Team Jarts
Flight Readiness Report

April 12, 2013

"THE RIPPER II"

Brett Foster, Joe Hintz, Eric Logisz, Cameron Schulz

Contents

Executive Summary

The objective of the 2013 Wisconsin Space Grant Consortium collegiate rocket competition was to construct a rocket to achieve an apogee of 3,000 feet and recover the rocket safely in flyable condition.

The rocket was designed to reach an apogee of 3000 feet. The main feature of the rocket includes passively controlling the drag to control the altitude reached. This is done by initial design of the airframe and fins. The other feature is deploying a drogue parachute on the ascent of the rocket in order to stop the rocket at 3000 feet. The rocket was designed to achieve an apogee greater than 3000 feet so that the deployment of a drogue parachute can stop the rocket at 3000 feet. Fiberglass tubing makes up the rocket due to its high strength and durability. Extra support was designed into the rocket to ensure that zippering does not occur. The rocket design included resulting in stable rocket based on the center of pressure and center of gravity.

The recovery system of the rocket involves a dual deployment system. A drogue parachute is to be deployed at 2900 feet to stop the rocket at 3000 feet and control the rockets descent until the main parachute is deployed. The main parachute is then deployed at 700 feet and controls the rocket descent until landing on the ground. There is also a second set of ejection charges that will deploy the drogue parachute at apogee and the main parachute at 500 feet if the main charges do not deploy them. These ejection charges are fired by the MARSA4 and PerfectFlite StratoLogger altimeters. There is also a motor ejection charge backup based on a time delay to ensure the deployment of a parachute.

Analysis of the predicted flight was done in both OpenRocket and MATLAB. OpenRocket predicted an apogee of 3074 feet and MATLAB predicted an apogee of 3092 feet without simulating the deployment of the drogue parachute. A predicted flight including the deployment of the drogue parachute resulted in an apogee of 2990 feet.

A variety of testing was done to confirm the design and function of the rocket. This testing includes ground testing for the deployment of the parachutes, altimeter testing for the confirmation of the programming and function and two test launches. During the first test launch, the parachutes were deployed at maximum acceleration. No damage occurred on the rocket confirming the structural design of the rocket. The second test launch resulted in a successful flight reaching an apogee of 3272 feet. The drogue parachute did not deploy during this launch. This allowed for a more accurate coefficient of drag to be calculated. This then lead to more accurate predictions of the achieved apogee with the deployment of the drogue parachute. Figure 1 shows the rocket on a component by component basis.

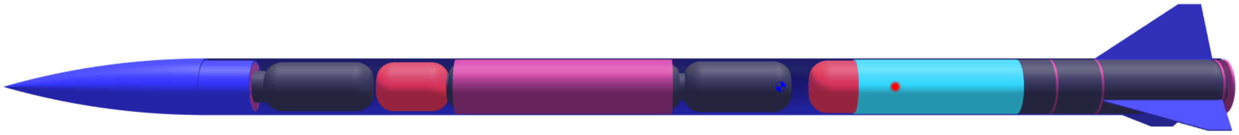

Figure 1: Rocket Components (OpenRocket Model)

Objective and Constraints

Scope

The objective of this project is to design a one-stage, high powered rocket that has an apogee of 3000 feet. The rocket must be recovered in flyable condition safely. The desired flight is shown in Figure 2.

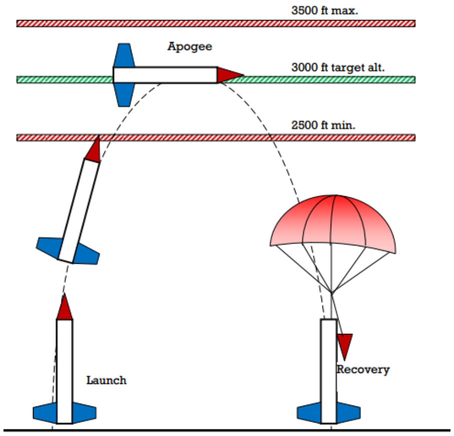

Figure 2: Desired Rocket Flight

Source: http://www.uwgb.edu/wsgc/collegiate_rocket_launch/2013/RocketCompetition13Handbook.pdf (4/3/2013)

Application

The Wisconsin Space Grant Consortium (WSGC) collegiate rocket competition allows students to apply engineering skills learned in class directly to a project. This results in a great experience in the aerospace field. WSGC intends to further the education and experience of students in order to benefit the future in Space and Aerospace science, design and technology in Wisconsin. This allows students to develop their skills for the future. High powered rocketry involves a lot of concepts learned in an engineering education. Concepts such as aerodynamics, statics, dynamics, numerical analysis, etc. can be applied.

Design Requirements

The following are design parameters for the 2013 WSGC Collegiate Rocket Competition:

- Altitude of at least 2500 feet but not to exceed 3500 feet reached
- Rocket recovered safely and in flyable condition
- Electronics used to deploy a parachute for your recovery system as well as have an engine based backup deployment
- Structural components and materials obtained from a reputable high powered rocket vendor or provide engineering analysis supporting their suitability
- Cesaroni I540 used as the motor
- Raven III altimeter carried on board as the competition altimeter
- Maximum body tube diameter of 4 inches
- Maximum overall length in launch configuration of 72 inches
- Maximum weight in launch configuration less motor of 7.5 pounds

Design

Summary of Design

A simple design for the rocket was chosen to eliminate as many unneeded risks as possible. The idea that many things can go wrong with a rocket was considered and therefore the design was chosen such that it added as little more risks. The drag force of the rocket was the main component used to control the apogee of the rocket. This is done both passively and actively. The rocket was designed such that the simulation of apogee is just over 3000 feet. Also, the design actively controls the altitude by releasing a parachute on the way up near 2900 feet the stop the rocket at 3000 feet.

Airframe Design

The overall length of the rocket is 67.75 inches with a body tube diameter of 3.129 inches. These overall dimensions determined by many factors. Factors that include the space inside the rocket for parachutes, electronics, shock cord and more, the resulting center of gravity and center of pressure of the rocket which determines stability and the approximate altitude the rocket will reach. The length, weight, and diameter of the rocket also meet the constraints given. The rocket also has three clipped delta fiberglass fins evenly spaced around the bottom of the rocket's airframe. The shape of the fin was chosen based on where the desired location for the center of pressure of the rocket should occur. The three fins are sufficient to maintain a stable flight with a center of pressure lower than the center of gravity. Not needing a fourth fin reduces the drag of the rocket. The fins are through the wall fins for sufficient strength and are attached to the motor mount tube that is secured by several centering rings spaced along the tube.

The rocket will be comprised of 5 sections. These sections include the nose cone, the upper tube, a coupler, the lower tube with a coupler attached, and an engine mount section. The rocket will break at the nose cone and right above the engine mount. The rest of the sections will be secured by pem nuts and screws. These sections are shown in Figure 3.

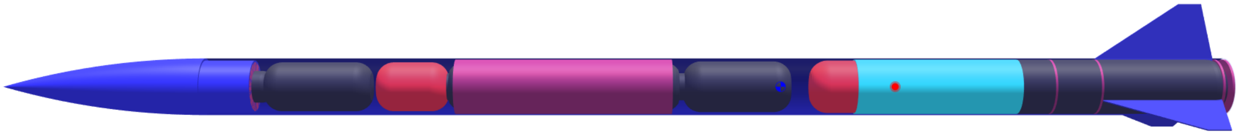

Figure 3: Rocket Components (OpenRocket Model)

Fiberglass tubing with a diameter of 3.129 inches makes up the main airframe of the rocket. Fiberglass was chosen due to its high strength as well as its low weight. Fiberglass also provides durability that other common rocket materials do not provide. With the plan of launching multiple times before the competition day, durability of the rocket is an important factor.

Along with the strength of fiberglass, extra design was done to reduce the chance of zippering. Zippering of the body tube becomes a big factor especially when launching a parachute on the ascent of the rocket flight. Besides the strength of the material, the rocket was designed with extra strength right above the engine mount where the drogue parachute will exit on the ascent of the rocket flight. A centering ring was added right above the engine mount in order to increase the surface area that the shock cord will rip down on. This will reduce the shearing force exerted by the impulse of the shock cord after the drogue parachute is released. An axial support tube was added to support this centering ring as it will only be attached to the engine mount tube and not the outer wall. A picture of this section is shown in Figure 4. In addition, the length of the shock cord is 15 feet in order to reduce the impulse force exerted.

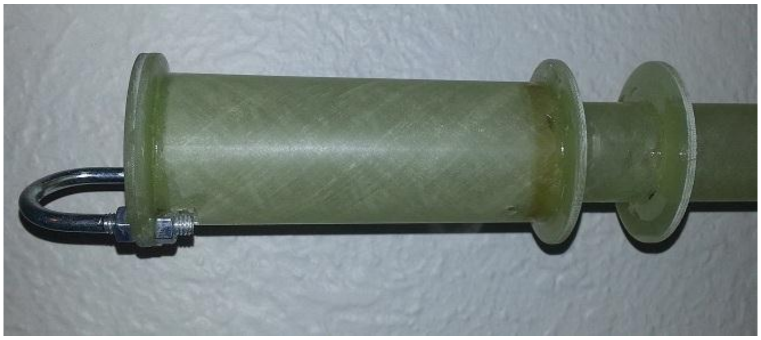

Figure 4: Zipperless Design Feature

Recovery System

In order to recover the rocket safely and in flyable condition, a dual deployment system will be used. The components used in the system include a 48 inch main parachute, a 24 inch drogue parachute, 30 feet of 1500 pound test Kevlar cord, lettuce and nylon shear pins. The 24 inch drogue parachute shrouds can be pulled tighter to decrease the drag force. The Kevlar cord was chosen due to its low weight but high strength properties as well as its fire resistance. Lettuce will not allow the parachutes to burn when the ejection charges are blown. Nylon shear pins will be used to attach the nose cone to the upper section tube and the engine mount section to the coupler attached to the lower section tube. These nylon shear pins will shear when an ejection charge is blown, causing the rocket to separate at the desired locations. Figure 5 shows the two parachutes and Kevlar cord.

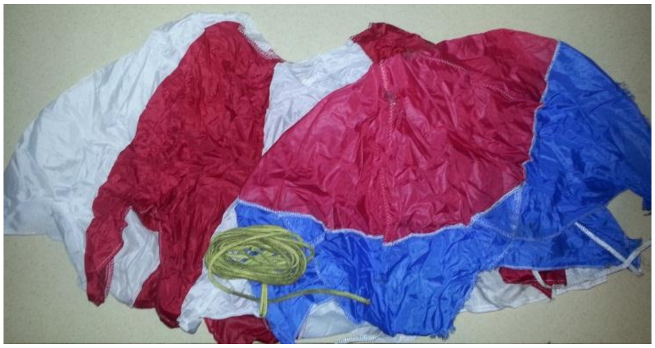

Figure 5: Parachutes and Shock Cord

The recovery system begins after the engine burnout occurs as well as the rocket coasting until 2900 feet. At 2900 feet an ejection charge will blow from an altimeter based trigger, deploying the drogue parachute from the break point right above the engine mount where extra support exists. The drogue is attached to 15 feet of 1500 pound test Kevlar cord. This drogue parachute acts as a break parachute in order to stop the rocket at 3000 feet. The rocket will then have a rapid, controlled descent under the control of the drogue parachute until 700 feet. At 700 feet above the ground, the main parachute will be deployed via another ejection charge from the same altimeter based trigger. The main parachute will exit the rocket from the upper section tube when the nose cone is blown off. A controlled descent will then occur until the rocket lands on the ground safely.

With the risk of parachutes not deploying, a second set of charges will be present, which are triggered from a separate altimeter. The first charge will be set to trigger at apogee in case the drogue parachute does not deploy on the way up at 2900 feet. The second charge will trigger at 500 feet in case the main parachute does not deploy at 700 feet during the rocket's descent. As well as having redundancy with two altimeters, a motor ejection charge based on a time delay will be used. This will ensure that a parachute will come out regardless of the functioning of the electronics. The built in redundancy as well as the backup motor ejection charge will ensure safe recovery of the rocket, resulting in a rocket returning in flyable condition.

Electronics Bay

The coupler section joining the upper and lower sections of the rocket is where the electronics are housed. The section consists of an 11.75 inch long coupling tube with a 2.5 inch fiberglass thrust ring epoxied around the coupling tube. This section is displayed in Figure 6.

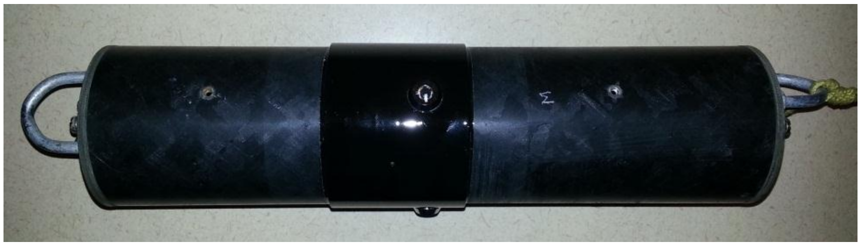

Figure 6: Electronics Bay

Each end of the electronics bay is capped with a bulkhead that has a U-bolt mounted on it as well as a bridge. The shock cord is attached to each of the U-bolts to connect the whole rocket. The bridges are used to connect the wires from the altimeters to the ejection charges. This is shown in Figure 7.

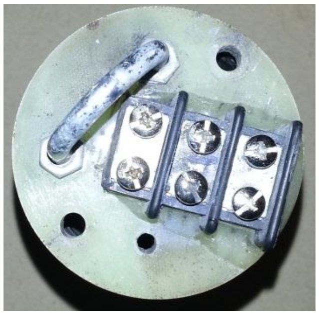

Figure 7: Bulkhead for Electronics Bay

Threaded rods are run through the whole length of the electronics bay to secure the bulkheads as well as provide a way to mount the altimeters. The altimeters are mounted on plywood sleds with standoffs and screws. These sleds also have metal tubing epoxied to them so that they can be slid onto the threaded rods in the electronics bay. For ease of assembly and access, two sleds were made to slide in from either end of the bay and meet in the middle. There are spacers in between these sleds such that key switches can be mounted. Three key switches are mounted to the outside of the electronics bay that will allow one to turn on each altimeter separately while the rocket is on the launch rail. This system ensures the altimeters are securely mounted as well as allows for ease of assembly and disassembly.

The primary altimeter used for this rocket is the MARSA4, a programmable parachute deployment system. This altimeter allows for 4 separate channels to trigger ejection charges. Some other features include field programmability, data analysis and diagnosis. The combination of these features allow for the descent of the rocket to be controlled as described in the recovery system section. The capability to launch the drogue parachute on the ascent is an important feature for the rocket's performance. The programmability of the altimeter along with the post flight data analysis ensures the most successful flight. The MARSA4 is shown in Figure 8.

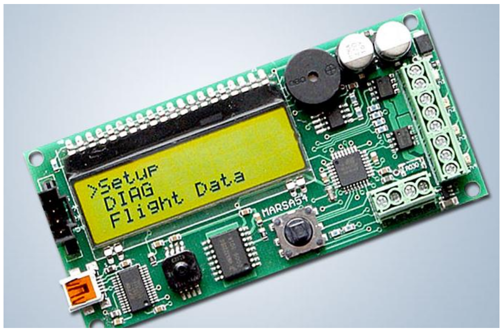

Figure 8: MARSA54 Parachute Deployment System
Source: http://www.rocketryplanet.com/content/view/3541/29/#axzz1KVksSdHZ (4/3/2013)

The second altimeter on board the rocket is the PerfectFlite StratoLogger Altimeter. This altimeter is capable of deploying the drogue parachute at apogee and the main parachute at an altitude ranging from 100 feet to 9,999 feet in 1 foot increments. This satisfies the conditions that this secondary altimeter needs to perform. This altimeter records the data from flights and will output altitude and velocity plots. Although this altimeter has lower accuracy than the MARSA4, it fulfills the requirements to serve as a redundant altimeter. The StratoLogger is shown in Figure 9.

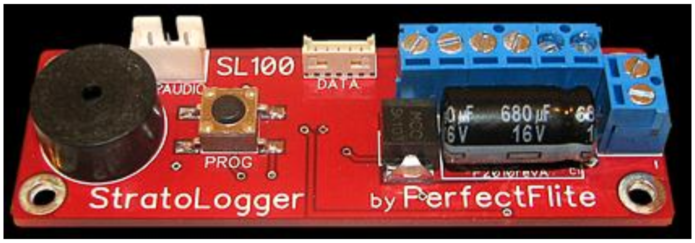

Figure 9: PerfectFlite StratoLogger
Source: http://www.perfectflite.com/sl100.html (4/3/2013)

Along with the two altimeters described above, a Raven III will be housed in the electronics bay. This altimeter will serve as the competition altimeter to record flight data. It will have the official data for the altitude of the rocket. This altimeter is housed in the same bay as the other electronics to ensure accurate, consistent data.

Center of Pressure & Center of Gravity

OpenRocket was used to determine the locations of the center of pressure and center of gravity. This program allows one to perform a component by component design with custom overrides for materials, weights, dimensions, etc… The properties of the purchased materials can then be updated into the model. The program also allows for ease of design for stability. As each component is added or changed the location of the center of pressure and center of gravity is updated. This analysis ensures that a stable rocket will be designed. The resulting calculations for the locations of the center of gravity and the center of pressure are shown in Figure 10. The center of pressure must be located more than 1 airframe diameters below the center of gravity to make the rocket stable. The design of the rocket resulted in a stability margin of 2.03. This margin is a little over stable. The result of this is not an optimal performance by the rocket in windy conditions. This would lead to a lower maximum altitude. Since the stability of the rocket flight is an important factor, this design is suitable. With the over design of the rocket reaching greater than 3000 feet, there is a factor built in for the rocket not reaching maximum altitude if the conditions are very windy.

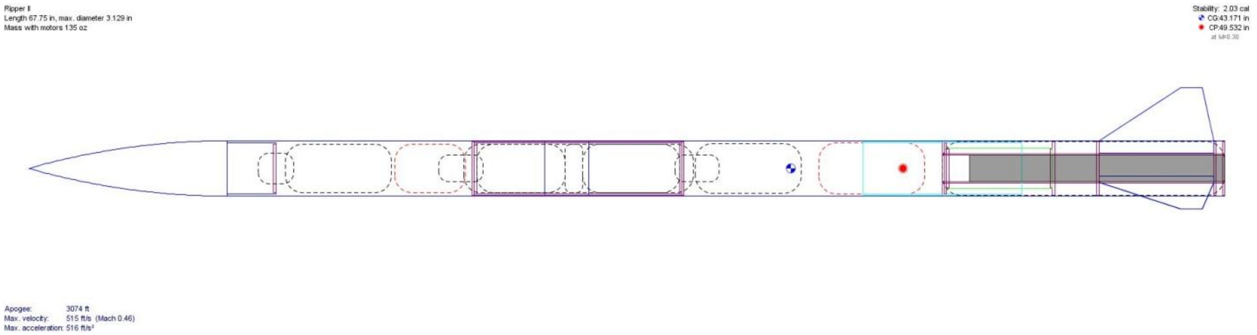
Ripper II
Length 67.75 in, max. diameter 3.129 in
Mass with motors 135 oz
Stability: 2.03 cal
CG:43.171 in
CP:49.532 in
at M=0.30
Apogee: 3074 ft
Max. velocity: 515 ft/s (Mach 0.46)
Max. acceleration: 516 ft/s²

Figure 10: OpenRocket Construction Analysis

CP$_{\textbf{Rocket}}$ =<u>49.5</u>” from top of rocket

CG$_{\textbf{Rocket}}$ = <u>43.2</u>” from top of rocket **with motor**

CG$_{\textbf{Rocket}}$ = <u>40.0</u>” from top of rocket **after burnout**

Stability Ratio = <u>2.03</u>

Analysis of the Anticipated Performance

Overview

Two methods of simulation were used in order to predict the performance of the rocket. These include the utilization of OpenRocket and MATLAB. OpenRocket was used for analysis earlier regarding the center of pressure and center of gravity as well as apogee predictions. OpenRocket can take into account wind forces, lift forces, drag forces, and much more. On the other hand, the MATLAB code written has more assumptions, but can be manipulated more. In MATLAB, there is more control over determining specific velocities at altitudes. Also, a simulation of the stopping distance after the drogue parachute is deployed is possible.

OpenRocket

Table 1 includes the predicted altitude and acceleration in ideal conditions.

Table 1: OpenRocket Anticipated Performance

Maximum Altitude	3074 ft
Maximum Acceleration	516 ft/s^2

The above anticipated performance was performed under ideal conditions, or no wind conditions. This will not be the case on the day of the launch, so analysis on wind effect on altitude was conducted. Anticipated performance of the rocket was done at wind speeds of 5 mph to 20 mph at 5 mph increments. Figure 11 illustrates the results of this analysis.

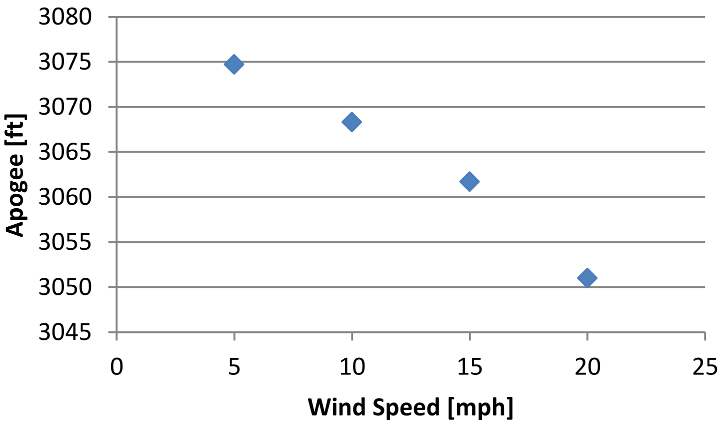

Figure 11: Effect of Wind Speed on Altitude

As illustrated above, even with extreme wind conditions of 20 mph, the predicted altitude of the rocket will still reach over 3000 feet.

MATLAB

The rocket flight consists of two segments:

i. During the 1st segment, the rocket is propelled upward by a variable thrust force opposing gravity and drag forces for a short time after which time the rocket fuel is exhausted and
ii. In the 2nd segment, the rocket continues to ascend with no thrust while slowing down due to gravity and drag until it reaches an apex in altitude.

With the experimental data of a thrust curve and known properties of a rocket, the rocket's flight can be modeled with aerodynamic drag while accounting for dependent thrust and variable change in mass of the rocket.

Assumptions

To model the rocket's flight to apogee taking into account every parameter would not be practical. Therefore, some assumptions were made. The following are the assumptions made when modeling the rocket flight.

i. Thrust Burn- Although thrust is not being assumed to be constant, it is being assumed that the thrust has a constant even burn.
ii. Gravity – Gravity is assumed to be a constant of 9.81 m/s^2. The change in gravity is so small, it will be considered negligible.
iii. Wind – It is assumed that there was minimal transverse wind, meaning that it will not be accounted for in the model.
iv. Vertical Flight – In order to simplify the model to fit our needs, it will be assumed that the flight of the rocket was strictly vertical.
v. Drag Force – When modeling this numerically, the following assumption will be made for the drag force.

Aerodynamic drag

$$F_D = \frac{1}{2}\rho C_D A V^2$$

where ρ is the density of air, C_D is the drag coefficient, A is the rocket cross-sectional area and V is the velocity of the rocket.

vi. Density-The density of air was assumed to be constant throughout the entire portion of the flight.

$$\rho_a = 0.00238 \text{ slug/ft}^3$$

Mathematical Model

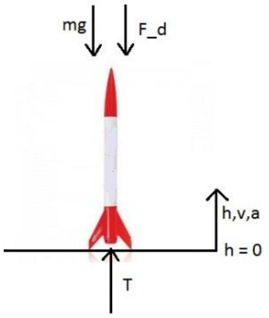

Figure 12: Free Body Diagram (thrust phase)

The conservation principle governing this free body diagram (thrust phase) is the following:

$$\sum F_{Vert} = \frac{d}{dt}(mv)$$

where the initial conditions are the following:

$$v(t=0) = 0\,ft/s$$

$$h(t=0) = 0\,ft$$

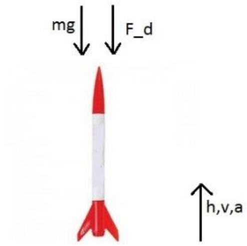

Figure 13: Free Body Diagram (after thrust phase)

The conservation principle governing this free body diagram (after thrust phase) is the following:

$$\sum F_{Vert} = \frac{d}{dt}(mv)$$

where the initial conditions are taken from the end of the thrust phase.

Numerical Model

Linear interpolation was used to extract data points of the thrust of the motor to take in account the variation. (http://www.thrustcurve.org/simfilesearch.jsp?id=1398)

The thrust curve determined by interpolating the given data was numerically integrated using the Trapezoidal Method as shown below.

$$\int T(t) = \left(\frac{T_i + T_{i+1}}{2}\right) dt$$

The percentage of the thrust burned off compared to the total impulse was calculated as shown below. (The integral of the thrust curve was cumulatively added while numerically integrating)

$$m_{fuel} = m_{fuel\_total} - m_{fuel\_total}\left(\frac{T_{cumulative}}{I}\right)$$

where I is the net impulse.

The mass during the thrust phase was calculated by the following expression.

$$m = m_{rocket} + m_{fuel}$$

After the thrust and mass were determined for each time step, fourth order Runge-Kutta numerical integration was used to calculate the velocity history. The acceleration expression used is from the conservation principle applied when determining the governing differential equation. The following describes the general form of fourth order Runge-Kutta numerical integration.

The four following slopes are determined first.

$$\varphi_1 = \varphi_i$$

$$\varphi_2 = \varphi(t_i + \frac{\Delta t}{2}, y_i + \varphi_1 \frac{\Delta t}{2})$$

$$\varphi_3 = \varphi(t_i + \frac{\Delta t}{2}, y_i + \varphi_2 \frac{\Delta t}{2})$$

$$\varphi_4 = \varphi(t_i + \Delta t, y_i + \varphi_3 \Delta t)$$

After the four slopes are calculated, the following relationship is used.

$$y_{i+1} = y_i + \frac{1}{6}(\varphi_1 + 2\varphi_2 + 2\varphi_3 + \varphi_4)\Delta t$$

The following algorithm was used to calculate the velocity history.

$$\varphi_1 = \frac{T_i}{m_i} - g - \frac{F_{D_i}}{m_i}$$

$$v^*{}_{i+1/2} = v_i + \varphi_1 \frac{\Delta t}{2}$$

$$\varphi_2 = \frac{T_{i+1/2}}{m_{i+1/2}} - g - \frac{F_{D_i+1/2}}{m_{i+1/2}}$$

$$v^{**}{}_{i+1/2} = v_i + \varphi_2 \frac{\Delta t}{2}$$

$$\varphi_3 = \frac{T_{i+1/2}}{m_{i+1/2}} - g - \frac{F_{D_i+1/2}}{m_{i+1/2}}$$

$$v^{***}{}_{i+1} = v_i + \varphi_3 \Delta t$$

$$\varphi_4 = \frac{T_{i+1}}{m_{i+1}} - g - \frac{F_{D_i+1}}{m_{i+1}}$$

$$v_{i+1} = v_i + \frac{1}{6}(\varphi_1 + 2\varphi_2 + 2\varphi_3 + \varphi_4)\Delta t$$

An Euler algorithm was used to solve for the altitude history.

The following is the basic form of the Euler algorithm.

$$x_{i+1} = x_i + \frac{dx}{dt}(i) * \Delta t$$

It is also known that:

$$h_{i+1} = h_i + v_i * \Delta t$$

The above algorithm is used during the thrust phase of the flight. Once the motor burns out, the algorithm remains the same except that the thrust term is taken out.

Predicted Velocity History

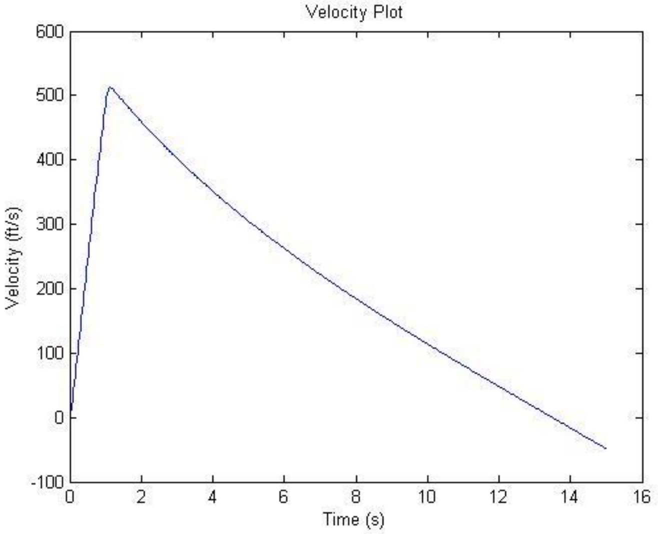

Figure 14: Velocity History Plot of the Rocket Produced by MATLAB

The maximum velocity during the flight is 513 ft/s. Note that the velocity after about 13 seconds (apogee) does not reflect the actual predicted velocity of the rocket.

Predicted Acceleration History

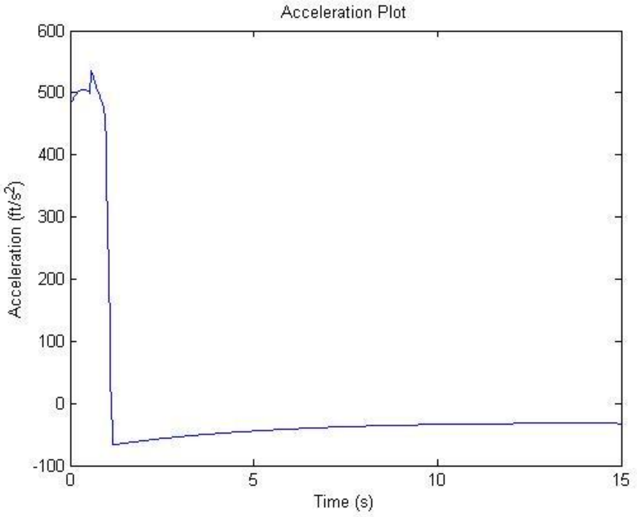

Figure 15: Acceleration History Plot of the Rocket Produced by MATLAB

It is anticipated that acceleration is initially positive since thrust is applied. The acceleration decreases during this phase because the drag force is increasing. Acceleration is negative after thrust ends because the rocket is slowing down. It should be noted that the maximum acceleration achieved in the flight is 537 ft/s^2.

Predicted Altitude History

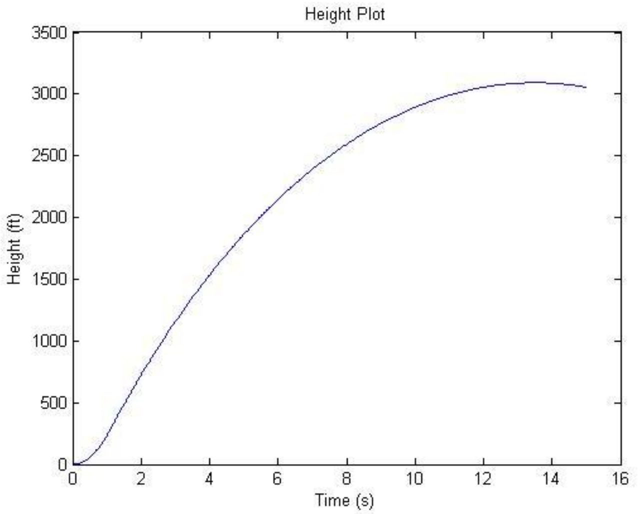

Figure 16: Anticipated Altitude Projection Plot of the Rocket Produced by MATLAB

The apogee predicted by this model occurs at 3092 feet. Note that the altitude of the rocket after about 13 seconds (apogee) does not reflect the actual predicted altitude of the rocket.

Summary of Flight Performance

Maximum Acceleration	**537 ft/s^2**
Maximum Altitude	**3092 $feet$**
Time to Apogee	**13 $seconds$**

Table 2: Pre-Flight Analysis Predictions of Results

Although the maximum altitude is predicted to be 3092 feet, the rocket will be stopped at 3000 feet due to a drogue chute. Therefore the time to apogee will also be less.

The analysis does not take into account launching a drogue parachute on the way up. Since there were test flights completed, more accurate data was used to predict the stopping distance. This analysis can be found in the testing section under stopping distance.

Testing

Ground Testing

A test was performed to size the ejection charges. The rocket was packed as it would when ready for launch, except the ejection charges were wired to an electronic trigger. Both the upper section and lower section were tested such that the size of the ejection charges would shear the shear pins and the parachutes would come out. This test was performed and the parachutes were deployed successfully on the ground.

Altimeter Testing

A test was performed in order to ensure the programming and function of the altimeter. This test was done on the MARSA4, the primary altimeter. The test was based on barometric pressure only. The altimeter was placed inside a plastic jar, where the wires to the e matches were wired through the jar and then epoxied to seal the hole created. A lid for the jar was constructed with fittings through the middle such that a tube can be attached to the outer fitting. A hand vacuum pump was then attached with a tube to this fitting. The pressure change caused by the vacuum pump will simulate a change in altitude by the barometric sensor reading the pressure. Through published tables, it was determined that a 3 or 4 in Hg change in pressure, would simulate the rocket reaching an altitude of 3000 feet (http://www.sablesys.com/baro-altitude.html). The jar is able to change the pressure by 5 or 6 in Hg. A picture of the setup is in Figure 17.

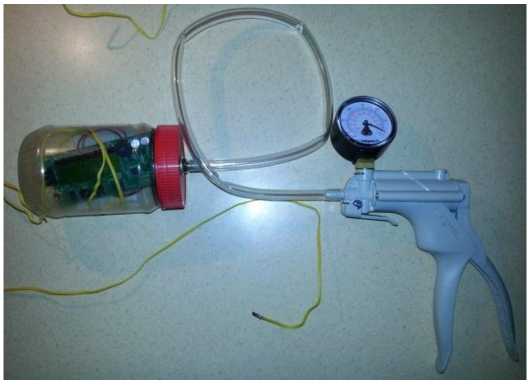

Figure 17: Altimeter Test Setup

To conduct the test, the altimeter was armed in the same way that it would be on the launch rail in the rocket. Then the pressure was changed via the hand vacuum pump to simulate a flight. The e matches fired when they were programmed to fire. Watching the test and looking at the data recorded on the computer confirmed that our altimeter was programmed correctly and functioning correctly.

Test Launches

Two test launches were performed in order to test the simulations and the overall function of all the components of the rocket. This will also allow for changes to be made before the competition launch day.

Test Launch #1

The first test launch had an error with the altimeter causing the parachutes to deploy immediately. After diagnosis of the altimeter, low voltage occurred on the altimeter causing it to fire the ejection charges. This occurred at maximum acceleration during the initial thrust phase. The rocket reached an altitude of around 700 feet and then descended down under the control of both parachutes. After retrieving the rocket safely, no damage occurred and the rocket returned in flyable condition. This verified the structural design of the vehicle. It confirmed that the additional support and choice of fiberglass prevented zippering. This then confirms that deploying a parachute on the way up will not cause zippering of the body tube.

Test Launch #2

The second test launch was a successful flight in regards to the competition parameters. The rocket achieved an apogee of 3272 feet and was recovered safely in flyable condition. This meets the window of apogee occurring between 2500 feet and 3500 feet. During the flight, the drogue parachute never deployed due to catching on the wing nut and screw of the rail button. Although this feature did not function properly, this allowed for recorded data of just the flight of the rocket. Without the drogue deploying, this altitude can now be compared to the simulated models. From this data, a more accurate coefficient of drag can be backed out. Even though there is still error in the simulated model, this will allow for more accurate calculations of stopping distances when the drogue is deployed correctly.

Stopping Distance Analysis

In order to model the stopping of the rocket once the drogue parachute is deployed, the following free body diagram was considered.

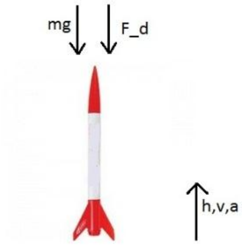

Figure 18: Free Body Diagram (Drogue Deployed)

The same forces are applicable as during the coasting phase of the rocket except that the coefficient of drag now includes the parachute. The parachute has a large cross sectional area and a larger drag coefficient, increasing the drag forces significantly. A coefficient of drag of 1.2 was used. The cross sectional are was calculated based on the size of the parachute. This was done for several different sizes of parachutes. Based on this data, the 24 inch drogue parachute has the capability to stop the rocket by 3000 feet when firing the parachute at 2900 feet. The parachute size could also be altered by tightening the shroud lines. A tenth of a second delay was added for the delay to fire the ejection charge based on the altimeter. Figure 19 shows an altitude history plot with the inclusion of the deployment of the 24 inch drogue parachute.

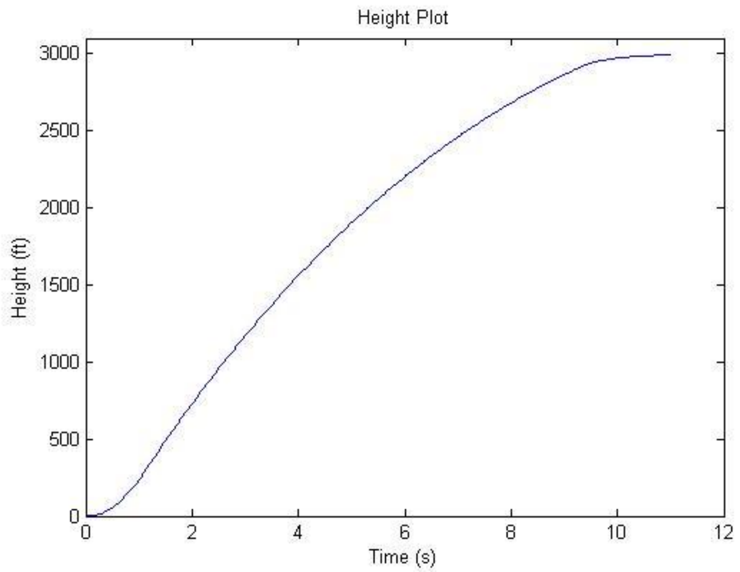

Figure 19: Altitude History with Drogue Deployment

The final altitude with the drogue deployment in the simulation is 2990 feet.

Conclusion

“The Ripper II” was designed with minimal additional features that can cause problems. Due to this simplicity, the main control of altitude is by passively controlling drag via the design of the airframe and fins of the rocket. The altitude is also controlled by actively controlling the drag. This is done by deploying a parachute on the ascent of the flight to stop the rocket at 3000 feet. Analysis simulating the flight of the rocket was conducted as well as the simulation of the stopping distance of the rocket after the drogue parachute is deployed. The simulated apogee of the rocket occurs at 2990 feet. This satisfies the parameters of the competition and is very close to the objective. The recovery system that includes dual deployment with redundancy along with a backup motor ejection charge will ensure the rocket returns safely in flyable condition. Along with testing in the form of stationary tests and test launches, the design of the rocket will lead to a successful flight with an apogee nearing 3000 feet.

Appendix

Budget

An itemized budget is shown in the table below.

Team Jarts Budget

ITEM	DESCRIPTION	MANUFACTURER	COST PER UNIT	# OF UNITS	TOTAL COST
1	5' 3" DIA FIBERGLASS TUBE	HAWK MOUNTAIN	$21.53	5	$107.65
2	TUBE CUTTING	HAWK MOUNTAIN	$2.75	7	$19.25
3	TUBE SLOTS	HAWK MOUNTAIN	$6.00	3	$18.00
4	3" COUPLER	HAWK MOUNTAIN	$22.19	2	$44.38
5	1.5" MOTOR MOUNT TUBE	HAWK MOUNTAIN	$10.51	2	$21.02
6	2.175" TUBE AXIAL SUPPORT	HAWK MOUNTAIN	$23.49	1	$23.49
7	3" TO 1.5" CENTERING RINGS	HAWK MOUNTAIN	$6.00	5	$30.00
8	2.17" TO 1.5" CENTERING RINGS	HAWK MOUNTAIN	$5.00	2	$10.00
9	3" BULKPLATE	HAWK MOUNTAIN	$5.00	2	$10.00
10	3" COUPLER BULKPLATE	HAWK MOUNTAIN	$5.50	2	$11.00
11	AERO PACK 38MM MOTOR RETAINER	HAWK MOUNTAIN	$29.00	1	$29.00
12	FINS	PUBLIC MISSILES	$6.69	3	$20.07
13	ADDITIONAL MOTORS	WILDMAN ROCKETRY	$55.95	3	$167.85
14	FLIGHT CARDS	NAR	$10.00	3	$30.00
15	MISCELLANEOUS HARDWARE	ACE HARDWARE	$75.00	1	$75.00
16	3" NOSE CONE	APOGEE ROCKETS	$18.85	1	$18.85
17	EXPOXY	WILDMAN ROCKETRY	$56.58	1	$56.58
18	KEY SWITCHES	JAMECO	$5.49	5	$27.45
19	30" SHOCK CORD PROTECTOR	WHATS UP HOBBY	NA	2	NA
20	12"X12" PARACHUTE BLAST PROTECTOR	WHATS UP HOBBY	NA	1	NA
21	6"X6" PARACHUTE BLAST PROTECTOR	WHATS UP HOBBY	NA	1	NA
22	0.5" TUBULAR NYLON SHOCK CORD	PUBLIC MISSILES	NA	20	NA
23	PERFECT FLITE STRATTOLOGGER	WHATS UP HOBBY	NA	1	NA
24	MARSA 54	WHATS UP HOBBY	NA	1	NA
25	9V BATTERY CLIPS	PARTS EXPRESS	$2.95	5	$14.75
26	MAIN PARACHUTE	PUBLIC MISSILES	NA	1	NA
27	BREAK PARACHUTE	TBD	$75.00	1	$75.00
28	TRACKING SYSTEM	WILDMAN ROCKETRY	NA	1	NA
29	ROCKET STAND(PVC)	MENARDS	$12.79	1	$12.79
30	TEST LAUNCH STATE PARK ADMISSION	STATE PARK	$10.00	1	$10.00
31	USB DATA TRANSFER KIT	PERFECT FLITE	$30.00	1	$30.00
32	DREMEL ACCESSORY KIT	TBD	$30.00	1	$30.00
33	OVERHEAD	NA	$50.00	1	$50.00
				TOTAL	$942.13

MATLAB CODE

```
% Team Jarts 2012-2013 Motor - I540
% Eric Logisz
clear
clc
% Define Constants
g = 32.174; % gravitational acceleration in ft/s^2
p = .00238; % air density in slug/ft^3
C_d = 0.41; % drag coefficient
D = 3.129/12; % fuselage diameter in ft
A = pi*(D/2)^2; % rocket cross sectional area in ft^2
burntime = 1.18; % duration of thrust from rocket engine in seconds
m_rocketwcasing = 7.72; % weight of rocket and casing in lb
m_prop_weight = 0.72; % weight of fuel in lb
m_rocket = m_rocketwcasing/g; % Mass of the rocket and motor casing in slugs
m_fuel_tot = m_prop_weight/g; % Total mass of the fuel in slugs
% Define time step and bondaries (s)
dt = 0.025;
t_min = 0;
t_max = 11;
t = t_min:dt:t_max;
% Define time with half the time step
t2 = t_min:(dt/2):t_max;
% Read time and thrust data from Excel spreadsheet
t_read=rot90(xlsread('thrust curve i540.xls', 1, 'A1:A19'));
T_read=rot90(xlsread('thrust curve i540.xls', 1, 'B1:B19'));
% Define Thrust
T(1) = 0;
% Interpolate Thrust Data with Linear Interpolation
for a = 2:1:length(t)
    if t_read(1) <= t(a) && t(a) <= t_read(2)
        T(a) = ((T_read(2)-T_read(1))/(t_read(2)-t_read(1)))*(t(a)-t_read(1))
+ T_read(1);
    elseif t_read(2) <= t(a) && t(a) <= t_read(3)
        T(a) = ((T_read(3)-T_read(2))/(t_read(3)-t_read(2)))*(t(a)-t_read(2))
+ T_read(2);
    elseif t_read(3) <= t(a) && t(a) <= t_read(4)
        T(a) = ((T_read(4)-T_read(3))/(t_read(4)-t_read(3)))*(t(a)-t_read(3))
+ T_read(3);
    elseif t_read(4) <= t(a) && t(a) <= t_read(5)
        T(a) = ((T_read(5)-T_read(4))/(t_read(5)-t_read(4)))*(t(a)-t_read(4))
+ T_read(4);
    elseif t_read(5) <= t(a) && t(a) <= t_read(6)
        T(a) = ((T_read(6)-T_read(5))/(t_read(6)-t_read(5)))*(t(a)-t_read(5))
+ T_read(5);
    elseif t_read(6) <= t(a) && t(a) <= t_read(7)
        T(a) = ((T_read(7)-T_read(6))/(t_read(7)-t_read(6)))*(t(a)-t_read(6))
+ T_read(6);
    elseif t_read(7) <= t(a) && t(a) <= t_read(8)
        T(a) = ((T_read(8)-T_read(7))/(t_read(8)-t_read(7)))*(t(a)-t_read(7))
+ T_read(7);
    elseif t_read(8) <= t(a) && t(a) <= t_read(9)
        T(a) = ((T_read(9)-T_read(8))/(t_read(9)-t_read(8)))*(t(a)-t_read(8))
+ T_read(8);
    elseif t_read(9) <= t(a) && t(a) <= t_read(10)
```

```
        T(a) = ((T_read(10)-T_read(9))/(t_read(10)-t_read(9)))*(t(a)-
t_read(9)) + T_read(9);
    elseif t_read(10) <= t(a) && t(a) <= t_read(11)
        T(a) = ((T_read(11)-T_read(10))/(t_read(11)-t_read(10)))*(t(a)-
t_read(10)) + T_read(10);
    elseif t_read(11) <= t(a) && t(a) <= t_read(12)
        T(a) = ((T_read(12)-T_read(11))/(t_read(12)-t_read(11)))*(t(a)-
t_read(11)) + T_read(11);
    elseif t_read(12) <= t(a) && t(a) <= t_read(13)
        T(a) = ((T_read(13)-T_read(12))/(t_read(13)-t_read(12)))*(t(a)-
t_read(12)) + T_read(12);
    elseif t_read(13) <= t(a) && t(a) <= t_read(14)
        T(a) = ((T_read(14)-T_read(13))/(t_read(14)-t_read(13)))*(t(a)-
t_read(13)) + T_read(13);
    elseif t_read(14) <= t(a) && t(a) <= t_read(15)
        T(a) = ((T_read(15)-T_read(14))/(t_read(15)-t_read(14)))*(t(a)-
t_read(14)) + T_read(14);
    elseif t_read(15) <= t(a) && t(a) <= t_read(16)
        T(a) = ((T_read(16)-T_read(15))/(t_read(16)-t_read(15)))*(t(a)-
t_read(15)) + T_read(15);
    elseif t_read(16) <= t(a) && t(a) <= t_read(17)
        T(a) = ((T_read(17)-T_read(16))/(t_read(17)-t_read(16)))*(t(a)-
t_read(16)) + T_read(16);
    elseif t_read(17) <= t(a) && t(a) <= t_read(18)
        T(a) = ((T_read(18)-T_read(17))/(t_read(18)-t_read(17)))*(t(a)-
t_read(17)) + T_read(17);
    elseif t_read(18) <= t(a) && t(a) <= t_read(19)
        T(a) = ((T_read(19)-T_read(18))/(t_read(19)-t_read(18)))*(t(a)-
t_read(18)) + T_read(18);
    elseif t_read(19) <= t(a)
        break
    end
end
% Define Thrust at every half time step
T2(1) = 0;
% Interpolate Thrust Data with Linear Interpolation
for b = 2:1:length(t2)
    if t_read(1) <= t2(b) && t2(b) <= t_read(2)
        T2(b) = ((T_read(2)-T_read(1))/(t_read(2)-t_read(1)))*(t2(b)-
t_read(1)) + T_read(1);
    elseif t_read(2) <= t2(b) && t2(b) <= t_read(3)
        T2(b) = ((T_read(3)-T_read(2))/(t_read(3)-t_read(2)))*(t2(b)-
t_read(2)) + T_read(2);
    elseif t_read(3) <= t2(b) && t2(b) <= t_read(4)
        T2(b) = ((T_read(4)-T_read(3))/(t_read(4)-t_read(3)))*(t2(b)-
t_read(3)) + T_read(3);
    elseif t_read(4) <= t2(b) && t2(b) <= t_read(5)
        T2(b) = ((T_read(5)-T_read(4))/(t_read(5)-t_read(4)))*(t2(b)-
t_read(4)) + T_read(4);
    elseif t_read(5) <= t2(b) && t2(b) <= t_read(6)
        T2(b) = ((T_read(6)-T_read(5))/(t_read(6)-t_read(5)))*(t2(b)-
t_read(5)) + T_read(5);
    elseif t_read(6) <= t2(b) && t2(b) <= t_read(7)
        T2(b) = ((T_read(7)-T_read(6))/(t_read(7)-t_read(6)))*(t2(b)-
t_read(6)) + T_read(6);
    elseif t_read(7) <= t2(b) && t2(b) <= t_read(8)
```

```
        T2(b) = ((T_read(8)-T_read(7))/(t_read(8)-t_read(7)))*(t2(b)-
t_read(7)) + T_read(7);
    elseif t_read(8) <= t2(b) && t2(b) <= t_read(9)
        T2(b) = ((T_read(9)-T_read(8))/(t_read(9)-t_read(8)))*(t2(b)-
t_read(8)) + T_read(8);
    elseif t_read(9) <= t2(b) && t2(b) <= t_read(10)
        T2(b) = ((T_read(10)-T_read(9))/(t_read(10)-t_read(9)))*(t2(b)-
t_read(9)) + T_read(9);
    elseif t_read(10) <= t2(b) && t2(b) <= t_read(11)
        T2(b) = ((T_read(11)-T_read(10))/(t_read(11)-t_read(10)))*(t2(b)-
t_read(10)) + T_read(10);
    elseif t_read(11) <= t2(b) && t2(b) <= t_read(12)
        T2(b) = ((T_read(12)-T_read(11))/(t_read(12)-t_read(11)))*(t2(b)-
t_read(11)) + T_read(11);
    elseif t_read(12) <= t2(b) && t2(b) <= t_read(13)
        T2(b) = ((T_read(13)-T_read(12))/(t_read(13)-t_read(12)))*(t2(b)-
t_read(12)) + T_read(12);
    elseif t_read(13) <= t2(b) && t2(b) <= t_read(14)
        T2(b) = ((T_read(14)-T_read(13))/(t_read(14)-t_read(13)))*(t2(b)-
t_read(13)) + T_read(13);
    elseif t_read(14) <= t2(b) && t2(b) <= t_read(15)
        T2(b) = ((T_read(15)-T_read(14))/(t_read(15)-t_read(14)))*(t2(b)-
t_read(14)) + T_read(14);
    elseif t_read(15) <= t2(b) && t2(b) <= t_read(16)
        T2(b) = ((T_read(16)-T_read(15))/(t_read(16)-t_read(15)))*(t2(b)-
t_read(15)) + T_read(15);
    elseif t_read(16) <= t2(b) && t2(b) <= t_read(17)
        T2(b) = ((T_read(17)-T_read(16))/(t_read(17)-t_read(16)))*(t2(b)-
t_read(16)) + T_read(16);
    elseif t_read(17) <= t2(b) && t2(b) <= t_read(18)
        T2(b) = ((T_read(18)-T_read(17))/(t_read(18)-t_read(17)))*(t2(b)-
t_read(17)) + T_read(17);
    elseif t_read(18) <= t2(b) && t2(b) <= t_read(19)
        T2(b) = ((T_read(19)-T_read(18))/(t_read(19)-t_read(18)))*(t2(b)-
t_read(18)) + T_read(18);
    elseif t_read(19) <= t2(b)
        break
    end
end

% Define initial conditions for Mass variables
m_p(1) = 0;
m_p_tot(1) = 0;
% Numerically integrate Thrust to determine the percent of fuel burned
% Note: finding a value at each half time step
for j = 1:1:length(T)-1
    m_p(j+1) = m_p(j) + ((dt/2)/2)*(T(j) + T(j+1));
    m_p_tot(j+1) = m_p_tot(j) + m_p(j+1);
end
% Define total area under curve
m_p_tot_tot = m_p_tot(length(T)-1);
% Determine the mass of the rocket at each time step
% Note: finding a value at each half time step
for k = 1:1:length(m_p_tot)-1
    m_fuel(k) = m_fuel_tot - m_fuel_tot*(m_p_tot(k)/m_p_tot_tot);
    m(k) = m_rocket + m_fuel(k);
end
```

```
for k = length(m_p_tot)-1:1:length(t2)
    m(k) = m_rocket;
end
% Define initial conditions for Mass variables
m_p2(1) = 0;
m_p_tot2(1) = 0;
% Numerically integrate Thrust to determine the percent of fuel burned
% Note: finding a value at each half time step
for j = 1:1:length(T2)-1
    m_p2(j+1) = m_p2(j) + ((dt/2)/2)*(T2(j) + T2(j+1));
    m_p_tot2(j+1) = m_p_tot2(j) + m_p2(j+1);
end
% Define total area under curve
m_p_tot_tot2 = m_p_tot2(length(T2)-1);
% Determine the mass of the rocket at each time step
% Note: finding a value at each half time step
for k = 1:1:length(m_p_tot2)-1
    m_fuel2(k) = m_fuel_tot - m_fuel_tot*(m_p_tot2(k)/m_p_tot_tot2);
    m2(k) = m_rocket + m_fuel2(k);
end
for k = length(m_p_tot)-1:1:length(t2)
    m2(k) = m_rocket;
end
% Define Initial Conditions of Velocity and Height
v_a(1) = 0;
h_a(1) = 0;
n(1) = 0;
% Compute Velocity and Height Data
for i = 1:1:length(t)-1
    if  t(i+1) <= burntime
        if T(i)/m(i) <= 32.174
            g = 0;
        else
            g = 32.174;
        end
        a1(i) = ((T(i))/(m(i))) - g - (((1/2)*p*C_d*A*(v_a(i))^2)/(m(i)));
        v_f1(i) = v_a(i) + a1(i)*(dt/2);
        a2(i) = ((T2(i+1+n(i)))/(m2(i+1+n(i)))) - g - 
(((1/2)*p*C_d*A*(v_f1(i))^2)/(m2(i+1+n(i))));
        v_f2(i) = v_a(i) + a2(i)*(dt/2);
        a3(i) = ((T2(i+1+n(i)))/(m2(i+1+n(i)))) - g - 
(((1/2)*p*C_d*A*(v_f2(i))^2)/(m2(i+1+n(i))));
        v_f3(i) = v_a(i) + a3(i)*(dt);
        a4(i) = ((T(i+1))/(m(i+1))) - g - 
(((1/2)*p*C_d*A*(v_f3(i))^2)/(m(i+1)));
        a(i) = (1/6)*(a1(i)+2*a2(i)+2*a3(i)+a4(i));
        v_a(i+1) = v_a(i) + a(i)*dt;
    else
        g = 32.174;
        a1(i) = - g - (((1/2)*p*C_d*A*(v_a(i))^2)/(m(i)));
        v_f1(i) = v_a(i) + a1(i)*(dt/2);
        a2(i) = - g - (((1/2)*p*C_d*A*(v_f1(i))^2)/(m2(i+1+n(i))));
        v_f2(i) = v_a(i) + a2(i)*(dt/2);
        a3(i) = - g - (((1/2)*p*C_d*A*(v_f2(i))^2)/(m2(i+1+n(i))));
        v_f3(i) = v_a(i) + a3(i)*(dt);
        a4(i) = - g - (((1/2)*p*C_d*A*(v_f3(i))^2)/(m(i+1)));
        aa(i) = (1/6)*(a1(i)+2*a2(i)+2*a3(i)+a4(i));
        v_a(i+1) = v_a(i) + aa(i)*dt;
    end
    h_a(i+1) = h_a(i) + v_a(i)*dt;
    n(i+1) = n(i) + 1;
    if h_a(i+1) >= (2900 + 0.1*v_a(i))
       C_d = 1.2;
       A = pi*((24/12)/2)^2;
       v_a_stored(i) = v_a(i);
    end
end
% Make length(t) equal to v,h vectors for plotting
t_new_a(1) = 0;
for p = 1:1:length(v_a)-1
    t_new_a(p+1) = t_new_a(p) + dt;
end
% Plot altitude graph
plot(t_new_a,h_a)
xlabel('Time (s)'); ylabel('Height (ft)');
title('Height Plot');
max_height = max(h_a);
disp(['Max Height = ', num2str(max_height),' ft'])
```