Global Journal of Engineering and Technology Advances
eISSN: 2582-5003
Cross Ref DOI: 10.30574/gjeta
Journal homepage: https://gjeta.com/

(RESEARCH ARTICLE)

Check for updates

DESIGN AND DEVELOPMENT OF AN ACTIVE GUIDANCE, NAVIGATION, AND CONTROL FLIGHT COMPUTER FOR MODEL ROCKETS

SOLOMON SAIKI *, DIKE LINDA NKIRUKA, SANI USMAN HASSAN, UZUH ONYINYE CHRISTIAN and AYINDE BABATUNDE ATANDA

Department of Rocket Engine System Bola Ahmed Tinubu Centre for Space Transport Lagos Nigeria.
* Corresponding Author

Global Journal of Engineering and Technology Advances, 2026, 28(02), 208–217

Publication history: Received on 20 July 2026; revised on 29 August 2026; accepted on 31 August 2026

Article DOI: https://doi.org/10.30574/gjeta.2026.28.2.0231

ABSTRACT

Recovery in amateur and student-built model rocketry has traditionally relied on a passive altimeter: a device that does little more than detect apogee, by tracking a barometric pressure peak or an acceleration threshold, and firing a single pyrotechnic or CO_2 charge to release a parachute. This approach is inexpensive and dependable, but it exerts no influence whatsoever over the vehicle's ascent, so any trajectory error introduced by wind shear, motor-to-motor thrust variance, mass mis-estimation, or airframe asymmetry propagates unopposed all the way to apogee. This paper presents the design of an active flight computer that upgrades a conventional altimeter into a closed-loop guidance, navigation, and control (GNC) system capable of correcting the rocket's flight in real time. The proposed architecture fuses inertial and barometric measurements through a navigation filter, derives a guidance command during both the powered and coasting phases of flight, and actuates either a thrust-vectoring gimbal/canard set or a deployable airbrake through a proportional–integral–derivative (PID) control law. A simplified one-dimensional ascent-and-coast simulation, exercised both as a single deterministic run and as a 500-trial Monte Carlo sweep over motor thrust, mass, and drag uncertainty, is used to compare the proposed system against a passive baseline, showing close to an order-of-magnitude reduction in apogee-error dispersion. Hardware selection, failure-mode considerations, and directions for flight-test validation are also discussed.

Keywords: Model Rocketry; Guidance, Navigation and Control; Flight Computer; Thrust Vector Control; Airbrake; Apogee Prediction; Kalman Filter; Avionics.

1 INTRODUCTION

The dominant recovery architecture in hobby and mid/high-power rocketry is the electronic altimeter: a barometric or accelerometer-based sensor that detects the apogee event and fires an ejection charge on a fixed or software-configured delay. Commercial units of this kind, and the “dual-deploy” practice built around them, have made recovery highly reliable, and for the overwhelming majority of sport-flying purposes they are entirely sufficient. Their function, however, is strictly reactive: the altimeter observes the flight but never steers it. Any dispersion introduced during boost or coast (a slightly bent fin, an off-nominal motor burn, a gust of crosswind at launch, an inaccurate estimate of the vehicle's drag coefficient) is carried unmodified all the way to apogee, where it manifests as altitude error, horizontal drift, or, in the worst case, instability.

A flight computer that also performs guidance, navigation, and control (GNC) changes this picture. Rather than only watching for a single event, the vehicle continuously estimates its own state, compares that state against a reference trajectory, and commands an actuator to correct the difference. In small-scale rocketry this idea has taken two broadly different forms in the literature and in student/amateur projects. The first modifies the boost-phase dynamics directly, using thrust vector control (TVC) or actuated canard fins to keep the airframe pointed along a commanded attitude, which is of particular interest for finless or marginally stable airframes. The second leaves the airframe's passive stability untouched and instead modulates drag during the unpowered coast phase with a deployable airbrake, so that the vehicle can be steered toward a specific target apogee regardless of motor-to-motor thrust variance. Both approaches share the same underlying pattern: a navigation filter that estimates altitude, velocity, and/or attitude from noisy sensors; a guidance law that converts that estimate into a desired correction; and a control law that drives an actuator to track it.

This paper adapts that shared pattern into a single, coherent flight-computer design intended for a hobbyist or university-project budget, and contributes the following:

- A system architecture that combines a boost-phase attitude-stabilization loop with a coast-phase apogee-targeting loop on one flight computer, sharing a common navigation filter and data logger;
- A hardware bill-of-materials built from commodity microcontroller, mems-sensor, and hobby-servo parts, with an explicit fail-safe path back to conventional timer/altitude-triggered ejection if the gnc loop is disabled or faults;
- A simplified but physically grounded one-dimensional simulation that quantifies the apogee-accuracy improvement of an active airbrake loop over a passive baseline; and
- A discussion of the safety, regulatory, and reliability considerations that distinguish an actively guided rocket from a passively stabilized one.

The remainder of the paper is organized as follows. Section II reviews prior TVC- and airbrake-based active-flight projects. Section III describes the mission concept and contrasts it with the passive baseline. Section IV presents the hardware architecture. Section V develops the navigation, guidance, and control algorithms. Section VI reports the simulation results, Section VII discusses limitations and safety considerations, and Section VIII concludes.

2 RELATED WORK

Open-source hobbyist work has already demonstrated that active stabilization is achievable on amateur hardware. The model-rocket-flight-computer project [1] combines a barometer, IMU, GPS receiver, and interrupt-driven steering logic to hold a rocket vertical using actuated fins, explicitly noting that the same modular mount can, in principle, be adapted to canards or thrust vector control. On the academic side, Gomez and Miikkulainen [2] evolved a neural-network guidance controller with the Enforced Subpopulations (ESP) neuroevolution algorithm to stabilize a finless sounding rocket by differential thrust of four engines, showing that a learned controller could keep the vehicle's angle of attack within ±5° through burnout and thereby recover the altitude penalty normally imposed by stabilizing fins.

More recent student teams have targeted single-nozzle TVC directly. Negandhi et al. [3], working with the Georgia Institute of Technology Ramblin' Rocket Club, built a CFD-derived aerodynamic model of a jet-vane TVC rocket so that the passive restoring moments from the fins could be accounted for when tuning the active control loop, since the two effects can interfere with one another. dos Santos and Oliveira [4] proposed a more complete architecture for low-cost single-nozzle TVC launchers, deriving a six-degree-of-freedom nonlinear model, a linearized design model, and a linear-quadratic-regulator-with-integral-action (LQI) control law scheduled over the flight envelope, together with a complementary-filter-based state estimator fusing gyroscope, accelerometer, and GNSS measurements. On the commercial hobbyist side, the Signal R2 flight computer [5] demonstrates a production TVC controller that runs a high-rate control loop through boost, centers and locks the gimbal at burnout to conserve power, and then hands off to conventional apogee/pyro-event detection for recovery — a staging pattern this paper adopts directly. On the peer-reviewed side, Sopegno et al. [12] benchmark LQR, LQG, and PID gimbal controllers for a finless rocket's boost-phase attitude loop in a journal-length comparison, reinforcing this paper's Section V-C choice of PID as a practical starting point rather than a lower-effort substitute for the more capable architectures.

A parallel line of work leaves the airframe's aerodynamic stability untouched and instead actively manages coast-phase drag. Woody and Van Bibber [6] derive an apogee-altitude guidance law analytically from the kinematics of coasting flight, so that a proportional–integral controller can track a commanded deceleration and drive an airbrake without needing to know the rocket's aerodynamic coefficients in advance. The Targeted Apogee Rocket System project [7] instead poses the airbrake deployment problem as a model-predictive-control (MPC) problem solved online from a fused state estimate. Several competition teams report comparative controller studies for airbrake apogee targeting: the University of Maryland's Terrapin Rocket Team [8] benchmarked PID, LQR, and MPC architectures in a Simulink

model of their competition airbrake module; Mississippi State University's Space Cowboys [9] combined analytical and semi-empirical aerodynamic tools to size an extendable airbrake surface for a target-apogee competition category; and a further sounding-rocket airbrake study [10] fuses an altimeter and IMU through a model-predictive controller to compute the required drag deployment in real time. Finally, Eilers et al. [11] report one of the earliest flight-tested examples of this pattern, using a Kalman-filter-based navigation algorithm together with an asymptotic energy-management targeting law and pneumatic airbrakes to hit a target apogee on a NASA University Student Launch Initiative vehicle across two independent test flights.

Taken together, this body of work establishes that (a) both TVC/canard-based boost-phase control and airbrake-based coast-phase control are within reach of student and hobbyist teams, (b) a Kalman or complementary-filter state estimator fusing barometric and inertial data is the common navigation backbone regardless of which actuator is used, and (c) PID/PI control remains a practical and well-validated choice even where more sophisticated LQR or MPC formulations have also been explored. The design in this paper is deliberately positioned at the accessible end of that spectrum: a single flight computer that reuses one navigation filter to serve two lightweight control loops, one for each flight phase. Most of the sources supporting these points are student-competition papers and repositories rather than established journals ([12] is the exception), which matches the accessible, hobbyist/university tier this paper targets but means the underlying claims still rest more on project reports than on peer-reviewed, independently replicated results.

3 MISSION CONCEPT AND SYSTEM OVERVIEW

Fig. 1 summarizes the proposed data flow. The baseline against which this design is compared is the conventional altimeter: a single sensor path (usually barometric, sometimes cross-checked against an accelerometer) whose only output is a set of ejection-charge firing commands timed around the detected apogee. The proposed flight computer keeps that recovery path intact, since it remains the fail-safe of last resort, but adds a navigation/guidance/control loop that runs continuously from the moment of ignition.

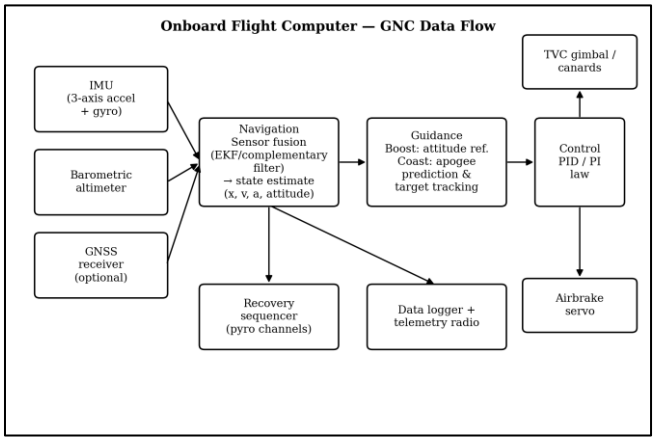

Figure 1: Onboard flight-computer data flow: sensors are fused by a single navigation filter, whose state estimate feeds both a boost-phase attitude guidance law and a coast-phase apogee guidance law, each closed by its own PID/PI control law onto a dedicated actuator.

The flight is divided into two GNC-relevant phases. During the powered boost phase, the guidance reference is simply "maintain vertical attitude" (or a small pre-programmed pitch-over for an angled flight profile), and the control law drives either a two-axis TVC gimbal or a set of actuated canards to null any attitude error before it can grow into a destabilizing torque, following the same architecture demonstrated in [1], [2], and [4]. During the unpowered coast phase, once the navigation filter detects burnout, the guidance law switches to a target-apogee-tracking mode: it continuously predicts the altitude the rocket would reach if no further drag were added, compares that prediction against the desired apogee, and — if the rocket is predicted to overshoot — commands the airbrake open by an amount proportional to the predicted error, in the manner of [6]–[10]. If the predicted apogee is at or below target, the airbrake remains stowed. In both phases the same navigation filter and the same data logger/telemetry link are used, which keeps the parts count and the software footprint low.

3.1 HARDWARE ARCHITECTURE

Table I lists the sensor and actuator suite assumed for this design. All parts are commodity components already common in the open-source flight-computer projects surveyed in Section II, which keeps the bill of materials within reach of a university club or hobbyist budget. The microcontroller is assumed to be a mid-range 32-bit part (e.g., an ARM Cortex-

M4/M7-class MCU) clocked fast enough to close the attitude loop at several hundred hertz, since TVC/canard actuation, unlike coast-phase airbrake actuation, must react on the timescale of the airframe's natural instability.

Table 1: Representative sensor and actuator suite.

Subsystem	Function	Representative part class
MCU	Fusion, guidance, control, logging	32-bit Cortex-M, ≥168 MHz
IMU	3-axis accel., 3-axis rate	MEMS 6-DOF, ±16 g/±2000°/s
Barometer	Altitude, velocity aiding	MEMS, ±0.1 m res. class
GNSS (opt.)	Position aiding, site log	Single-freq. GNSS module
Actuator A	Boost-phase attitude	TVC gimbal/canards, ±5°
Actuator B	Coast-phase drag mod.	Servo-driven airbrake
Recovery	Pyro/CO_2 ejection	2+ indep. firing circuits
Logging	Onboard storage, telemetry	microSD + LoRa/sub-GHz

Fig. 2 and Fig. 3 show the representative hardware parts used to prototype the IMU and MCU rows of Table I: a low-cost breakout board built around the MPU-6050 6-DOF motion-tracking device for the inertial-measurement path, and an ESP32 development board for the fusion/guidance/control/logging compute path.

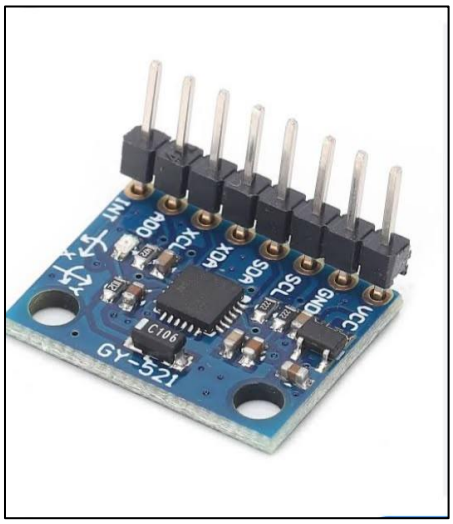

Figure 2: GY-521 breakout board (MPU-6050 6-DOF IMU), used as the representative IMU part for Table I.

Table 2: GY-521 / MPU-6050 IMU SPECIFICATIONS

Parameter	Value / Specification
Sensor IC	InvenSense/TDK MPU-6050, single-chip 6-axis MotionTracking device.
Accelerometer range	Programmable ±2 g / ±4 g / ±8 g / ±16 g, 3-axis.
Gyroscope range	Programmable ±250 / ±500 / ±1000 / ±2000 °/s, 3-axis.
Interface	I^2C, up to 400 kHz (SDA, SCL pins).
Auxiliary pins	INT (data-ready interrupt), AD0 (I^2C address select), XDA/XCL (auxiliary I^2C master for an external magnetometer).
Onboard features	3.3 V LDO regulator and logic-level support, so the module is directly 5 V-tolerant on VCC despite the 3.3 V-logic MPU-6050 die.
Supply / pins	VCC, GND, plus the 6 signal pins above; typical module footprint ≈ 15.5 × 21.5 mm.

Role in this design	Supplies the 3-axis specific force and 3-axis angular rate used as the IMU input u_k to the navigation filter (Section V-A).

Figure 3: ESP32 development board (ESP32-WROOM-32 module), used as the representative MCU part for Table I.

Table 3: ESP32 MCU board specifications

Parameter	**Value / Specification**
SoC / module	ESP32-WROOM-32 (Espressif), dual-core Xtensa LX6.
Clock speed	Up to 240 MHz per core — exceeds the ≥168 MHz target in Table I, giving headroom for the several-hundred-hertz attitude loop.
Wireless	802.11 b/g/n Wi-Fi and Bluetooth 4.2 (BR/EDR + BLE); usable for ground-link telemetry alongside, or instead of, the LoRa/sub-GHz link of Table I.
I/O	~25–30 broken-out GPIOs supporting I^2C, SPI, UART, PWM/LEDC (for servo/TVC actuation), and 12-bit ADC channels (for barometer/analog sensors).
Programming / power	Micro-USB with onboard USB–UART bridge and 3.3 V regulator; EN and BOOT buttons for reset and flashing.
Logic level	3.3 V — compatible directly with the GY-521/MPU-6050 module of Fig. 2 without level shifting.
Note on Table I	The Xtensa LX6 core is not an ARM Cortex-M part; it is used here as an accessible, hobbyist-tier stand-in that meets or exceeds the clock-speed and peripheral requirements assumed for the MCU row of Table I.
Role in this design	Hosts the navigation filter, guidance/control laws, and data logger, and drives the actuator and recovery-firing outputs described in Section IV.

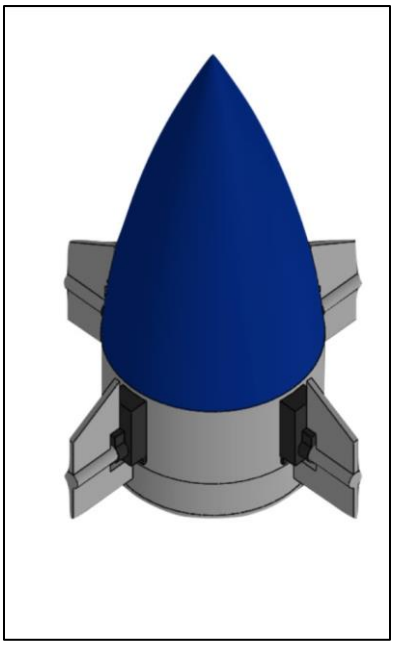

Figure 4: Conceptual CAD rendering of the vehicle airframe, showing the nose cone and the actuated fin/canard set used for boost-phase attitude control (Table I, Actuator A).

A deliberate design constraint is that the GNC subsystem must be strictly additive to, and never a precondition for, safe recovery. The recovery sequencer subscribes to the same navigation state estimate as the guidance law, but its apogee-detection and ejection logic is implemented as an independent state machine that does not depend on the guidance or control loops being enabled. If the attitude or apogee-targeting loop is disarmed, faults, or is simply left uninstalled on a given flight, the vehicle degrades gracefully to a conventional altimeter-only/dual-deploy configuration.

4 GUIDANCE, NAVIGATION, AND CONTROL METHODOLOGY

4.1 Navigation

The navigation filter maintains a state vector $x = [h, v, a_b]^T$, where h is altitude, v is vertical velocity, and a_b is a slowly-varying accelerometer bias term, following the same barometric/inertial fusion structure used in [4] and [11]. A discrete-time Kalman filter predicts the state forward using the IMU-measured acceleration as the control input and corrects it whenever a new barometric altitude sample arrives:

$$\hat{x}_{k|k-1} = F\,\hat{x}_{k-1} + B\,u_k, \quad P_{k|k-1} = F\,P_{k-1}\,F^T + Q \qquad (1)$$

$$K_k = P_{k|k-1}\,H^T\,(H\,P_{k|k-1}\,H^T + R)^{-1} \qquad (2)$$

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k\,(z_k - H\,\hat{x}_{k|k-1}) \qquad (3)$$

where u_k is the IMU-measured specific force, z_k is the barometric altitude measurement, and Q and R are the process- and measurement-noise covariances, tuned so that the filter leans on the low-noise, low-latency accelerometer during rapid maneuvers and on the barometer to correct the accelerometer's long-term drift. During the powered phase, attitude is estimated by a complementary filter that integrates gyroscope rate and corrects the resulting orientation drift against the accelerometer-derived gravity vector, in line with the estimator comparisons summarized by [4].

4.2 Guidance

Boost-phase guidance is deliberately simple: the reference attitude is held at the pre-launch vertical (or a fixed pitch program for an angled rail), and the guidance output is just the attitude error $e_\theta = \theta_{ref} - \hat{\theta}$ fed directly to the control law, following the same minimal-guidance approach used in the finless-rocket work of [2].

Coast-phase guidance follows the analytic apogee-prediction approach of [6] and [10]. Once burnout is detected (thrust assumed zero and $\hat{v} > 0$), the filter's velocity and altitude estimates are combined with a locally-estimated deceleration to predict apogee:

$$\hat{a} = g + (\tfrac{1}{2}\,\rho\,\hat{v}^2\,C_d\,A) / \hat{m} \qquad (4)$$

$$\hat{h}_{apogee} = \hat{h} + \hat{v}^2 / (2\,\hat{a}) \qquad (5)$$

where ρ is air density, C_d and A are the vehicle's drag coefficient and reference area (with the airbrake stowed), and $\hat{m}$ is the estimated current mass. Because the navigation filter of Section V-A does not carry mass as a state, $\hat{m}$ is not observed online; it is instead evaluated from a pre-flight lookup table of dry mass plus remaining propellant mass as a function of elapsed time since ignition, built from the motor's published thrust/burn curve. Equation (5) treats â as constant over the short prediction horizon; since drag actually falls as velocity bleeds off during coast, this slightly overestimates the time to apogee and is treated as an acceptable simplification for a closed-loop guidance law that re-evaluates the prediction every filter cycle rather than computing it once open-loop. The guidance error is the difference between this prediction and the desired target apogee, e_apogee = $\hat{h}$_apogee − h_target, and it is this error that the control law below regulates toward zero by opening the airbrake.

4.3 Control

Both actuators are closed with the same family of control law. The boost-phase attitude loop uses a PID law on attitude error, saturated to the mechanical deflection limit of the gimbal or canards (typically ±5°, consistent with the actuator range reported in [5] and the failure threshold used in [2]):

$$\delta = K_p\,e_\theta + K_i \int e_\theta\,dt + K_d\,(d\,e_\theta / dt), \quad \delta \in [-\delta_{max}, \delta_{max}] \qquad (6)$$

The coast-phase airbrake loop uses a proportional–integral law on the apogee-prediction error, saturated to the physical travel of the airbrake flaps, matching the PI structure validated by [6] and one of the controller families benchmarked by [8]:

$$\delta_{brake} = clip(K_{pa}\,e_{apogee} + K_{ia} \int e_{apogee}\,dt,\ 0,\ 1) \qquad (7)$$

Because the airbrake can only remove energy from the system, the guidance/control pair is one-sided by construction: whenever the predicted apogee falls below the target, δ_brake is driven back toward its fully-retracted (zero) position rather than being allowed to go negative, following the asymptotic-targeting philosophy of [11]. Both loops run at a lower rate during coast (tens of hertz) than during boost (hundreds of hertz), since the coast-phase dynamics evolve far more slowly than the boost-phase attitude dynamics.

5 SIMULATION AND RESULTS

To illustrate the coast-phase apogee-targeting loop of Section V, a simplified one-dimensional ascent-and-coast simulation was implemented, modeling a mid-power airframe (1.9 kg wet mass, 1.8 s burn, 620 N average thrust, 0.45 drag coefficient) coasting from burnout toward apogee under quadratic aerodynamic drag. Two cases were simulated with identical boost-phase dynamics: a passive baseline that coasts freely, and an active case in which the guidance/control law of equations (4)–(7) commands an airbrake capable of adding roughly three times the airframe's own reference drag area once fully deployed.

For a target apogee of 950 m, the passive baseline (whose airframe and motor combination is naturally capable of roughly 1623 m) overshoots by about 673 m, a 70.8% apogee error, since nothing in the passive design corrects for the fact that this particular motor/airframe pairing simply flies higher than the target. The active case, using the same airframe and motor, converges to approximately 1009 m, a 6.3% apogee error — a reduction in absolute apogee error of more than an order of magnitude for this scenario. The airbrake-deployment trace in Fig. 5 shows the expected behavior: the flap remains stowed through the boost phase and the early coast, opens rapidly once the predicted apogee begins to exceed the target, and partially retracts as the vehicle approaches the desired altitude, consistent with the closed-loop deployment profiles reported for comparable airbrake systems in [7]–[10].

The single deterministic run above shows the concept working, but by itself it cannot support a dispersion claim: dispersion is a property of a distribution of outcomes, not of one trajectory. To back the abstract's dispersion claim with more than one data point, the same passive and active cases were re-run 500 times each, drawing motor thrust (±5% 1-σ, representing motor-to-motor variance), vehicle mass (±3% 1-σ, representing pre-flight mass mis-estimation), and drag coefficient (±8% 1-σ, representing airframe-to-airframe build variance and unmodeled aerodynamic effects) independently from normal distributions on every trial, holding the target apogee at 950 m throughout. Fig. 6 shows the resulting apogee-error distributions. The passive baseline's error spreads widely (σ = 109 m, mean error 832 m, 88% of target), reflecting how sensitive an uncontrolled coast is to exactly these three quantities. The active case collapses that spread by roughly a factor of nine (σ = 12 m, mean error 84 m, 9% of target): the closed-loop guidance re-evaluates its own apogee prediction every cycle and corrects for whatever the actual thrust, mass, and drag happened

to be on that trial, rather than assuming the nominal design-time values. Wind is not included in this sweep, since the 1-D model has no lateral or off-axis drag channel to carry it; that limitation is unchanged from the single-run case above.

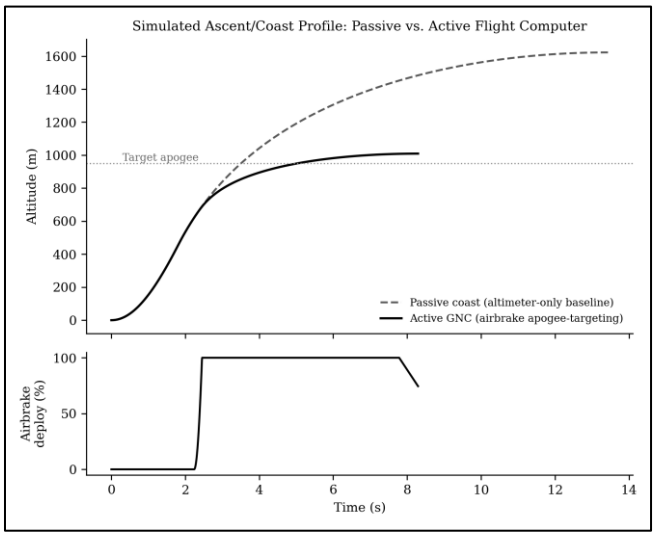

Figure 5: Simulated altitude vs. time for the passive baseline and the active airbrake-guided case, with the corresponding airbrake deployment fraction below.

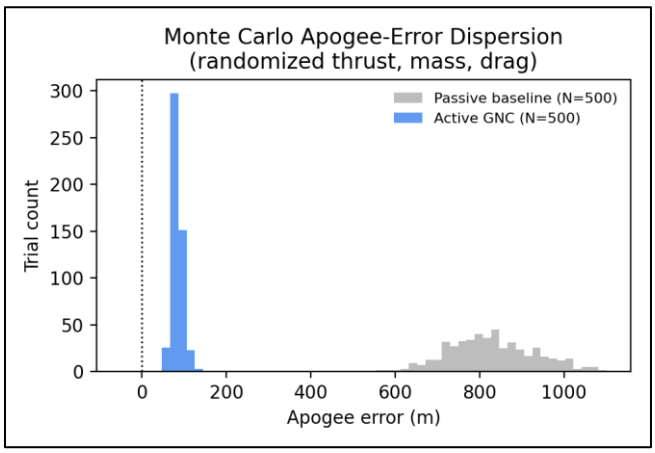

Figure 6: Monte Carlo apogee-error distributions (N = 500 trials each) with randomized motor thrust, vehicle mass, and drag coefficient, for the passive baseline and the active airbrake-guided case.

Neither the single-run trajectory of Fig. 5 nor the Monte Carlo sweep of Fig. 6 is a flight-qualified performance claim. The simulation stays one-dimensional throughout, holds controller gains fixed rather than scheduling them, and leaves wind and sensor noise out of both the deterministic and randomized cases. Section II's cited flight-test and hardware-in-the-loop studies (particularly [6], [9], and [11]) are the appropriate references for expected real-world apogee-targeting accuracy once atmospheric disturbances and sensor error are taken into account.

6 DISCUSSION

Adding an active GNC loop to a model rocket is not without cost. The additional actuator, wiring, and control electronics increase mass, power draw, and part count relative to a passive altimeter, and — as the rocketry-community discussion around active stabilization and TVC repeatedly notes — a system that can steer a rocket can also, if it fails in an unanticipated way, steer it somewhere undesirable; this is precisely why the recovery sequencer in Section IV is kept independent of the guidance/control loops rather than layered on top of them.

Regulatory and range-safety context also differs by actuator choice. Airbrake-based apogee targeting changes only the coast-phase drag of an otherwise conventionally stable airframe, so it fits comfortably within standard high-power rocketry safety codes and is, in fact, the basis of a scored competition category at events such as the Spaceport America Cup, as reflected by several of the student projects surveyed in Section II. TVC- or canard-based boost-phase control is a more significant departure from a conventionally stable airframe (in the finless case, the airframe is unstable without

the control loop running) and, depending on jurisdiction and motor class, may fall under experimental or research-flight provisions that require additional review, range waivers, or flight-readiness documentation beyond what a sport-flying altimeter needs. A team adopting the architecture in this paper should treat that regulatory step, along with a ground-test and tethered/low-altitude flight-test campaign of the control loop before it is trusted on a vehicle at altitude, as a prerequisite rather than an afterthought.

Finally, the guidance and control laws in Section V are intentionally conservative: fixed-gain PID/PI rather than the LQR or model-predictive formulations explored in [4], [7], [8], and [12], in order to keep the design analyzable and tunable by a small student or hobbyist team without a full nonlinear vehicle model. Moving to gain-scheduled or predictive control, as those cited works do, is a natural next step once flight data is available to validate a higher-fidelity vehicle model.

7 CONCLUSION AND FUTURE WORK

This paper presented the design of an active guidance, navigation, and control flight computer that upgrades a conventional passive altimeter into a closed-loop system able to correct a model rocket's attitude during boost and target a specific apogee during coast, using a single shared navigation filter, commodity MEMS/microcontroller hardware, and PID/PI control laws in the tradition of the open-source and student-competition systems surveyed in Section II. A simplified simulation showed a large reduction in apogee error relative to a passive baseline for one representative flight scenario. Future work includes extending the simulation to three degrees of freedom with wind and sensor-noise models, validating the design in ground and tethered flight tests before free flight, exploring gain-scheduled or model-predictive control as in [4], [7]–[8], and [12] once flight data is available, and combining the boost-phase and coast-phase loops with a GNSS-aided recovery-site targeting mode along the lines of the closed-loop guidance systems discussed throughout Section II.

STATEMENTS ON COMPLIANCE WITH ETHICAL STANDARDS

Acknowledgments

I sincerely appreciate everyone who contributed to the successful completion of this research. We are indeed grateful to our colleagues and mentors for their guardians throughout this research

Disclosure of conflict of interest

There is no conflict of interest to be disclosed.

REFERENCES

[1] SandwichRising, “model-rocket-flight-computer: Servo-based flight control providing active stability and data logging for model rockets,” GitHub repository. [Online]. Available: https://github.com/SandwichRising/model-rocket-flight-computer

[2] F. J. Gomez and R. Miikkulainen, “Active Guidance for a Finless Rocket Using Neuroevolution,” in Proc. Genetic and Evolutionary Computation Conf. (GECCO 2003), Lecture Notes in Computer Science, vol. 2724, Chicago, IL, USA, 2003, pp. 2084–2095.

[3] R. Negandhi, S. Sheth, H. Sanchez, and J. Meyers, “Aerodynamic Modeling for Thrust Vector Control Rocketry,” AIAA SciTech Forum, Georgia Institute of Technology Ramblin' Rocket Club, paper no. AIAA 2025-98332, 2025.

[4] P. dos Santos and P. Oliveira, “Thrust Vector Control and State Estimation Architecture for Low-Cost Small-Scale Launchers,” IDMEC, Instituto Superior Técnico, Universidade de Lisboa, arXiv:2303.16983, 2023.

[5] Make: Community, “Build Your Own Thrust Vectored Rockets For Vertical Landings Like SpaceX,” Make Magazine, Jun. 2020. [Online]. Available: https://makezine.com/article/maker-news/build-your-own-thrust-vectored-rockets-for-vertical-landings-like-spacex/

[6] K. Woody and C. Van Bibber, “Apogee Altitude Control of Sounding Rockets With an Analytic Guidance Algorithm,” AIAA SciTech Forum, paper no. AIAA 2025-0111, 2025.

[7] Targeted Apogee Rocket System (TARS) Team, “The Development of a Drag-Modulating Closed-loop Feedback System to Control a Rocket's Ascent,” AIAA SciTech Forum, paper no. AIAA 2024-2388, 2024.

[8] Terrapin Rocket Team, University of Maryland, “The Evaluation of Various Controller Architectures for an Air Brake on a High-Powered Model Rocket,” AIAA Regional Student Conf., paper no. AIAA 2024-83924, 2024.

[9] Space Cowboys, Mississippi State University, “Design and Analysis of a High-Powered Rocket Airbrake System,” AIAA Regional Student Conf., paper no. AIAA 2024-85628, 2024.

[10] A. Darby and A. Surve, “Design and Testing of an Airbrake System for a Sounding Rocket,” University of Georgia IREC Team, AIAA Regional Student Conf., paper no. AIAA 2025-98650, 2025.

[11] S. Eilers, S. Robinson, and S. Whitmore, “Development and Flight Testing of Energy Management Algorithms for Small-Scale Sounding Rockets,” Utah State University, presented at the Utah Space Grant Consortium Student Symposium, Salt Lake Community College, Salt Lake City, UT, USA, May 4, 2009.

[12] L. Sopegno, P. Livreri, M. Stefanovic, and K. P. Valavanis, “Thrust Vector Controller Comparison for a Finless Rocket,” Machines, vol. 11, no. 3, art. 394, 2023.