-network weight updates shall occur during final evaluation.	Equipment: Model input schema, controller source/configuration, recorded input tensors, model hashes, and evaluation logs. Procedure: Inspect and record all history-encoder and policy inputs during payload, incline, and drag trials. Compare model hashes before and after trials and inspect controller-mode logic. Results: Provide the final model-input list and model hashes. Pass if evaluation-only variables are absent, model weights remain unchanged, and no disturbance-specific controller switching occurs.
L4. The integrated learned-control system shall sustain a 50 Hz onboard policy loop for 10 minutes with zero missed 20 ms deadlines while control observations and telemetry are acquired/logged at 100 Hz.	Equipment: Complete rover, Raspberry Pi 4, STM32, telemetry logger, timing instrumentation, and restrained or otherwise safe test setup. Procedure: Run the complete control and communication stack continuously for 10 minutes. Record every policy-cycle timestamp, execution time, telemetry sequence number, and deadline indicator. Results: Report total policy cycles, maximum loop execution time, missed-deadline count, and telemetry continuity. Pass if no 20 ms policy deadlines are missed during the 10-minute test while 100 Hz observation/telemetry operation is maintained.

2.9 S7 Baselines and Evaluation

The Baselines and Evaluation subsystem defines the controlled experiments used to compare the rover's adaptive controller against appropriate reference controllers. This subsystem does not provide an additional physical controller or sensor. Instead, it defines the controller configurations, test conditions, repeated-trial procedure, synchronization, performance metrics, and acceptance rules used to determine whether HL1, HL2, and HL3 have been satisfied.

Three controller configurations will be evaluated on the same rover: a conventional fixed-gain PI controller with feedforward and anti-windup, the learned policy using a fixed nominal latent representation, and the same learned policy using the history-derived adaptive latent representation.

All three controllers will share the same calibrated sensors, motor-current limits, duty limits, command slew limits, communication system, and test profiles. This prevents a controller from receiving a hardware or safety advantage unrelated to its control strategy.

2.9.1 Controller Baselines

The first baseline is a fixed-gain wheel PI controller with feedforward and anti-windup. Its gains and feedforward mapping will be tuned using nominal and permitted validation conditions and then frozen before final evaluation.

The second baseline uses the learned policy with a fixed nominal latent representation. The nominal latent is generated from the calibrated nominal model and remains constant during the test. This controller therefore uses the same learned policy as the adaptive mode but does not receive history-based changes to its dynamics representation.

The third configuration is the adaptive learned controller. It uses the same learned policy checkpoint as the fixed-latent mode, but its latent representation is continuously generated from recent onboard history.

This comparison is important because fixed-latent versus adaptive learned control isolates the effect of history-based adaptation, while comparison against PI provides a conventional control reference.

2.9.2 Controlled Test Conditions

The primary test conditions are:

- nominal unloaded operation on a level dry surface;
- 25% added payload;
- 50% added payload;
- unloaded traversal of a nominal 5° incline;
- an approximately 0-to-1 N drag change at approximately 0.10 m/s;
- a separate nominal turning profile using positive and negative yaw-rate commands.

An additional 50% payload + 5° incline at 0.10 m/s condition will be reserved from physical controller tuning and reported separately as a combined-disturbance test. It does not create an additional HL1 or HL2 pass condition.

Five valid repeats will be performed for each controller under each defined primary condition. Controller order will be randomized or counterbalanced where practical. Matched trials will use the same command profiles, starting conditions, and comparable battery-voltage and motor-temperature ranges. The proposal explicitly calls for five valid repeats per controller and condition and retention of all valid outcomes rather than discarding unfavorable runs [28].

Condition	Test Setup Illustration	Description	PI Controller (fixed-gain)	Fixed Latent Learned Policy	Adaptive Learned Policy
Nominal (level floor)		Unloaded rover on level, dry floor; straight speed profile and separate turning profile.	5 trials	5 trials	5 trials
25% payload		Added mass = 25% of measured unloaded mass; payload at central tray location.	5	5	5
50% payload		Added mass = 50% of measured unloaded mass; payload at central tray location.	5	5	5
5° incline		Unloaded rover on nominal 5° ramp; straight speed profile at 0.10 m/s uphill.	5	5	5
~1 N drag		Unloaded rover on level floor; measured change from ~0 to 1 N drag at 0.10 m/s.	5	5	5
Turning (level floor)		Unloaded rover on level floor; positive and negative yaw-rate commands.	5	5	5
50% payload + 5° incline (additional test)		50% payload on nominal 5° ramp at 0.10 m/s. Reported separately; not an additional HL1/HL2 pass condition.	5	5	5

Controllers Evaluated			Test Procedure (for each condition)	Performance Metrics	Results and Analysis
PI Controller - Fixed-gain wheel PI with feedforward - Anti-windup and actuator limits - Gains frozen before final evaluation	Fixed Latent Learned Policy - Same trained policy checkpoint as adaptive - Constant nominal latent representation - No history-based adaptation	Adaptive Learned Policy - Same trained policy checkpoint - Latent representation from recent history (30 samples at 50 Hz) - Frozen weights during evaluation	- Place rover in test configuration (payload / incline / drag). - Run predefined command profile (straight or turning). - Collect synchronized logs (commands, onboard sensors, evaluation measurements). - Repeat for 5 valid trials per controller, with randomized order.	- Forward-speed RMSE ≤ 0.02 m/s (HL1, adaptive controller) - Yaw-rate RMSE ≤ 0.10 rad/s (HL1, turning test) - Drag recovery within 1.5 s and maintain for 0.5 s (HL1) ≥ 10% lower forward-speed RMSE than fixed latent in drag test (HL2) 5 valid repeats per controller and condition (HL3 evaluation)	- Compute RMSE for each trial using calibrated evaluation measurements. - Compare controller performance across all conditions. - Report means, variability, and all valid outcomes (including adverse runs). - Combined 50% + 5° condition reported separately.

Figure 9: Controller Evaluation and Experimental Comparison Matrix

2.9.3 Motion-Tracking Metrics

Forward tracking performance will be evaluated using forward-speed root-mean-square error:

$$RMSE_v = \sqrt{\frac{1}{N} \sum_{k=1}^N (v_{ref,k} - v_{cmd,k})^2}$$

where v_{ref} is calibrated ground-relative forward velocity from the evaluation sensing system, v_{cmd} is requested forward velocity, and N is the number of synchronized samples in the predefined analysis window.

Yaw-rate tracking will be evaluated using the equivalent RMSE calculation with calibrated IMU yaw rate.

For HL1, the adaptive controller shall achieve forward-speed RMSE ≤ 0.02 m/s and yaw-rate RMSE ≤ 0.10 rad/s in at least four of five valid trials for each applicable primary condition. The nominal turning profile is used separately to evaluate the yaw-rate requirement.

2.9.4 Adaptive-Controller Improvement

The percentage improvement of the adaptive controller relative to a baseline is calculated as

$$\text{Improvement}(\%) = 100 \frac{R_{\text{baseline}} - R_{\text{adaptive}}}{R_{\text{baseline}}}$$

where R is the mean trial RMSE for the corresponding controller and condition.

For HL2, the adaptive learned controller shall reduce mean forward-speed RMSE by at least 10% relative to the fixed-latent learned policy in the predefined primary drag condition. Five matched trial pairs will be used.

The proposal's larger improvements—25% relative to fixed-latent and 10% relative to PI across all disturbed conditions—remain research targets rather than required pass conditions [28].

2.9.5 Drag-Recovery Measurement

Drag onset is defined from the measured force signal rather than from the time at which the fixture is manually activated. The onset time is the first crossing of the midpoint between the measured pre-change and post-change force levels that persists for at least 50 ms. Measured acquisition and filtering delay will be corrected during analysis.

Recovery occurs when both applicable HL1 tracking-error bands are re-entered and remain satisfied for 0.5 s. The initial recovery target is 1.5 s following the measured drag change.

A trial in which the rover does not recover before the end of the defined test window is retained as a failed recovery rather than removed from the dataset.

2.9.6 Data Retention and Experimental Validity

Raw synchronized logs will be retained for every valid trial. Each record will include the controller configuration, model hash where applicable, requested motion, reference motion, motor duties, motor currents, limit flags, force measurement, relevant timestamps, and validity information.

Analysis windows, controller settings, and trial-exclusion rules will be established before final data collection. Valid adverse outcomes, including controller saturation or failure to recover, will remain in the dataset. Trials will only be excluded for predefined experimental failures such as invalid measurement data or test-fixture interference.

Five repeats are intended to support evaluation of this rover prototype and are not treated as evidence for broad statistical generalization.

2.9.7 Requirements and Verification

Requirement	Verification
-------------	--------------

V1. The PI, fixed-latent learned, and adaptive learned controllers shall each complete five valid repeats per defined primary condition using common command profiles, sensor calibration, duty/current limits, command-slew rules, and observation timing. All valid adverse outcomes shall be retained, and controller order shall be randomized or counterbalanced where practical.	Equipment: Complete rover, all three controller configurations, payloads, ramp, drag fixture, evaluation sensors, synchronized logger, and test-record sheet. Procedure: Execute the predefined test matrix for nominal, 25% payload, 50% payload, 5° incline, drag, and separate turning conditions. Perform five valid trials per controller and condition. Record controller order, starting conditions, battery voltage, motor temperature where available, and any predefined exclusion reason. Results: Produce a completed test matrix identifying controller, condition, trial number, validity, and exclusion reason where applicable. Retain all valid outcomes, including adverse results.
V2. In at least four of five valid trials for each applicable primary condition, the adaptive controller shall achieve forward-speed RMSE ≤ 0.02 m/s. In the separate nominal turning test, it shall achieve yaw-rate RMSE ≤ 0.10 rad/s in at least four of five valid trials.	Equipment: Calibrated floor-motion reference, calibrated IMU, requested-motion logs, synchronized timestamps, and analysis software. Procedure: Apply the same predefined filtering, synchronization, and analysis window to every controller trial. Calculate forward-speed RMSE from ground-reference velocity and commanded velocity. Calculate yaw-rate RMSE from calibrated IMU yaw rate and commanded yaw rate. Results: Report individual-trial RMSE, mean and spread by controller and condition, and number of adaptive trials meeting each threshold.
V3. In the predefined primary drag condition, the adaptive learned controller shall reduce mean forward-speed RMSE by at least 10% relative to the fixed-latent learned controller, using five matched trial pairs and subject to the validated measurement-resolution rule	Equipment: Five matched fixed-latent/adaptive drag-trial pairs, synchronized evaluation logs, calibrated floor-motion measurement, and analysis software. Procedure: Calculate forward-speed RMSE for each trial using the predefined analysis method. Calculate paired differences and mean RMSE for the fixed-latent and adaptive controllers. Compute percentage improvement as $100(R_{fixed} - R_{adaptive})/R_{fixed}$. Compare the observed difference against validated measurement uncertainty. Results: Report all five paired results, controller means and spread, percentage improvement, and uncertainty assessment. Pass HL2 if the required $\geq 10\%$ reduction is demonstrated and the measurement-resolution condition is satisfied.

V4. Following the measured in-motion drag change, the adaptive controller shall recover to the HL1 forward-speed and yaw-rate error bands within the initial target of 1.5 s and remain within both applicable bands for at least 0.5 s in at least four of five valid drag trials. A trial with no recovery during the defined test window shall be retained as a failed recovery	Equipment: Calibrated force measurement, floor-motion reference, IMU, synchronized controller logs, and analysis software. Procedure: Determine drag onset as the first crossing of the midpoint between measured pre-change and post-change force that persists for 50 ms. Correct for characterized acquisition/filter delay. Determine recovery time as the first entry into both applicable HL1 error bands that persists for 0.5 s. Results: Report measured force rise time, corrected drag-onset time, recovery time for each trial, and whether the 1.5 s target was achieved. Explicitly record trials with no recovery as failures.
--	---

2.10 Tolerance Analysis

2.10.1 Critical Design Parameter

A critical hardware interface for the Rapid Adaptation Rover is the bidirectional motor-current sensing system. Motor current is important because it is both a protection input and an onboard observation available to the history-based adaptive controller. Inaccurate current measurements could provide incorrect information about the rover's changing dynamics or interfere with current protection.

Each of the two motor channels uses a PE2512FKE070R033L 33 mΩ current-sense shunt (R19/R29) and an INA240A1D current-sense amplifier (U5/U6), whose output is measured by the ADC of the STM32G431CBT6 (U1) [31], [4]. The requirement for this interface is to measure bidirectional motor current over ± 1.5 A with a calibrated measurement error no greater than ± 0.05 A, while keeping the amplifier output within the usable range of the 3.3 V ADC.

Because real components have finite accuracy, the analysis considers shunt resistance tolerance and temperature variation, amplifier gain and offset error, ADC/reference error, and residual errors caused by PCB noise and PWM operation.

2.10.2 Mathematical Analysis

The nominal current-sense output is described by

$$V_{out} = V_{ref} + GR_s I$$

where $V_{ref} = 1.65V$, the INA240A1 gain is $20V/V$, and the shunt resistance is 0.033Ω [4]. Therefore,

$$V_{out} = 1.65 + (0.66V/A)I$$

At the maximum positive current of $+1.5A$, the output is

$$V_{out} = 1.65 + (0.66)(1.5) = 2.64V$$

and at $-1.5A$,

$$V_{out} = 1.65 - (0.66)(1.5) = 0.66V.$$

Therefore, under nominal conditions, the full $\pm 1.5A$ measurement range fits within the ADC voltage range.

To determine whether the design remains within its requirements when component variations are considered, a conservative worst-case error analysis is used. The design allocates approximately 1.05% error to the shunt including temperature effects and 0.21% to amplifier scaling, producing approximately 18.90 mA of current error [31], [4]. ADC/reference residual gain error contributes up to 7.50 mA, INA240 input-offset error contributes approximately 0.84 mA, and an allocated 10 mV residual output error from effects including zero-reference error, PWM pickup, filtering, ADC settling, quantization, and noise corresponds to approximately 15.16 mA.

These errors are conservatively added rather than assuming that they cancel:

$$|\Delta I| \leq 1.5(0.0176) + \frac{27.5\mu V}{0.033\Omega} + \frac{0.010V}{0.660V/A}$$

This gives a worst-case current-measurement error of approximately

$$|\Delta I| \approx 0.0424A.$$

Since

$$0.0424A < 0.050A,$$

the calculated error remains within the required $\pm 0.05A$ accuracy. The remaining error margin is

$$0.050 - 0.0424 = 0.0076A = 7.6mA.$$

Finally, the current-sense shunt must safely tolerate the power dissipated at the maximum measurement current. The maximum shunt power is

$$P_{shunt} = I^2 R_s$$

so

$$P_{shunt} = (1.5)^2(0.033) = 0.0743W.$$

A shunt rated for at least 1 W therefore provides substantial steady-state power margin at the required current endpoint.

2.10.3 Expected Worst-Case Performance

The amplifier output must remain within the ADC range even when supply and component tolerances are considered. With the 3.3 V rail allowed to vary by $\pm 3\%$, the reference generated from half of that rail, the combined shunt/amplifier scaling uncertainty, and an additional ± 10 mV residual output error, the conservative amplifier-output range is approximately

$$0.588V \leq V_{out} \leq 2.712V$$

Even at the minimum 3.3 V rail value of 3.201 V, this remains within the restricted ADC design window of approximately 0.200 V to 3.001 V. The circuit therefore maintains approximately 0.289 V of minimum voltage margin, indicating that the ADC should not saturate anywhere within the required ± 1.5 A current range under the analyzed tolerances.

Overall, the tolerance analysis predicts that the motor-current sensing interface can satisfy the ± 1.5 A measurement range and ± 0.05 A accuracy requirement while remaining within the allowable ADC voltage range. The predicted worst-case measurement error is ± 0.0424 A, leaving 7.6 mA of error margin, while the maximum shunt dissipation is approximately 0.0743 W. The largest modeled sources of error are the shunt/amplifier scaling uncertainty and residual output error. Accurate component selection, Kelvin routing, zero-current calibration, filtering, and PWM-synchronized ADC acquisition will therefore be important to achieving the predicted performance. The completed circuit will be verified experimentally using known positive and negative currents and a calibrated reference, including measurements during PWM motor operation.

3 Cost and Schedule

3.1 Cost Analysis

The project cost consists of student engineering labor, the electrical and mechanical components required to construct one rover, and ECE Machine Shop fabrication. Component prices are planning estimates and may vary slightly when orders are placed. Purchase links are included in the bill of materials. Bare PCB fabrication and standard laboratory equipment are provided by the University at no additional project cost.

3.1.1 Labor

Each team member is estimated to contribute approximately 120 hours to the project over the remaining design, assembly, integration, and testing period. An engineering labor rate of \$50 per hour is assumed. This rate is consistent with the engineering salary data used in the project references [24].

Following the required design-document cost formula,

$$\text{LaborCost} = (\text{HourlyRate})(2.5)(\text{ProjectHours})$$

Thus, the estimated labor cost for each team member is

$$\$50 \text{ hour} * 2.5 * 120 \text{ hours} = \$15,000$$

Table 3.1-1. Estimated student engineering labor

Team member	Hours	Rate	Multiplier	Estimated cost
Devansh Agarwal	120	\$50/hr	2.5	\$15,000
Srish Vijay	120	\$50/hr	2.5	\$15,000
Bhavya Agrawal	120	\$50/hr	2.5	\$15,000
Total	360			\$45,000

The labor estimate represents the engineering cost of developing the prototype and is separate from the team's direct expenditure on components.

3.1.2 Parts and Materials

Components installed on the custom PCB are grouped by manufacturer part number. Quantities correspond to the number required for one populated board.

Table 3.1-2. Custom PCB bill of materials

Component / specification	Manufacturer	Part number	Qty.	Est. total	Purchase
2-pin PCB screw terminal	Phoenix Contact	1715721	7	\$17.50	Mouser
Reverse-polarity MOSFET	Diodes Inc.	DMP4015SPS-13	1	\$1.00	Mouser
100 k Ω , 1%, 0805 resistor	Yageo	RC0805FR-07100KL	10	\$1.00	DigiKey
12 V Zener diode	Diodes Inc.	BZT52C12-7-F	1	\$0.20	Mouser
15 V unidirectional TVS	Littelfuse	SMBJ15A	1	\$0.75	Mouser
470 μ F, 35 V capacitor	Panasonic	EEU-FM1V471	3	\$6.00	Mouser
3.3 V regulator	Texas Instruments	TLV75533PDBVR	1	\$0.75	Mouser
10 μ F, 0805 capacitor	Murata	GRM21BR61A106KE19L	4	\$1.40	Origin IC

1 μ F, 50 V, X7R, 0805 capacitor	Murata	GRM21BR71H105KA1 2L	5	\$0.75	LCSC
5 V regulator	Microchip	MCP1799T-5002H/DB	1	\$0.80	Mouser
Comparator with reference	Texas Instruments	TLV3012AIDBVR	1	\$2.50	Mouser
120 k Ω , 1%, 0805 resistor	Yageo	RC0805FR-07120KL	1	\$0.10	Mouser
10 k Ω , 1%, 0805 resistor	Yageo	RC0805FR-0710KL	17	\$1.70	DigiKey
1 M Ω , 1%, 0805 resistor	Yageo	RC0805FR-071ML	1	\$0.10	Mouser
1 nF, 0805 capacitor	KEMET	C0805C102J5GACTU	3	\$0.60	Mouser
10 nF, 0805 capacitor	Murata	GRM21BR71H103KA0 1L	4	\$0.40	Mouser
100 nF, 0805 capacitor	Murata	GRM21BR71H104KA0 1L	23	\$2.30	Mouser
100 Ω , 1%, 0805 resistor	Yageo	RC0805FR-07100RL	1	\$0.10	Mouser
Braking MOSFET	Infineon	IRLZ44NPBF	1	\$1.50	DigiKey

1 k Ω , 1%, 0805 resistor	Yageo	RC0805FR-071KL	10	\$1.00	DigiKey
Schottky diode	Nexperia	BAT54H,115	4	\$0.80	Mouser
STM32 microcontroller	STMicroelectronics	STM32G431CBT6	1	\$6.00	DigiKey
0 Ω , 0805 resistor	Yageo	RC0805JR-070RL	1	\$0.10	DigiKey
6-pin, 2.54 mm header	Würth Elektronik	61300611121	2	\$1.20	Mouser
RESET pushbutton	Omron	B3F-1000	1	\$0.50	DigiKey
Green status LED	Lite-On	LTST-C170KGKT	1	\$0.20	Mouser
H-bridge motor driver	Texas Instruments	DRV8874PWPR	2	\$8.00	DigiKey
22 nF, 50 V, X7R, 0805 capacitor	Murata	GRM21BR71H223KA01L	2	\$0.30	DigiKey
3.01 k Ω , 1%, 0805 resistor	Yageo	RC0805FR-073K01L	2	\$0.20	Mouser
0.033 Ω current shunt	Yageo	PE2512FKE070R033L	2	\$1.60	DigiKey

Current-sense amplifier	Texas Instruments	INA240A1D	2	\$7.00	DigiKey
10 Ω , 1%, 0805 resistor	Yageo	RC0805FR-0710RL	4	\$0.40	Mouser
100 pF, 50 V, C0G, 0805 capacitor	KEMET	C0805C101J5GACTU	4	\$0.80	Mouser
Schmitt-trigger inverter	Texas Instruments	SN74LVC14ADR	1	\$0.80	Mouser
4-pin, 2.54 mm header	Würth Elektronik	61300411121	3	\$1.35	Mouser
Six-axis IMU	STMicroelectronics	LSM6DSLTR	1	\$5.00	DigiKey
3-pin, 2.54 mm header	Würth Elektronik	61300311121	1	\$0.35	Mouser
330 Ω , 1%, 0805 resistor	Yageo	RC0805FR-07330RL	2	\$0.20	Mouser
2-pin, 2.54 mm header	Würth Elektronik	61300211121	2	\$0.50	Mouser
N-channel small-signal MOSFET	Nexperia	2N7002,215	1	\$0.25	Mouser
Dual D-type flip-flop/latch	Nexperia	74LVC74AD,118	1	\$0.80	Mouser

Supply/reset supervisor	Texas Instruments	TPS3808G33DBVR	1	\$1.50	Mouser
External watchdog	STMicroelectronics	STWD100NPWY3F	1	\$1.50	DigiKey
8-pin, 2.54 mm test header	Würth Elektronik	61300811121	2	\$1.50	DigiKey
PCB components subtotal			140	\$81.30	

The remainder of the rover consists of the computation platform, drivetrain, sensing hardware, battery system, braking components, controls, wiring, and mechanical hardware.

Table 3.1-3. Rover electronics, drivetrain, and sensing components

Component / specification	Manufacturer / part number	Qty.	Est. total	Purchase
Raspberry Pi 4 Model B, 1 GB	Raspberry Pi, RPI4-MODBP-1GB	1	\$35.00	Newark
32 GB microSDHC	SanDisk, SDSQQNR-032G-GN6IA	1	\$10.00	SanDisk
Pi fan/heatsink	Adafruit 4794	1	\$8.00	Adafruit
12 V encoder gearmotor	Waveshare DCGM-3865-12V-EN-240RPM	2	\$19.98	Waveshare
80 × 10 mm driven wheels	Pololu #3690	1 pair	\$10.00	Pololu

6 mm M3 shaft hubs	Pololu #1999	1 pair	\$10.00	Pololu
12 V, 3 Ah LiFePO4 battery	Bioenno BLF-1203AB	1	\$49.99	Bioenno Power
14.6 V, 2 A LiFePO4 charger	Bioenno BPC-1502DC	1	\$21.99	Bioenno Power
7–24 V to 5 V/4 A buck converter	DFRobot DFR0831	1	\$4.90	DFRobot
Optical-flow breakout	Pimoroni PIM573 / PAA5100JE	1	\$30.00	Pimoroni
HX711 load-cell amplifier	SparkFun SEN-13879	1	\$12.00	SparkFun
500 g TAL221 load cell	SparkFun SEN-14728	1	\$12.00	SparkFun
10 Ω / 50 W braking resistor	TE Connectivity HSA5010RJ	2	\$18.36	Mouser
Brake-resistor thermal mounting plate/heatsink	Custom aluminum plate, ECE Machine Shop	1	\$12.00	Custom fabrication/material allowance
Subtotal			\$254.22	

Table 3.1-4. Power controls, wiring, and mechanical hardware

Component / specification	Manufacturer / part number	Qty.	Est. total	Purchase
----------------------------------	-----------------------------------	-------------	-------------------	-----------------

In-line ATO fuse holder and 5 A ATO fuse	Littelfuse FHAC0003ZXJA + 0287005.PXCN	1 each	\$7.78	Fuse holder – DigiKey ; 5 A fuse – DigiKey
DC-rated DPST main ON/OFF switch	NKK Switches S21A	1	\$12.48	Mouser
22 mm red mushroom emergency-stop, twist release, 1 NC contact, IP65	AutomationDirect GCX3131	1	\$12.00	AutomationDirect
Panel-mount momentary ARM pushbutton, SPST-NO, 30 VDC	E-Switch LP10A1AB	1	\$3.79	DigiKey
15 A red/black Powerpole connector sets	Anderson Power / Powerwerx PP15-10	10-set pack	\$12.99	Powerwerx
18 AWG, 2-conductor power/charging cable	Alpha Wire 1897C SL212	5 ft	\$7.82	DigiKey
22 AWG, 3-conductor UART cable	Alpha Wire 1173C SL013	5 ft	\$5.15	DigiKey
22 AWG stranded sensor/control wiring assortment, six colors	SparkFun PRT-11375	1 set	\$25.95	SparkFun
2.54 mm mating housings and female contacts for 2-, 3-, 4-, 6-, and 8-pin headers	Pololu/PCX #1901, #1902, #1903, #1905, #1907, #1930	1 pack each	\$23.90	2-pin #1901 ; 3-pin #1902 ; 4-pin #1903 ; 6-pin #1905 ; 8-pin #1907 ; contacts #1930

Heat-shrink tubing, cable ties, and strain-relief clamps	3M FP-301 1/8", Panduit PLT1M-M109, Essentra 22CC50C0125B	1 set	\$5.73	3M heat-shrink – DigiKey ; Panduit cable ties – DigiKey ; Essentra clamps – DigiKey
Controls and wiring subtotal			\$117.59	
Wooden chassis/base material	Custom plywood base, ECE Machine Shop	1	\$20.00	Custom fabrication/material allowance
40 mm front swivel caster wheel, ABS, top-plate mount	VXB Bearings VXB20040AL	2	\$5.98	Small Bearings
M3 × 10 mm female-female brass PCB standoffs	Keystone Electronics 24393	4	\$3.68	DigiKey
ABS electronics enclosure, approximately 190 × 110 × 61 mm	Hammond Manufacturing 1591ESFLBK	1	\$12.19	DigiKey
M3 stainless-steel screws, nuts, washers, and spacers assortment	MYWISH SF-TZ00085	1 set	\$17.73	Home Depot
Mechanical/material subtotal			\$59.58	

The motor mounts, payload platform, battery mounting features, sensor mounts, and related chassis features will be fabricated as part of the machine-shop chassis rather than purchased as separate assemblies. This avoids counting the same mechanical structure twice.

The complete estimated hardware and materials cost is therefore \$512.69

Machine Shop Fabrication

The ECE Machine Shop will fabricate the rover chassis and mechanical mounting structure. The team will provide the motors, wheels, battery, sensors, PCB, Raspberry Pi, and other purchased components so that the chassis can be constructed around the actual hardware.

Based on the preliminary discussion with the machine shop, approximately **8 labor hours** are budgeted for chassis fabrication and mechanical integration. The shop indicated that the fabrication is straightforward and should take approximately three working days from start to completion. Machine-shop labor is provided through University resources and therefore has an estimated **team cash cost of \$0**. A conservative **\$20 allowance for wooden chassis material** is included in the bill of materials in case the material must be purchased rather than supplied by the shop.

Bare PCB fabrication is also University-provided and therefore contributes **\$0 additional cash cost**.

3.1.3 Total Project Cost

Table 3.1-5. Estimated project cost

Cost category	Estimated cost
PCB components	\$81.30
External electronics, drivetrain, and sensors	\$254.22
Power controls and wiring	\$117.59
Mechanical/chassis materials and hardware	\$59.58
ECE Machine Shop labor, approximately 8 hours	\$0.00
Bare PCB fabrication	\$0.00

Hardware and materials subtotal	\$512.69
Student engineering labor	\$45,000.00
Total estimated project development cost	\$45,512.69

The approximately **\$513 hardware and materials subtotal** represents the expected direct cost of constructing one rover under the stated University fabrication assumptions. The larger total development cost includes the salary-equivalent engineering labor calculation required by the design-document guidelines and is not an amount the team expects to pay directly.

3.2 Schedule

3.2.1 Weekly Project Schedule and Team Responsibility Distribution Table

Week	Task	Person
October 5 – October 11	Finalize custom STM32 PCB design and prepare it for fabrication submission	Prish & Devansh (Hardware)
	Finalize mechanical dimensions, component locations, mounting requirements, and rover design for machine-shop fabrication	Prish & Devansh (Hardware)
	Finalize parts list and submit mechanical requirements and parts information to the machine shop	Prish & Devansh (Hardware)
	Order or acquire remaining electrical and mechanical components	Everyone
	Begin STM32 firmware and Raspberry Pi–STM32 UART communication	Bhavya (Software)
October 12 – October 18	Begin rover simulation and baseline PI controller implementation	Bhavya (Software)
	PCB and mechanical rover structure in fabrication	Machine Shop / PCB Fabrication
	Verify purchased parts against PCB and mechanical interfaces as they arrive	Prish & Devansh (Hardware)
	Prepare PCB components, connectors, wiring, motors, sensors, and test equipment for assembly	Prish & Devansh (Hardware)
	Develop encoder, IMU, motor-current, and battery-voltage acquisition firmware	Bhavya (Software)
October 19 – October 25	Continue PI controller and rover simulation development	Bhavya (Software)
	Continue PCB and mechanical fabrication and inspect completed components as available	Prish & Devansh (Hardware)
	Prepare wiring harnesses and hardware integration plan	Prish & Devansh (Hardware)
	Complete STM32–Raspberry Pi communication protocol and telemetry format	Bhavya (Software)
	Implement 50 Hz controller loop and 100 Hz sensor/telemetry handling	Bhavya (Software)
October 26 – November 1	Begin learned-controller and history-encoder implementation	Bhavya (Software)
	Receive and inspect fabricated PCB and machine-shop rover components	Prish & Devansh (Hardware)
	Solder components onto custom PCB	Prish & Devansh (Hardware)
	Perform PCB bring-up and verify power, STM32, motor drivers, current sensing, IMU, UART, ARM circuit, watchdog, and emergency stop	Prish & Devansh (Hardware)
	Develop diagnostic firmware and support PCB bring-up	Bhavya (Software)
November 2 – November 8	Prepare functioning hardware and software for progress demonstration	Everyone
	Assemble chassis with motors, wheels, battery, payload tray, Raspberry Pi, PCB, and optical-flow sensor	Prish & Devansh (Hardware)
	Integrate custom PCB, Raspberry Pi, motors, encoders, sensors, and safety hardware	Everyone
	Perform restrained motor tests and verify emergency-stop and watchdog behavior	Everyone
	Calibrate wheel-speed, current, battery-voltage, IMU, and optical-flow measurements	Everyone
November 9 – November 15	Refine simulation using measurements from physical rover	Bhavya (Software)
	Characterize rover motor response, wheel speed, current, acceleration, communication delay, and battery behavior	Everyone
	Set up and calibrate TAL221/HX711 drag-force measurement system	Prish & Devansh (Hardware)
	Tune and freeze baseline PI controller	Bhavya (Software)
	Train and validate fixed-latent and history-adaptive learned controllers	Bhavya (Software)
November 16 – November 22	Benchmark learned-controller inference time on Raspberry Pi 4	Bhavya (Software)
	Deploy fixed-latent and history-adaptive controllers onto the physical rover	Bhavya (Software)
	Conduct low-speed integrated testing of PI, fixed-latent, and adaptive controllers	Everyone
	Perform preliminary payload, incline, turning, and measured-drag experiments	Everyone
	Debug mechanical, sensing, power, wiring, and PCB issues	Prish & Devansh (Hardware)
November 23 – November 29	Debug controller, communication, timing, and telemetry issues	Bhavya (Software)
	Prepare rover for mock demo and verify complete system operation	Everyone
	Correct remaining minor hardware problems and prepare spare wiring/components	Prish & Devansh (Hardware)
	Correct remaining software/controller issues and freeze final software	Bhavya (Software)
	Freeze final hardware configuration, controller parameters, network weights, and experimental procedures	Everyone
November 30 – December 4	Prepare final test profiles and data-logging configuration	Everyone
	Conduct final matched PI, fixed-latent, and adaptive-controller experiments	Everyone
	Complete nominal, payload, incline, drag, and turning tests	Everyone
	Calculate forward-speed RMSE, yaw-rate RMSE, recovery time, adaptation improvement, and timing results	Bhavya (Software)
	Verify final hardware requirements, sensing accuracy, and safety operation	Prish & Devansh (Hardware)
	Final Demo and Presentation	Everyone

4 Discussion of Societal Impact, Engineering Standards, Ethics, and Safety Considerations

4.1 Societal Impact

The Rapid Adaptation Rover contributes to public health, safety, and welfare by investigating control methods that can maintain more consistent motion when a small mobile robot experiences changes in payload, incline, or mechanical resistance. A robot that responds unpredictably to changing operating conditions could move faster or slower than intended, turn incorrectly, or fail to respond appropriately to disturbances. The history-adaptive controller is intended to reduce these changes in motion response by using recent onboard measurements to account for changing rover dynamics. The project also incorporates physical safety features including a hardware emergency stop, command watchdog,

motor-current monitoring, command limits, controlled acceleration, battery protection, and hardware drive-disable functionality. The prototype is restricted to controlled testing and is not intended for autonomous operation around untrained users.

The project also considers economic, environmental, social, cultural, and global factors. Economically, adaptive control could allow small transport robots to perform effectively across different loads without requiring separate manual controller tuning for every operating condition. Environmentally, the rover uses a rechargeable battery, and adaptation may support more efficient use of the motors by producing commands appropriate for the current operating condition. However, electronic components and batteries also create manufacturing and end-of-life environmental impacts that should be considered. Socially, more predictable mobile robots could support applications such as indoor material transport while reducing repetitive transportation tasks for people. The design does not depend on a particular language, culture, or user demographic, although future deployment would need to consider accessibility and differences in how people interact with autonomous systems. Globally, low-cost embedded computers, sensors, and adaptive-control techniques could make similar robotic systems useful in many environments, but differences in regulations, infrastructure, cost, maintenance resources, and expectations of robot safety would need to be considered before broader deployment.

4.2 Engineering Standards

Several engineering standards and professional guidelines are relevant to the rover's electrical, battery, embedded-computing, and safety design. IEC 62133-2 specifies safety requirements and tests for portable rechargeable lithium cells and batteries under intended use and reasonably foreseeable misuse [29]. This is relevant to the project's use of a rechargeable lithium-based battery and reinforces the importance of appropriate charging, electrical protection, mechanical protection, and operation within the battery manufacturer's limits.

UL 2271, Batteries for Use in Light Electric Vehicle Applications, is another useful battery-safety reference because it addresses battery packs used in light electrically powered vehicles and evaluates safety-related aspects of those energy-storage systems [30]. The Rapid Adaptation Rover is not being represented as UL 2271 certified, but the standard provides relevant safety guidance for the design of a mobile battery-powered system.

The IEEE Code of Ethics and ACM Code of Ethics and Professional Conduct are also applicable professional guidelines because the project combines electrical engineering, embedded computing, and learned control. IEEE emphasizes placing public safety, health, and welfare first, considering the consequences of technology, and being honest about technical results. The ACM Code similarly emphasizes contributing to human well-being, avoiding harm, and considering the environmental and societal consequences of computing systems. These standards and guidelines are used as references for responsible design rather than as claims that the prototype has received formal certification [25], [26].

4.3 Ethics

The IEEE and ACM codes of ethics directly relate to the rover because failures in electrical hardware, software, or learned control could potentially cause physical harm or property damage. IEEE emphasizes that engineers should prioritize public safety, health, and welfare and should be honest and realistic when reporting technical claims [25]. Similarly, the ACM Code emphasizes that computing professionals should contribute to human well-being and avoid harm, including unintended harm resulting from computing systems [26]. These principles are reflected in our design through the hardware emergency stop, communication watchdog, motor-current protection, restricted command ranges, controlled testing environment, and separation of safety-critical motor shutdown functions from the Raspberry Pi learned controller.

An additional ethical concern is the honest evaluation of the adaptive controller. Because this is an experimental learned-control system, the team should not claim that adaptation improves performance unless the predefined tests support that conclusion. Poor-performing but valid trials should not be removed simply because they weaken the results.

Evaluation-only information such as payload mass, ramp angle, drag force, optical-flow reference velocity, and trial labels should not enter the learned controller. Acceptance requirements should also not be weakened after seeing the final results simply to make the project appear successful. Finally, performance demonstrated within the project's tested payload, incline, speed, and disturbance ranges should not be represented as proof that the rover is safe in arbitrary environments. The team should clearly report the rover's validated operating range, limitations, failures, and experimental results.

4.4 Electrical and Mechanical Safety

The rover presents electrical, thermal, mechanical, and control-related safety concerns. Testing and construction will follow applicable laboratory safety practices and established machinery and emergency-stop safety principles [19]-[21], [23].

Electrical risks include battery short circuits, incorrect polarity, excessive motor current, damaged wiring, and voltage transients produced during motor acceleration or braking. A fuse will be installed near the battery, high-current connections will be insulated, and the battery will only be charged using its matched charger [9], [10], [22]. The STM32 motor-control system will enforce current and command limits independently of the Raspberry Pi. The emergency-stop circuit, external watchdog, and supply supervisor provide additional protection if communication or software fails [17], [18], [20].

Regenerative braking can increase the DC-bus voltage and heat the braking resistors. The two braking resistors will therefore be mounted to an appropriate heat-dissipating surface and tested for safe temperature before extended operation [16].

Mechanical hazards include moving wheels, pinch points, unexpected rover motion, loose payloads, and rollback during ramp testing. The battery, electronics, and payload will be

mechanically secured to the chassis. Ramp testing will use a controlled test area and an independent physical restraint so that the rover cannot roll away if control is lost.

As the design developed beyond the original proposal, additional safety features were added, including the emergency-stop system, watchdog, supply supervisor, braking circuit, and specific battery and drivetrain components. These additions require verification of shutdown behavior, thermal performance, wiring, mechanical stability, and fault handling before unrestricted rover testing. The team will report safety failures, overheating, unstable behavior, and other adverse results accurately rather than excluding unsuccessful tests.

4.5 Safety Mitigation Procedures

Safety testing will proceed in stages. Before the rover is powered, the team will inspect solder joints, component orientation, connector wiring, battery polarity, fuses, motor connections, and exposed conductors. Initial power-up will use a current-limited bench supply with the motors disabled. The 5 V and 3.3 V rails will be verified before the battery or drivetrain is used.

The battery will be charged only with the matched charger and will be inspected for damage or abnormal heating before use [10], [22]. Initial motor tests will be performed with the wheels raised or the rover physically restrained. Motor current and component temperature will be monitored while commands are increased gradually.

Before normal floor operation, the emergency-stop and fault-response systems will be tested. These tests will include emergency-stop activation, communication loss, malformed or stale commands, MCU reset, watchdog activation, supply-supervisor response, driver faults, and invalid learned-controller outputs. Any detected fault will disable motor drive, and clearing a fault will not automatically restart the rover.

The braking circuit will initially be tested at low rover speed. Bus voltage and braking-resistor temperature will be monitored before higher-speed or repeated braking tests are performed. Mechanical checks before each test will include the wheels, motor mounts, caster, battery restraint, payload attachment, wiring clearance, and electronics mounting.

Floor tests will be conducted in a clear area with at least one team member able to activate the emergency stop. Ramp tests will additionally use side guidance or an independent restraint to prevent uncontrolled rollback. Payloads will only be added or removed while the rover is stopped and disarmed. If any test produces overheating, unstable motion, damaged wiring, abnormal current, or unexpected controller behavior, testing will stop until the problem is identified and corrected. Relevant safety tests will then be repeated before normal operation resumes.

5 References

- [1] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid Motor Adaptation for Legged Robots," in Proc. Robotics: Science and Systems XVII, Virtual, Jul. 2021, doi: 10.15607/RSS.2021.XVII.011. [Online]. Available: <https://roboticsproceedings.org/rss17/p011.html>. [Accessed: Oct. 2, 2026].
- [2] STMicroelectronics, "STM32G431x6, STM32G431x8, STM32G431xB: Arm Cortex-M4 32-bit MCU," datasheet. [Online]. Available: <https://www.st.com/resource/en/datasheet/stm32g431cb.pdf>. [Accessed: Oct. 2, 2026].
- [3] Texas Instruments, "DRV8874 H-Bridge Motor Driver With Integrated Current Sense and Regulation," datasheet, Rev. A. [Online]. Available: <https://www.ti.com/lit/ds/symlink/drv8874.pdf>. [Accessed: Oct. 2, 2026].
- [4] Texas Instruments, "INA240 -4-V to 80-V, Bidirectional, Ultra-Precise Current Sense Amplifier With Enhanced PWM Rejection," datasheet, Rev. C. [Online]. Available: <https://www.ti.com/lit/ds/symlink/ina240.pdf>. [Accessed: Oct. 2, 2026].
- [5] STMicroelectronics, "LSM6DSL: iNEMO Inertial Module—3D Accelerometer and 3D Gyroscope," datasheet. [Online]. Available: <https://www.st.com/resource/en/datasheet/lsm6dsl.pdf>. [Accessed: Oct. 2, 2026].
- [6] Waveshare, "DCGM-3865-12V-EN-240RPM Encoder Gear Motor," product documentation, SKU 22346. [Online]. Available: <https://www.waveshare.com/wiki/DCGM-3865-12V-EN-240RPM>. [Accessed: Oct. 2, 2026].
- [7] Pololu Corporation, "Multi-Hub Wheel w/Inserts for 3 mm and 4 mm Shafts—80×10 mm, Black, 2-Pack," product 3690. [Online]. Available: <https://www.pololu.com/product/3690>. [Accessed: Oct. 2, 2026].
- [8] Pololu Corporation, "Universal Aluminum Mounting Hub for 6 mm Shaft, M3 Holes, 2-Pack," product 1999. [Online]. Available: <https://www.pololu.com/product/1999>. [Accessed: Oct. 2, 2026].
- [9] Bioenno Power, "12 V 3 Ah LiFePO4 Battery—BLF-1203AB," product documentation. [Online]. Available: <https://www.bioennopower.com/products/12v-3ah-lifepo4-battery-pvc>. [Accessed: Oct. 2, 2026].
- [10] Bioenno Power, "14.6 V, 2 A AC-to-DC Charger for 12 V LiFePO4 Batteries—BPC-1502DC," product documentation. [Online]. Available: <https://www.bioennopower.com/products/lithium-12v-2a-amp-lifepo4-battery-charger>. [Accessed: Oct. 2, 2026].
- [11] DFRobot, "DC-DC Buck Converter 7–24 V to 5 V/4 A," DFR0831. [Online]. Available: <https://www.dfrobot.com/product-2240.html>. [Accessed: Oct. 2, 2026].
- [12] Raspberry Pi Ltd., "Raspberry Pi 4 Model B," product documentation and product brief. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/>. [Accessed: Oct. 2, 2026].

- [13] Pimoroni Ltd., “PAA5100JE Near Optical Flow SPI Breakout,” PIM573. [Online]. Available: <https://shop.pimoroni.com/products/paa5100je-optical-tracking-spi-breakout>. [Accessed: Oct. 2, 2026].
- [14] SparkFun Electronics, “SparkFun Load Cell Amplifier—HX711,” SEN-13879. [Online]. Available: <https://www.sparkfun.com/sparkfun-load-cell-amplifier-hx711.html>. [Accessed: Oct. 2, 2026].
- [15] SparkFun Electronics, “Mini Load Cell—500 g, Straight Bar (TAL221),” SEN-14728. [Online]. Available: <https://www.sparkfun.com/mini-load-cell-500g-straight-bar-tal221.html>. [Accessed: Oct. 2, 2026].
- [16] TE Connectivity, “HSA5010RJ: 10 Ω , 50 W Aluminum-Housed Power Resistor,” part 1625984-5. [Online]. Available: <https://www.te.com/en/product-1625984-5.html>. [Accessed: Oct. 2, 2026].
- [17] STMicroelectronics, “STWD100 Watchdog Timer Circuit,” datasheet. [Online]. Available: <https://www.st.com/resource/en/datasheet/stwd100.pdf>. [Accessed: Oct. 2, 2026].
- [18] Texas Instruments, “TPS3808 Low-Quiescent-Current, Programmable-Delay Supervisory Circuit,” datasheet, Rev. N. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tps3808.pdf>. [Accessed: Oct. 2, 2026].
- [19] International Organization for Standardization, Safety of Machinery—General Principles for Design—Risk Assessment and Risk Reduction, ISO 12100:2010, Nov. 2010. [Online]. Available: <https://www.iso.org/standard/51528.html>. [Accessed: Oct. 2, 2026].
- [20] International Organization for Standardization, Safety of Machinery—Emergency Stop Function—Principles for Design, ISO 13850:2015, Nov. 2015. [Online]. Available: <https://www.iso.org/standard/59970.html>. [Accessed: Oct. 2, 2026].
- [21] University of Illinois Urbana-Champaign, ECE 445 Senior Design Laboratory, “Safety Guidelines.” [Online]. Available: <https://courses.grainger.illinois.edu/ece445/guidelines/safety.asp>. [Accessed: Oct. 2, 2026].
- [22] University of Illinois Urbana-Champaign, Division of Research Safety, “Battery Safety.” [Online]. Available: <https://www.drs.illinois.edu/Page/SafetyLibrary/BatterySafety>. [Accessed: Oct. 2, 2026].
- [23] University of Illinois Urbana-Champaign, Division of Research Safety, “Electrical Safety in the Research Laboratory.” [Online]. Available: <https://www.drs.illinois.edu/Page/SafetyLibrary/ElectricalSafetyInTheResearchLaboratory>. [Accessed: Oct. 2, 2026].
- [24] U.S. Bureau of Labor Statistics, “Occupational Employment and Wages—May 2024,” Occupational Employment and Wage Statistics, Apr. 2, 2025. [Online]. Available: https://www.bls.gov/news.release/archives/ocwage_04022025.pdf. [Accessed: Oct. 2, 2026].

[25] Institute of Electrical and Electronics Engineers, "IEEE Code of Ethics," IEEE Policies, sec. 7.8. [Online]. Available: <https://www.ieee.org/about/corporate/governance/p7-8.html>. [Accessed: Oct. 2, 2026].

[26] Association for Computing Machinery, "ACM Code of Ethics and Professional Conduct," 2018. [Online]. Available: <https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct>. [Accessed: Oct. 2, 2026].

[28] D. Agarwal, S. Vijay, and B. Agrawal, "Rapid Adaptation Rover: Learned Dynamics Compensation for a Differential Drive Robot," ECE 445 project proposal, University of Illinois Urbana-Champaign, Sep. 17, 2026.

[29] International Electrotechnical Commission, Secondary Cells and Batteries Containing Alkaline or Other Non-Acid Electrolytes—Safety Requirements for Portable Sealed Secondary Cells, and for Batteries Made from Them, for Use in Portable Applications—Part 2: Lithium Systems, IEC 62133-2:2017, Feb. 2017, Amendment 1, 2021. [Online]. Available: <https://webstore.iec.ch/en/publication/32662>. [Accessed: Oct. 2, 2026].

[30] UL Standards & Engagement, Batteries for Use in Light Electric Vehicle (LEV) Applications, ANSI/CAN/UL/ULC 2271, 3rd ed., Sep. 14, 2023. [Online]. Available: <https://www.shopulstandards.com/ProductDetail.aspx?productId=UL2271>. [Accessed: Oct. 2, 2026].

[31] YAGEO Corporation, "Current Sensor—Low TCR Automotive Grade, PE_L Series," product specification, ver. 13, Aug. 7, 2024. [Online]. Available: https://yageogroup.com/content/Resource%20Library/Datasheet/PYU-PE_521_ROHS_L.pdf. [Accessed: Oct. 2, 2026].

[32] Texas Instruments, "TLV755P 500-mA, Low-IQ, Small-Size, Low-Dropout Regulator," datasheet, Rev. D. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tlv755p.pdf>. [Accessed: Oct. 2, 2026].