Scarlet Gimbaled Rocket Design and Launch Report

Krish Mishra*, Ilan Villard*, Cathleen Bolger*, Syona Gupta*, William Castro*, Ethan Koh*, Isaac Wengier*, Michael J. Wallace*, Jackson D'Ambrosio*, Jinhe Li*
Georgia Institute of Technology, Atlanta, Georgia, 30332, United States

Thrust vector control (TVC) is commonly used in rocketry to enhance vehicle stability. Active stabilization can be achieved through various TVC methods. One method of active stabilization involves gimbaling the rocket motor, which allows for controlled attitude adjustments by pivoting the motor to generate pitch and yaw moments. The GNC Project in the Ramblin' Rocket Club aims to design, build, and launch a low-powered rocket stabilized using a gimbaled mechanical guidance system. The gimbal consists of two servos connected via linkages to a ball-and-socket joint that allows for the actuation of the motor about the pitch and yaw axes. The gimbal receives servo input from a custom flight computer which uses an Inertial Measurement Unit (IMU) and microcontroller in a closed-loop system. All components were built in-house, including the 3D-printed internal components and custom PCB design. A simulation was also created to help tune the controller and analyze projected flight performance. This rocket builds upon a previous GNC gimbaled rocket, incorporating multiple iterative design improvements. The project establishes a framework for testing TVC systems, laying the groundwork for future large-scale TVC implementations. This paper outlines the redesign, software development, simulations, manufacturing, and testing of a gimbaled motor rocket.

I. Nomenclature

ψ_o	=	Output Yaw Angle from PID
θ_o	=	Output Pitch Angle from PID
ϕ	=	Roll of Rocket
ψ_s	=	Output Yaw Angle of the Servo
θ_s	=	Output Pitch Angle of the Servo
$\angle$	=	General Angle
T	=	Thrust
τ	=	Tuned Thrust
$\angle_a$	=	Adjusted Angle

II. Introduction and Background

High and low-power model rockets traditionally rely on fins for passive stabilization. However, this method has limitations; fins cannot actively respond to atmospheric conditions, nor can they offer precise attitude adjustments to the rocket. For example, a well-documented phenomenon for passively stabilized rockets is 'windcocking', or rockets turning into the wind as they launch [1]. Alternatively, rockets can be actively stabilized. Methods to do so take various forms; one is gimbaling motors. The GNC project within the Ramblin' Rocket Club at Georgia Tech has explored gimbaling for thrust vector control (TVC) for two years. During Spring 2024, GNC launched two low-powered rockets called *Gru* and *Vector*. Using the lessons learned from those designs, the team launched *Scarlet* in early 2025 as the next iteration in the process to achieving TVC via a gimbaled system.

This paper outlines the process of designing *Scarlet*, focusing on design requirements derived from shortcomings in *Gru* and *Vector*'s systems. It also covers the simulation, manufacturing, and testing processes that shaped the final rocket design.

*Student Member Undergraduate Category
AIAA numbers in order of names: *1603269, 1603560, 1748708, 1810830, 1810831, 1760396, 1810025, 1810967, 1810128, 1811110*

III. Rocket/Design Improvements Overview

Several design improvements were implemented in *Scarlet* based on previous launches. The primary concern was reducing dry mass, as *Gru* and *Vector* had low thrust-to-weight ratios that resulted in limited flight data. To address this, 3D-printing techniques were used to minimize the weight of nonstructural components, and lighter commercial off-the-shelf (COTS) components, such as the flight computer battery, were selected. A target mass of 2 lbs. was set to ensure the rocket's weight remained above the motor's axial thrust at all times. For the current project (*Scarlet*), the TVC mount was redesigned to provide a greater angular range of motion. Additionally, the flight computer was upgraded from the previous iteration of the breadboard setup to a fully integrated, custom-designed PCB, significantly improving reliability, ease of integration, and software testing. One of the most critical shortcomings in the previous iteration was the lack of a proper TVC simulation for controller tuning before launch. This time, a Simulink model was developed, allowing for pre-launch tuning and performance analysis.

Scarlet features a structural design optimized for weight, strength, and cost, as shown in Fig. 1. From top to bottom, the rocket houses the recovery system, which includes a lightweight, 3D-printed nosecone made from PETG filament, a recovery bulkhead, a shock cord, and a parachute. Below the recovery system is the PCB mount, with the TVC mount positioned towards the base. This design allowed for weight reduction and relative ease in integration, eliminating the need for additional subsystems in the rocket. Additionally, a custom launchpad was developed to ensure clearance for the TVC mount above the pad.

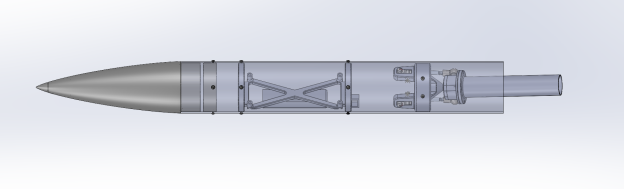


Fig. 1 Overall rocket structure of *Scarlet*

Scarlet was launched with a AeroTech G12ST motor, chosen due to its long burn time of about 12.7 seconds and consistent thrust. It was also the most powerful motor with these characteristics that would qualify as a low-powered rocket. This allowed *Scarlet* to be launched at nearby launch sites which proved to be crucial in allowing two additional test flights.

IV. Rocket Subsystems

A. TVC Assembly

The TVC can gimbal the G12ST motor ±20 degrees in any direction, actively stabilizing *Scarlet* and correcting deviations from its intended flight path. This marks the second major iteration of a TVC system developed and flown by the GNC project team, incorporating significant improvements from the previous design.

1. TVC Engineering Goals and Improvements

After analyzing footage and flight data from the *Gru* and *Vector* 2024 launches, increasing the TVC mount's range of motion was identified as a primary design priority to enhance control authority and overall flight performance. The previous mount, closely derived from [2], had a limited angular range of motion of ±2.86° in yaw and ±4.60° in pitch. This constraint stemmed from the gimbal design, which was restricted by the 3" Blue Tube airframe, preventing a greater range of motion.

Additional design requirements for the new TVC mount included minimizing mechanical play, or looseness in connection, achieving full 360-degree actuation, and utilizing primarily 3D-printed and COTS components.

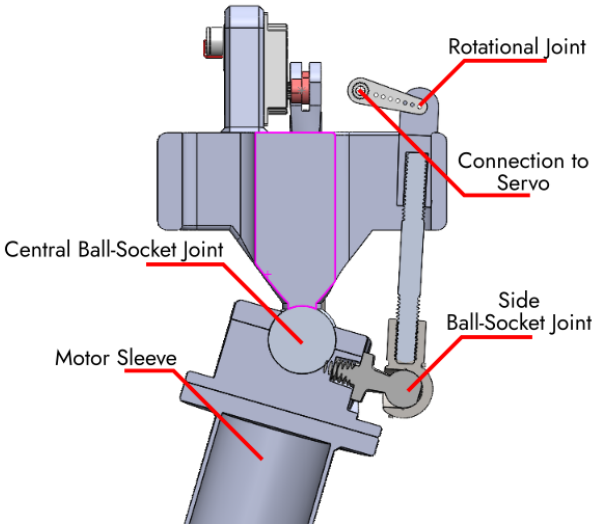


Fig. 2 Front cross-section of TVC mount

2. Mount Design

To increase the range of motion of the TVC, the gimbal was replaced with a unique articulation system previously prototyped in [3]. As depicted in Fig. 2, this system has a central ball-and-socket joint that allows the motor to pivot around the fixed top plate.

The servos pitch and yaw the G12 rocket motor through a rotational joint connected to a linkage bar and a side ball-socket joint. See the cross-sectional view below of the assembly in Fig. 2. The rotational joint was formed from the COTS servo horn, an M2 screw, and an FDM linkage adapter as shown in Fig. 4. This moment is then transferred through a COTS linkage rod to a side ball-socket joint. The two side ball-socket joints allow each axis to rotate and move independently of the other, allowing for uncoupled controls.

Scarlet utilized two KST X08 digital micro servos to gimbal the motor. With an output torque of 2.8 kg-cm at 8.4 volts and an angular speed of 0.09 seconds per sixty degrees, these servos provided high performance and structural reliability as they held up through all of the launches. This was an improvement from the MG90S servos used previously that had a larger footprint and greater mass that would have hindered the new design. The smaller footprint of the X08 servos allowed for intentional vertical positioning in line with the pitch and yaw axes within the constraining 3" body tube to achieve precision movements.

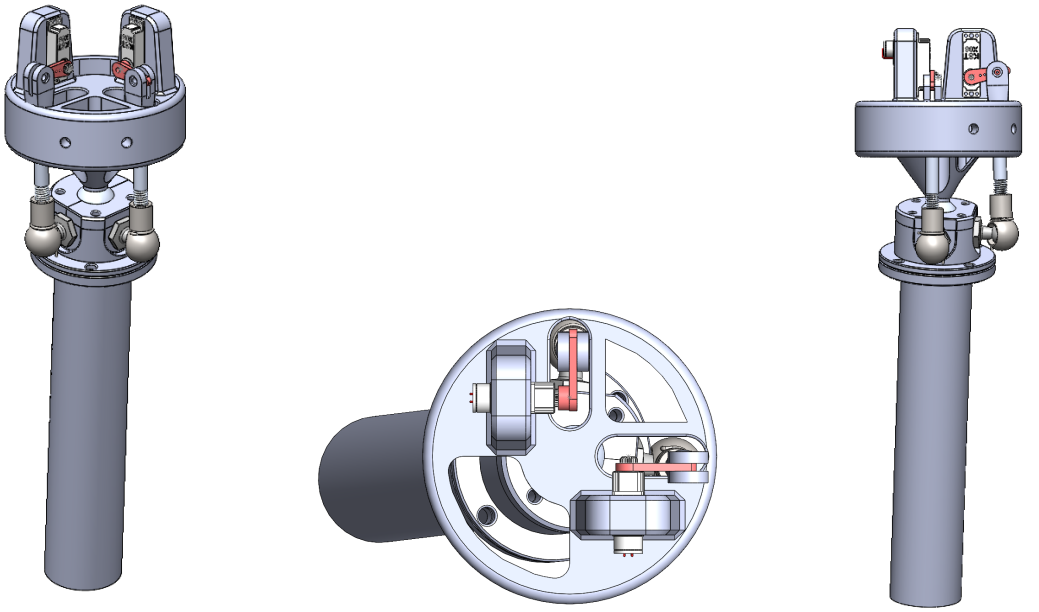
Since *Scarlet* used an Aerotech G12ST motor with an expendable single-use casing system, the motor sleeve shown in Fig. 4 was purposefully designed to be single use and easily removable by unscrewing 4 M4 screws. The motor sleeves are 3D-printed from PETG.

3. Additive Manufacturing Techniques and Assembly

Figure 4 shows an exploded view of the various COTS and 3D-printed components on the TVC mount. Of these, the top plate that housed the servos, the motor sleeve, and the linkage adapter were all FDM printed from PETG. PETG was chosen because of its higher heat deflection temperature compared to comparable materials such as PLA, while balancing ease of printing and cost. The center socket and center socket lock plates for the ball-socket joint were printed on a Form 4 MSLA 3D-printer with standard Formlabs White V4.1 resin. Resin was chosen for this ball and socket joint due to its smoothness and fine layer height, which was essential to allow the ball to rotate and glide while minimizing friction.

The Top Plate of the TVC was split in two parts to be FDM 3D-printed, before being glued together. This was chosen as there wasn't an ideal flat surface for the mount to be supported on during printing. After the first test flight, it was discovered that the portion of the top plate holding the servos was a weak point. Design tweaks were made to increase the contact surface, as well as using modifiers in the 3D slicer to increase the wall thickness and infill for those weak spots accordingly. Additionally, modifiers were used to add more material for the ball connection joint to screw into.

Overall, the TVC mount performed well and will be refined in the future. Some areas tof improvement include reducing the play in the mount when the servos are engaged, refining the assembly procedure, and redesigning it to be more inherently stable.



(a) TVC Isometric View **(b) TVC Top View** **(c) TVC Right View**

Fig. 3 TVC Mount Views

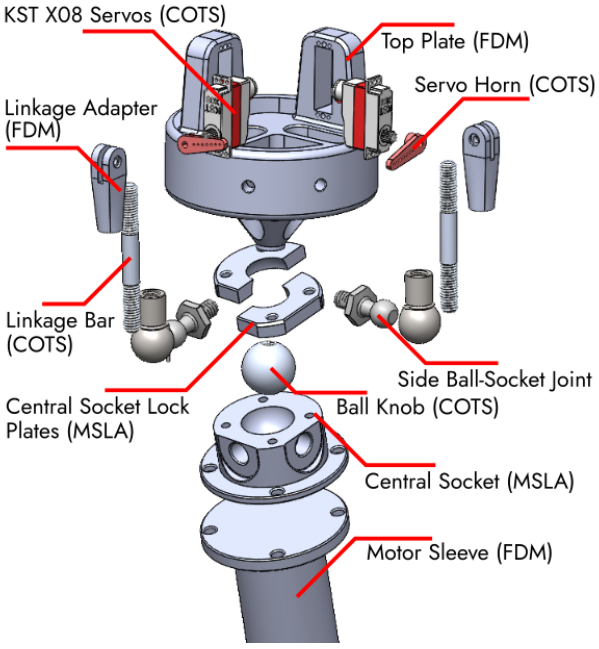
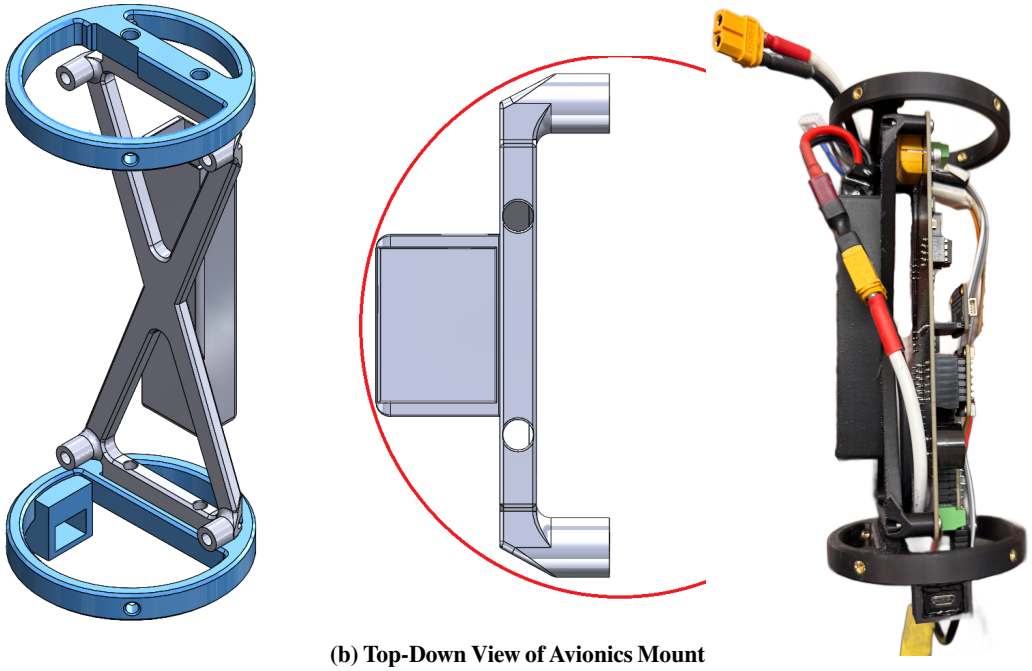


Fig. 4 Exploded View of TVC Mount

B. Avionics Bay

The Avionics Bay is composed of three assembled 3D-printed components made of PETG that hold the *Scarlet* flight computer, the battery, and a micro-USB port to interface with the flight computer when it is integrated inside the rocket. The largest physical constraint for the avionics bay was the 3" diameter body tube. The 6” long by 2.8” wide flight computer had little clearance within the 3” diameter rocket, especially when considering clearances for the barometer and IMU breakout boards, the Teensy 4.1 micro controller on the top face of the board, and the XT60 power plug located on the bottom face. Cutouts through the bulkheads on either side were needed for pryo channel and servo wires to run through and interface with the flight computer. A cutout for the XT-60 power cable was also included.



(a) Isometric View of Avionics Bay **(b) Top-Down View of Avionics Mount with 3” Body Tube** **(c) Integrated Avionics Bay**

Fig. 5 Avionics Bay CAD Views

The two bulkheads (blue colored) in Fig. 5a include slots for #4-40 heat-set inserts, through which the avionics bay was mounted on the body tube. #4-40 heat-set inserts were used throughout the design to increase the strength of the connection and allow screws to be repeatedly inserted and removed. This was especially important for integration, as the entire avionics bay was designed to be fully integrated before final assembly with the body tube.

The tight clearances in fitting a 2.8” wide flight computer into a 3” diameter rocket tube with sufficient clearances for all components meant that a unique X-shaped design was needed to avoid conflicts with components on the flight computer. This gave maximum clearance to all the components and connectors. The edges of the X’s were chamfered to allow the mount to fit into the circular tube, as shown in Fig. 5b.

C. Simulink Simulation

Simulink was used with the purpose of simulating rocket trajectories to tune the gain values of the PID controller. Within the control loop, the input gimbal angles were converted into a thrust vector in Cartesian coordinates. This thrust vector was then transformed from the rocket body frame to the world frame via a rotation matrix and passed into a Simscape 6-DOF joint to model rotational and translational dynamics. From this block, the angle errors (measured from the vertical) in both the yaw and pitch angles were passed into PID blocks. The control outputs from these PID blocks were then adjusted according to an imported AeroTech G12 motor thrust curve. This is due to the non-constant nature of the rocket’s thrust, where certain PID gain values may work well for a certain thrust value but fail when used with a lower thrust force. The exact method for this adjustment is covered in more detail in subsection E, Flight Software.

The newly adjusted values constitute the new input gimbal angles, thus closing the control loop. Drag and wind

were simulated using the drag equation and were based on the simplifying assumption that the active stabilization of the rocket would result in near constant orientation and a negligible change in reference area with coefficient of drag.

Monte Carlo simulations were then run using the Simulink model with variations in parameters such as wind speed and direction, rocket mass, and the positions of the center of gravity and center of pressure. Trajectories and error measurements were plotted from these simulations, which allowed for the validation of a certain set of PID gain values over various circumstances and launch conditions. After tuning the gain values over multiple simulation runs, the values of 0.7, 0.4, and 0.1 for the proportional, integral, and derivative gains were found to give the best results. These values yielded the set of trajectories shown in Figure 6. With this set of values, the simulation results showed the rocket achieving an apogee of ≈ 35 meters and landing no more than 5 meters from its initial launch site in gentle wind conditions. The angle errors are also shown in Figure 7 demonstrating the quick response of the PID controller. The simulation shows that with the aforementioned gain values, the rocket should be able to minimize angle errors to less than one degree in size after two seconds, thus demonstrating an ability to achieve active attitude stabilization.

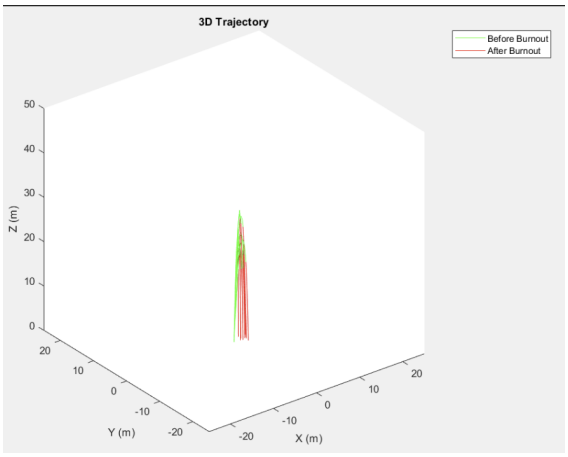


Fig. 6 Monte Carlo Trajectories at ideal PID values

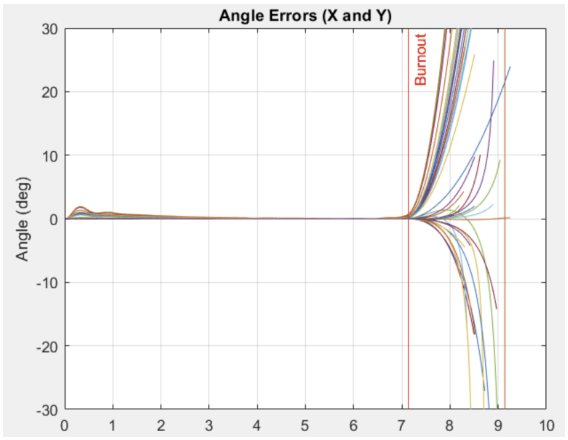


Fig. 7 Simulated Rocket Pitch and Yaw Angles

D. OpenRocket Simulation

An OpenRocket simulation was used to gauge the thrust-to-weight ratio. There were various challenges with simulating *Scarlet*, which had no fins, in OpenRocket. Mainly, it was impossible to accurately simulate ascent, even with ideal conditions without wind. OpenRocket consistently introduced instability in the rocket, ending up with preliminary tumbling at apogee, before motor burnout. Even so, OpenRocket was still useful to determine initial rocket architecture. For example, it was used to track the total mass of the rocket and allowed for simulation of the thrust to weight ratio for

the entire flight duration. The team ensured the weight of the rocket never exceeded thrust, which would ensure that the thrust vector always pointed in the same direction as total vehicle acceleration. To ensure robustness, the thrust of the rocket was graphed against the weight, assuming a constant 30 degree actuation of the gimbal, beyond its max range. This was a worse case scenario; the results are shown below in Figure 8, including the masses from flight 1 and flights 2 and 3. It can be seen that the sufficient mass was achieved for flights 2 and 3.

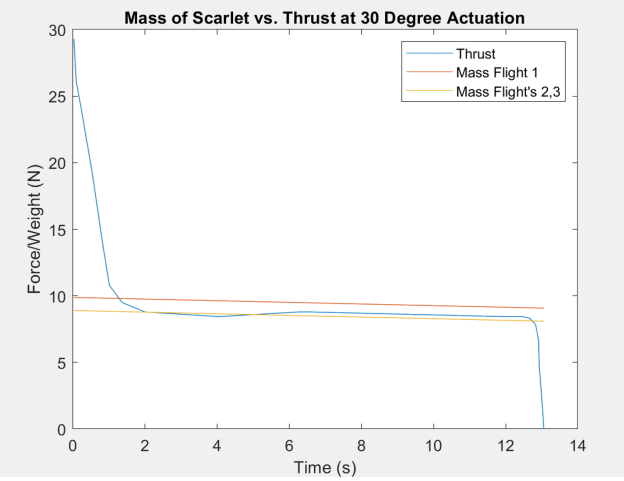


Fig. 8 Graph of Simulated Masses vs. Thrust

E. Flight Software

The flight software for *Scarlet* was written using Platform.IO for the Teensy 4.1 microcontroller platform. The codebase is object-oriented and designed such that the "user-facing" state machine is easy to understand, with all stabilization and recovery functions abstracted. The main file of the codebase contains a state machine for all the different phases of the rocket's flight.

An instance method of the rocket object reports its state to this state machine, which runs the appropriate detection and control functions. In the rocket's Armed State, for example, the state machine calls upon the launch detection instance method, which will update the rocket's state to ascent from within the rocket object itself when launch is detected. This removes the need to handle state updates from the state machine itself, improving readability. As for stabilization, the state machine calls the update instance method, specifying if active control is enabled, which is only the case in the Ascent State. This method runs at a frequency of around 500 Hz but only updates the PID controllers and servo outputs when a new sensor event is available from the BNO085 IMU, which operates at around 250 Hz in stabilized mode. The IMU has built-in sensor fusion and provides an orientation quaternion, as well as accelerations along all three axes. This quaternion is accepted as truth, but it is not in the desired frame of reference. To address this, the rocket object provides a tare instance method that saves the current orientation quaternion as the "tare quaternion". Multiplying the inverse of the tare quaternion by the current orientation quaternion gives a quaternion relative to the tare quaternion. Taring the rocket while it is pointed vertically therefore gives a quaternion relative to the vertical, which is then converted into Tait-Bryan angles to feed into the double PID controller. Two additional problems must be addressed, however. The Tait-Bryan angles used in the PID calculations are in a global reference frame, meaning that, were the rocket to roll 180 degrees, the outputs would be doing the exact opposite of stabilizing the rocket. This problem is addressed by making the servo angles a combination of the pitch PID output multiplied by a trigonometric function of the roll and the yaw PID output multiplied by a trigonometric function of the roll. Specifically, the equations are show as Eq. (1) and Eq. (2):

$$\theta_s = \theta_o \cdot cos(\phi) - \psi_o \cdot sin(\phi) \tag{1}$$

$$\psi_s = \psi_o \cdot cos(\phi) + \theta_o \cdot sin(\phi) \tag{2}$$

The second problem is the fact that the thrust of the motor is not constant throughout the flight. There are several ways to address this, such as dynamically adjusting PID gains or switching to a nonlinear controller altogether, but the

simplest solution found was to tune the PIDs for the average thrust (referred to as tuned thrust) of the motor and use a trigonometric relationship to transform the PID servo outputs to produce the same moment given the thrust of the motor at that specific point in time. During initialization, the rocket object reads a thrust curve file from the onboard SD card and creates a map of certain times since ignition and their corresponding thrust. A method is then able to provide the thrust of the motor at a given time after launch by linearly interpolating between the closest points at times before and after the given time. To perform the moment conservation calculation, the following equation (3) is used:

$$\angle_a = arcsin((\frac{\tau}{T} \cdot sin(\angle)) \tag{3}$$

This will result in larger output angles for the servos at lower thrust levels without requiring any adjustments to the controller architecture. In summary, the pitch and yaw PID outputs are first converted into pitch and yaw servo angles based on the rocket's roll using Eq. (1) and Eq. (2), and these pitch and yaw servo angles are then individually adjusted to produce a consistent moment regardless of the current motor thrust using equation (3).

The final aspect of the flight software is the various event detection instance methods used in the state machine as detailed by the diamonds in figure 1. The first of these methods is launch detection, which detects if the rocket's acceleration along the Z-axis (defined as the axis along the rocket's length) is greater than $-15m/s^2$. Since this acceleration value includes gravity, the minimum acceleration caused by the motor for launch to be detected is around $5m/s^2$, which is very conservative. The second method is unsafe conditions detection. This method is called in the ascent state and disables active control if the rocket encounters more than 45 degrees of deviation from vertical in the pitch or yaw direction. The third method is apogee detection. There are several ways to detect apogee using either the barometer or IMU acceleration values. The method used relies on the barometer, with the update method updating a maximum altitude value and saving the timestamp whenever the barometer reports a higher altitude than previously recorded. The apogee detection method will trigger the Chute State if the maximum altitude was more than half a second ago. Lastly, the landing detection method is triggered when the rocket's altitude is less than its starting altitude on the elevated pad. While this detector will not go off if the rocket lands at a higher elevation than it launched, such as on a hill, its only purpose is to terminate data logging for the flight, and as such, it was decided to not be worth the effort to implement a more sophisticated landing detection algorithm. However, if a more sophisticated algorithm were to be implemented, it would be easiest to detect landing by calculating the standard deviation of a set amount of altitude readings and checking if it is less than a predetermined constant.

F. Flight Computer

The schematics and layout for *Scarlet*'s flight computer were created with Altium Designer. To achieve all desired capabilities, a Teensy 4.1 serves as the microcontroller, with the BNO085 and BMP390 breakout boards as the chosen IMU and barometer, respectively. Four electrical output channels were added to trigger pyrotechnic devices for recovery, and two 3-pin servo headers were added to control the pitch and yaw servos. The board contains three voltage-mode control buck converters. These converters step a twelve-voltage input into two five-voltage power planes supplied to the Teensy 4.1 and pyrotechnic channels and a 7.4-voltage power plane routed to the servo motor channels. To ensure safety, a connector is used to arm the pyrotechnic channels, preventing devices from being inadvertently triggered. An LED is used to communicate when the pyrotechnic channels are active. A buzzer is also included to give feedback based on flight states when the rocket is on the pad.

G. Recovery

To recover the rocket, a single deployment system was utilized consisting of a 3D-printed recovery bulkhead, 18" nylon parachute, and Kevlar shock cord.

The recovery bulkhead served to house the black powder used for the ejection charges to generate the bulkhead force to separate the parachute from the nosecone. The black powder is contained within two charge wells shown in Fig. 9 with a 0.2" retaining wall to ensure its durability. Initial calculations based on a body tube length of 20", inner diameter of the body tube of 3.9", and desired pressure within the range of 8-15 psi led to the design of the charge wells to accommodate black powder of 1 gram in each pocket. This design was later finalized during a successful ground test utilizing 1 gram of black powder, the amount used during the first launch. However, during the second and third launches, the amount of black powder filled in each charge well was reduced to 0.5 grams because of weight-reducing modifications made to the nosecone. At the bottom of each of the charge wells, there is a hole that allows the E-match to pass through, as shown in Fig. 9. The bulkhead also has two holes with a diameter of 0.4" to attach the shock cord to

the bulkhead.

A big consideration when designing the recovery system was weight reduction. The 18" nylon parachute was chosen due to its compact size and ability to support the estimated weight of the rocket of 2 lbs. To protect the parachute from potential damage from ejection charges, recovery wadding was used due to its fire-resistant properties and low mass. Within the bulkhead design, pocketing around the radial edge of the bulkhead was used to reduce weight which is shown in Fig. 9. The pockets were designed to ensure mass reduction without compromising the structural integrity of the bulkhead by adding them to the outer edge. The weight of the system was also reduced by using a bowline knot instead of a stainless steel U-bolt to secure the bulkhead to the shock cord and proved to be effective during ground testing.

Fig. 9 Final recovery bulkhead (Top isometric view).

H. Launch Pad

A custom, multi-use launchpad was designed and manufactured to provide greater flexibility in how the rocket was mounted on the launchpad. The main components of the design were a T-slot rail, a 3D-printed PETG support, and an aluminum base. The rail was five feet long to provide enough time for the rocket to reach a sufficient velocity in the initial moments of launch. The support was designed for the body tube to rest and provide ease of access to the ignition wire. It also allowed part of the TVC assembly to extrude from the body through the support, keeping it elevated from the base. The base was necessary to provide a flat surface to integrate the rail and stand, and it was spaced to center the rocket above the base. The fully integrated launchpad is shown in Fig. 10. The base was secured to the ground via stakes to keep the assembly fixed. Movement in the launchpad was further constrained by ratchet straps attached via eyebolts at the top of the rail, which limited any swaying of the rail due to wind or forces from the rocket during launch.

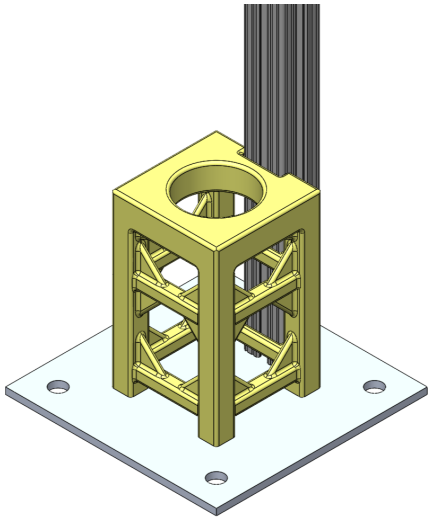


Fig. 10 Integrated launchpad base.

The support went through multiple iterations before arriving at the final design. The initial support had sufficient

stability to support the rocket, but during integration testing, the leg cracked due to torsion from screwing into the threaded holes. To alleviate this, the threads were made wider and deeper to allow more clearance with the screws. Additionally, the support was 3D-printed with increased gyroid-patterned infill and more walls to provide better structural integrity. This final iteration was sufficient to withstand the stress of the screws and provide stable support.

V. Results and Discussion

Scarlet was launched three times, once at Mill Springs Academy in Alpharetta, GA, and twice at Ball Ground, GA. These launches occured two weeks apart, allowing for iterative design improvements between flights. At both sites, wind conditions were relatively mild, with gusts reaching approximately 10 mph.

During the first launch, the gimbal was able to react to an initial pitch in the rocket. It is speculated that, due to the relatively heavy wind during launch, the motor struggled to generate sufficient corrective force. Approximately one second into the burn, the system detected unsafe conditions, triggering a TVC lock. A software defect caused the TVC to lock at its maximum 20-degree deflection, further exacerbating the deviation from the intended trajectory. Flight data (including pitch, yaw, and PID response) can be seen in Figure 11. As observed in the figure, the PID can respond to the pitch and yaw input from the IMU; however, there is a delayed response, which is likely due to the low gain values. Additionally, the PID values dropped to zero at the end of the graph, indicating that the system had entered an unsafe state. Notably, the controller was not designed to compensate for strong lateral forces from wind, suggesting that the mechanical guidance system was unable to counteract significant side forces. Based on these findings, minimizing wind conditions became a priority for subsequent launches.

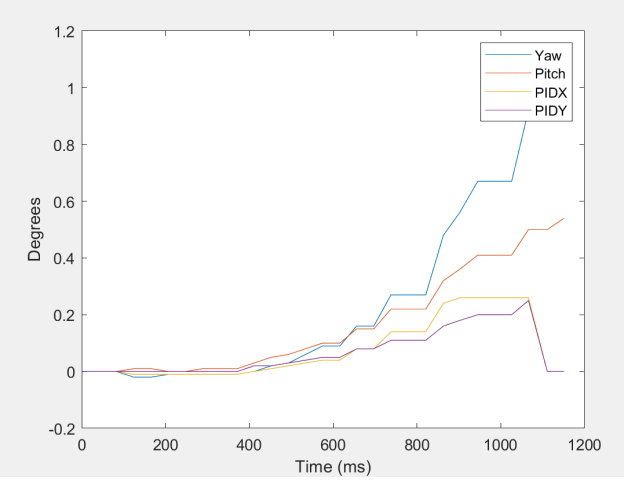
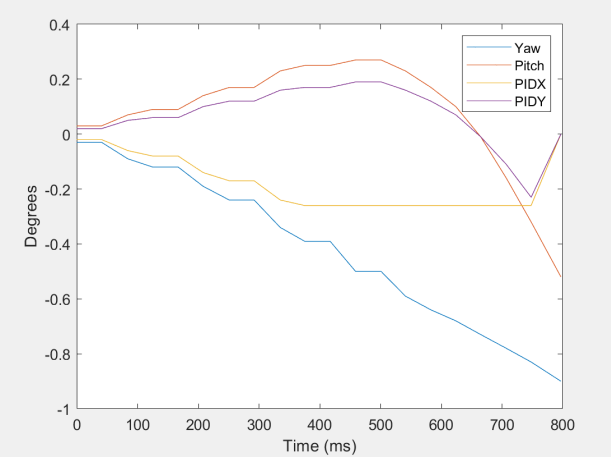


Fig. 11 Data from Flight 1 of *Scarlet*

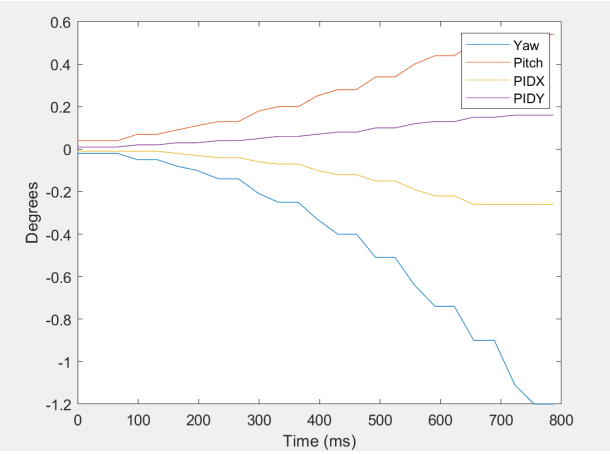
For the second launch, various design changes, as discussed in previous sections, were implemented. The controller gains were increased, with a higher proportional (P) gain to enhance responsiveness to pitch and yaw deviations. However, the increased gains resulted in excessive corrections, leading to oscillations and an inability to stabilize before reaching unsafe conditions. This is due to two reasons: a slow TVC response time and an actuation that was physically too large. The data from the second launch can be seen in Figure 12a. With the larger P gain, the PID responds more rapidly to attitude changes in the rocket. Note that the PIDX, corresponding to the pitch values, reached a software-implemented maximum in order to prevent the servos from overheating. These lessons were taken into account in the third launch.

For the third launch, the P-value was significantly reduced compared to the first launch. It was determined that in low-wind conditions, the rocket would naturally correct over time, making a lower P-value preferable to avoid overcorrection. Additionally, the derivative (D) gain was omitted, as previous testing showed that including it caused the controller to move erratically. Despite these adjustments, the controller failed to adequately respond to the rocket's pitch deviations, and the TVC mount was unable to generate sufficient corrective moments. Figure 12b illustrates the undesirable response of the PID values to the attitude of the rocket.

After launch, it was found that the IMU was sampling at only 25 HZ. This was due to an oversight in the code in which the significantly lower sampling rate of the BMP390 limited the sampling rate of the IMU. The result was a



(a) Data from Flight 2 of *Scarlet*



(b) Data from flight 3 of *Scarlet*

Fig. 12 Launch data from day 2 of testing

failure to get a functioning derivative term on the PID controller and an inability of the TVC to respond to changes in pitch and yaw; therefore, attitude adjustment was not possible. This provides a further explanation of the discontinuous nature of Figures 11, 12a, and 12b. The data in these figures come from an in-flight SD card that writes every loop. The sampling rate deficiency may also provide a reason for the over-correction during the second flight. It can be concluded that with a faster TVC mount response time, the rocket may not have over-corrected since attitude adjustment would have occurred more rapidly.

VI. Conclusion

The GNC project, under the Ramblin' Rocket Club at Georgia Tech, iterated upon a previous gimbaled rocket design and successfully flight-tested the new design. The rocket used a barometer (BMP390), IMU (BNO085), and microcontroller (Teensy 4.1) mounted on a Printed Circuit Board (PCB) to control two servos on a TVC mount for motor actuation. This system was tested three times, and there were several successes and lessons taken from these launches.

A common issue with the three tests was the lack of a solid sampling rate; however, it is possible to tune the controller and gain useful qualitative information. For the second and third launches, *Scarlet* was able to over-correct or under-correct based on the PID values used. However, since the rocket was unable to rapidly update attitude changes, it was unable to achieve active stabilization.

The GNC project is aiming to further tune the controller for active stabilization. Launching with a higher sampling rate of the IMU is essential for successful stabilization, along with reducing the range of the TVC, which could allow for increased accuracy of output angles. Moving forward with these changes, GNC hopes to continually test and improve the gimbaled system.

Acknowledgments

GNC would like to acknowledge Jorge Blanco for his help in offering a launch site for testing.

References

[1] Benson, T., "Model Rocket Weather Cocking," , n.d.

[2] Barnard, J., "Thrust Vectoring Mount - Build Signal R2," , 2018.

[3] Aerospace, O., "How to Thrust Vector Control | The Thrust Vectoring Mount," , 2019.