Rocket Guidance and Control

an Olin Self Study

by Andrew Heine and Luis Rayas

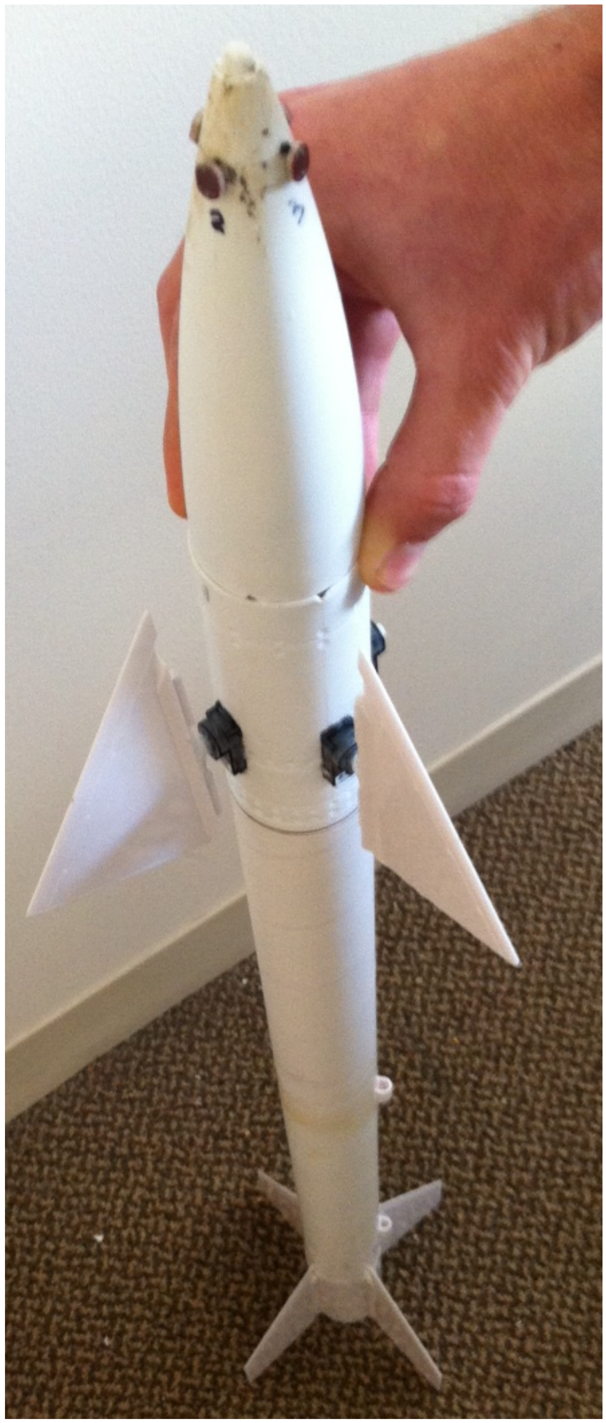

Table of Contents

Abstract

This paper describes the research, design, construction, and testing of a low-budget guided rocket, whose goal is to fly towards the sun. The system is defined by requirements, which are determined from a use model of how the rocket should operate, and under what conditions. The rocket is divided into a frame, provided by a hobby rocket kit, and a “Sense, Think, Act” architecture, consisting of a photoresistive array for sensing, an Arduino Pro Micro for processing, and 4 servo-mounted nose fins for steering the rocket. The “Sense, Think, Act” paradigm is modeled after the Olin robotics architecture, with a focus on compensating for stability and control. A full simulation was made of the system, a linearized control system was analyzed, and subsystem tests were conducted, including torque measurements in a wind tunnel. The rocket was tested in flight, but did not get to a final sun-tracking test, due to malfunctioning after crash-landings. However, the initial test suggested the potential to control the rocket with finer tuning of control parameters.

Introduction

This semester, we investigated the physics, controls, and construction of a flight-path controlled rocket. We chose to target the sun, as it was a safe target that was easy to spot. This field has had a great deal of research, owing largely to military technology. However, our interest was more inspired by the growing private involvement in space exploration, and the ability to do so with lower costs. Expanding on projects ranging from sidewinder missiles, which have held government contracts since the 1950’s, and DIY projects in modified, steerable hobby rockets, we sought to bring the best of both worlds to the table, using budget, quality sensing and actuating, with a robust and informed control system. We believe that this project provides a good way to understand rocket dynamics, and demonstrates that complex controls can be successful for reasonable costs.

System Definition

Use Model

The goal of the system is to launch and steer a projectile towards the sun for a budget of under $1,200. It will be designed and tested by two college students in the Spring of 2013 for their Self Study in Controls and Rocketry. The entire use of the rocket is for testing and demonstration. As a research project, robustness and character are less important than function and usability.

The users of the rocket are its designers. They will have full knowledge of how to operate it, and have studied the physics carefully.

The system will be used in a testing environment- an approximately 30 acre patch of farmland in Sudbury, MA. During testing, only the users will be near the rocket, and they will be behind a plywood shield, with slits for viewing and filming the rocket. No one else will be within eyesight of the launch site. There will be patches of trees up to 50 ft high in the vicinity.

Testing will only take place when wind is below 20 miles per hour. To aim the rocket vertically toward the sun, testing will need to take place between the hours of 10am and 3pm. Testing will take place in spring/summer. There will be a total of two trips to the test site, allowing for 5 hours of testing for design purposes, and 5 hours of testing to demonstrate the rocket's ability. During this time, the users will be taking video of the rocket, taking data about the performance, and potentially modifying the rocket, both mechanically and in software.

The rocket will be transported to the launch site in the trunk of a sedan with two people available for packing and unpacking the rocket.

Steering a rocket towards the sun is considered within the safety and legal guidelines set by the National Association of Rocketry. However, law enforcement in the area may not be familiar with these guidelines, and may be alarmed if the rocket looks dangerous.

Requirements/Parameters

To realize the scenario described above, we define a set of requirements that our final system must meet. Each of these parameters will be met by functional subsystems, which are outlined below.

Achieving its Goal

- Our system must be able to meet its primary goal of steering towards the sun. To do this, it must be able to:
    - Rotate itself by 60 degrees during 1.7 second powered-flight. The maximum starting angle of the rocket will be 60 degrees from the angle of the sun.
    - Must be able to sense the angle of the sun when the sun is within 60 degrees of the rocket angle.
    - Must be stable
- Our system must cost less than $1,200
- We plan to compensate our system using control theory. The compensation will be modeled using MATLAB. While many of these metrics will not be precisely measured during flight tests, we have created metrics to be met by our MATLAB model that, if

met, should improve our rocket's performance.

  - Steady-State Error to a Step must equal *0%.* Our system only operates in powered-flight for 1.7 seconds, however, theoretically, it should be able to target the exact location of the sun with enough time.
  - Disturbance rejection should be at least 90%. This will help us reduce the effect of wind gusts.
  - Noise Rejection should be at least 90% above 100rps.
  - Settling Time to within 2% accuracy must be less than 1.7 seconds, since this is the burn time of Estes D-12 engines.
  - Peak Overshoot is not applicable to our system. So long as the system is stable and the settling time is within tolerance, the sampling rate is
  - Sampling Rate: Our sensor should measure the angle relative to the sun at at least twice the system's Nyquist frequency, to avoid aliasing. The bandwidth (-3dB gain) of our system is 1 radian per second, or 1/2π Hz, which is easily surpassed by many data acquisition units.

Usability

- Can be relaunched several times in a few hours
  - Fins, parachute, motor, and sensor are replaceable in 30 minutes
  - Can be retrieved if it lands in a tree
  - All electronics are accessible for replacement within 30 minutes
  - Nothing other than replaceable components breaks after 10 hours of testing
- Can be transported to the launch site
  - Must fit in the backseat of a sedan
  - Must be transportable by two people on foot
- Rocket is modifiable in the field
  - Fins, parachute, motor, and sensor are replaceable in 30 minutes
  - Rocket control parameters can be accessed within 5 minutes

Safety

- Never steers below a 60 degree angle while powered
  - Is stable
  - Control algorithm avoids shallow angles
- Doesn't go far enough to leave our line of sight
  - Motors push no farther than 1k ft

Functional Subsystems

The functional subsystems have been chosen and/or designed to meet the parameters described above. More details on the designs are in the section "Subsystem Design."

Sense - We will use an array of four infrared LEDs to detect the sun's angle.

Act - We will servo-mounted fins on the nose of the rocket to steer the rocket.

Think - We will use an Arduino Pro Micro to determine the angle of the rocket's motion, using the sensor input. This will also actuate the control fins.

Electronics Power Supply - E-flite 250 mAh 3.7 volt lithium polymer battery.

Recovery - Comes with Rocket

Frame - Comes with Rocket

Thrust - Comes with Rocket

Launch System - Comes with Rocket

Estes Tomahawk was chosen, as its frame and thruster meet the size and range requirements.

Interfaces

The Sense, Think, and Act components are wired together and housed in the nose of the rocket. The fins must be on the outside of the rocket, with their actuator on the inside.

Subsystem Design

Rocket Platform and Modifications

We were able to meet many of our systems requirements with the commercially available Estes D-Region Tomahawk hobby rocket. Several hobby rockets exist that include suitable subsystems for frame, propulsion, launch system, and recovery. Since our requirements for the launch system and recovery system are "it has them," we focused on propulsion and the frame. With some small mechanical modifications, we were able to add our other subsystems without harming the core functionality of the rocket.

The propulsion system for the Tomahawk strikes a good balance between having enough flight time to be controlled, but not flying to unsafe distances. The maximum altitude that it attains with a D-12 motor is 750 ft. This motor also provides 1.7 seconds of burn time, which sets the turning speed requirement for our control system. As mentioned above, the requirement is to be able to rotate by 60 degrees and settle within 1.7 seconds.

The frame of our system must be able to protect our added subsystems, namely, the sensor, actuator, and computer modifications. Taking a hint from KoD and Navic [1], we used the Tomahawk for its hollow nose section, which can be cut open and repurposed as a payload bay to contain electronics. This isolates all of the electronics from the heat and debris of the engines, allows the control system to be near the actuated fins on the nose, and keeps the center of mass close to the nose for increased stability. Furthermore, keeping all of our modifications in the nose prevents interference with the parachute recovery system of the rocket.

To grant access to the interior of the payload bay, we cut horizontally across the nose. This allowed us to place our modifications anywhere in the payload bay. We used support strips cut from another piece of the nose to keep the detachable portion in position, and duct tape to secure it.

Once we determined that this hobby rocket would fit our needs, we conducted test flights of the unmodified rocket to learn about the intricacies of its operation, what the important parts of the construction are, and to gain an intuition for how our models should behave. We gained a few key takeaways from these launches. We learned which parts of the rocket were fragile, and what construction techniques were effective under flight loads, so that we could avoid mistakes on our final rocket. We learned that the rocket was incredibly stable. Even with imperfections in the body (we had one fin that liked to wiggle and a few pieces of duct tape on the exterior of the rocket), the rocket launched nearly straight up and did not oscillate or spin. This information would also come in handy when we took torque data in a wind tunnel, because we could compare the amount of torque that kept that rocket stable to the amount of torque that we would add to the rocket. Finally, we were able to observe the qualitative behavior of the rocket, such as the speed of it’s rotation during descension, to compare to our model.

Sensor

Although most commercial heat seeking sensors use a scanning system where a single, albeit potentially complex, infrared sensor is mounted on a rotating and tilting device or gimbal (Figure 1 shows one such sensor), our system uses a number of stationary photoresistors whose relative resistances are used to infer the angle of the greatest light source.

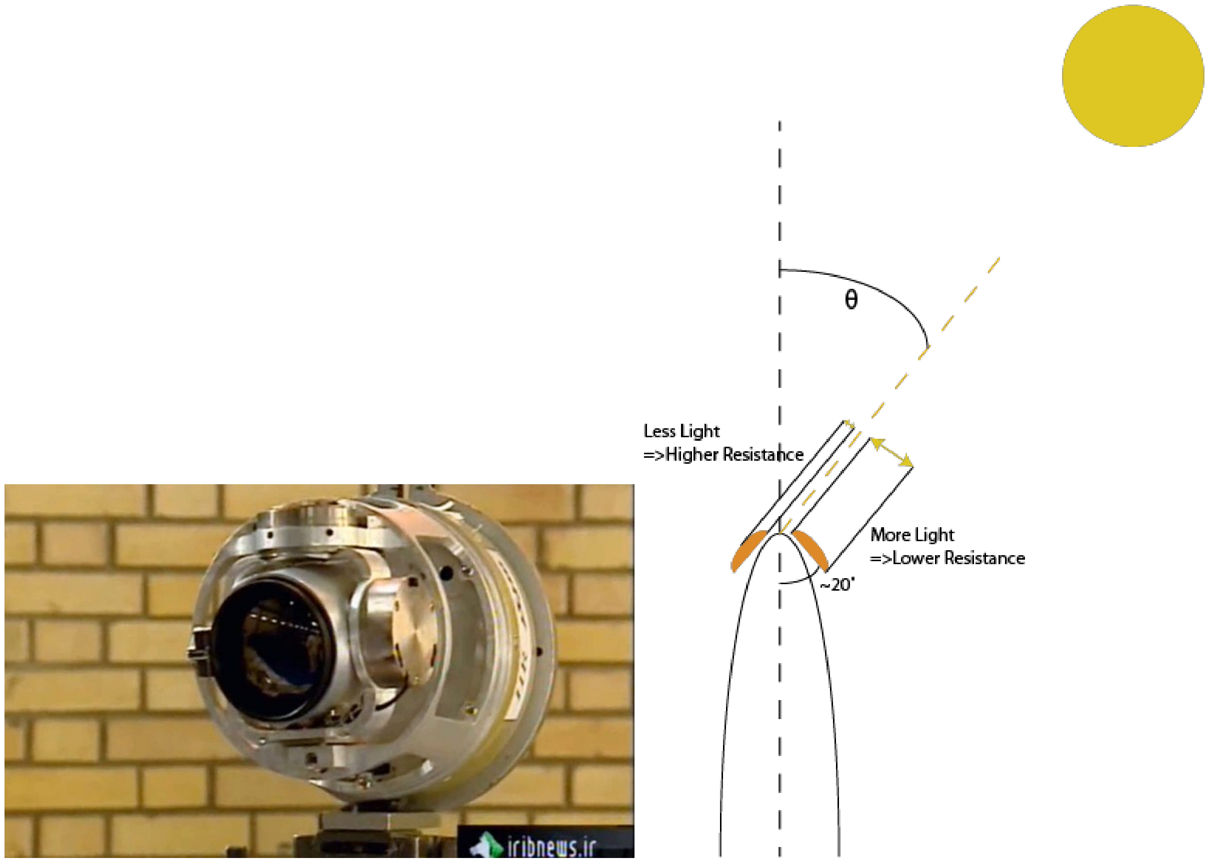

Figure 1. Light sensor. (Left) A commercial gimbal-mounted sun sensor. (Right) The theoretical basis for our photoresistive array sun sensor. More sunlight will be absorbed by photoresistors with a direct line of sight. Comparing the resistors' values will allow us to estimate the angle between the rocket and the sun.

We developed a "light" sensor that utilizes four photoresistors arranged in an array to give us data about the direction of the greatest source of light. By angling each photoresistor in a different direction, the angle of attack of the light results in a disparity between photoresistors which can be used to calculate the approximate angle of the net source of light. A single opposing pair of photoresistors illustrate this concept in Figure 1.

Although our sensor is not particularly meant for the infrared spectrum, it is adequate for our more specific goal of targeting the sun. We developed a simple platform to test how a photoresistor acts at different positions in the sun. This testing rig scanned across a range of 90 degrees at 5 degree increments and recorded the calculated resistance of the photoresistor. The results of our scans are plotted in Figure 2.

What we most needed to validate in our testing was whether the resolution of the array would be sufficient. What this means practically is that the difference in resistance between two angles separated by the maximum acceptable angular error should not be so small that it could be attributed to noise. We validate that this is the case by careful inspection of the graph (Figure 2) of resistances. The noise to signal ratio is surprisingly low. With the exception of sections in which a cloud obscured the sun, the resistance monotonically decreases as the photoresistor approaches directly facing the sun. Thus we conclude that our sensor's resolution is sufficient to the accuracy that we are able to measure the mounting angle of the photoresistors (i.e. the mounting error is more than the error of the sensors).

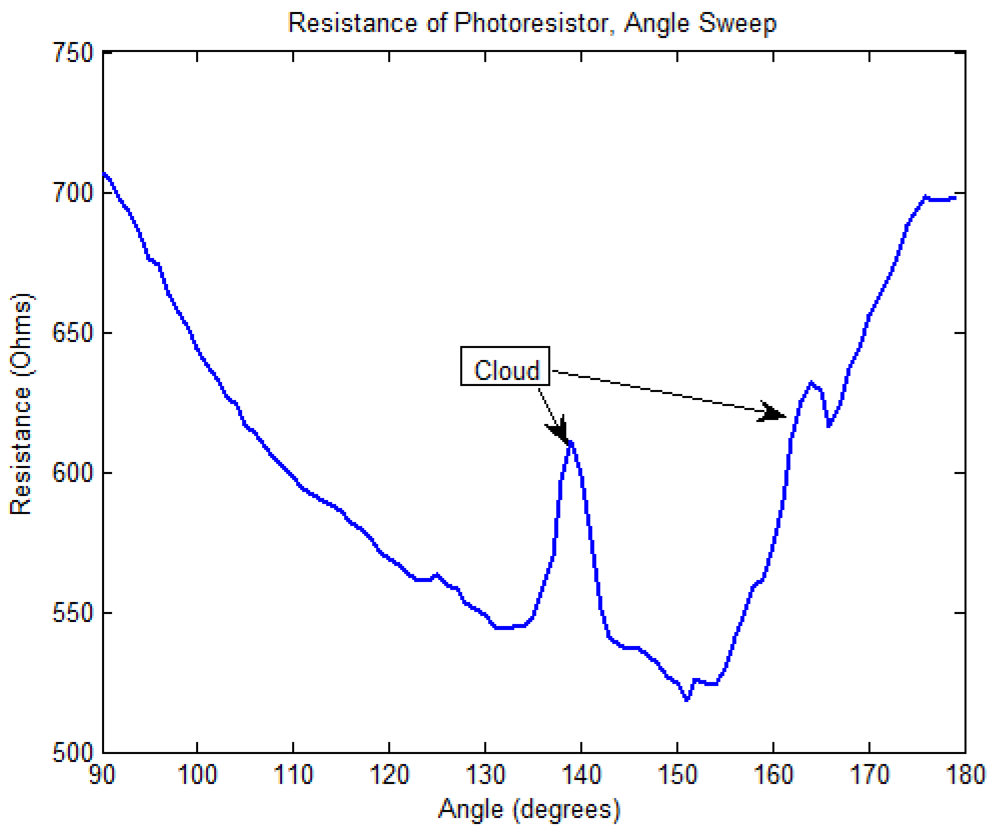

Figure 2. Photoresistor Response over 90° sweep. The sun was approximately 55° from vertical. As the photoresistor approaches a direct line of sight to the sun, the resistance drops.

A voltage divider was created for each photoresistor, so that the varying resistances could be measured by an Arduino Pro Micro via the analog input ports. The resistance of the complementing resistors was chosen to be 240 Ohms which is the same order of magnitude as the resistances we expect to measure in the field. This gives us minimal loss when converting to a voltage through the voltage divider equation ($V_{measured} = V_{cc}R_c/(R_m + R_c)$) since we take better advantage of the analog input's range.

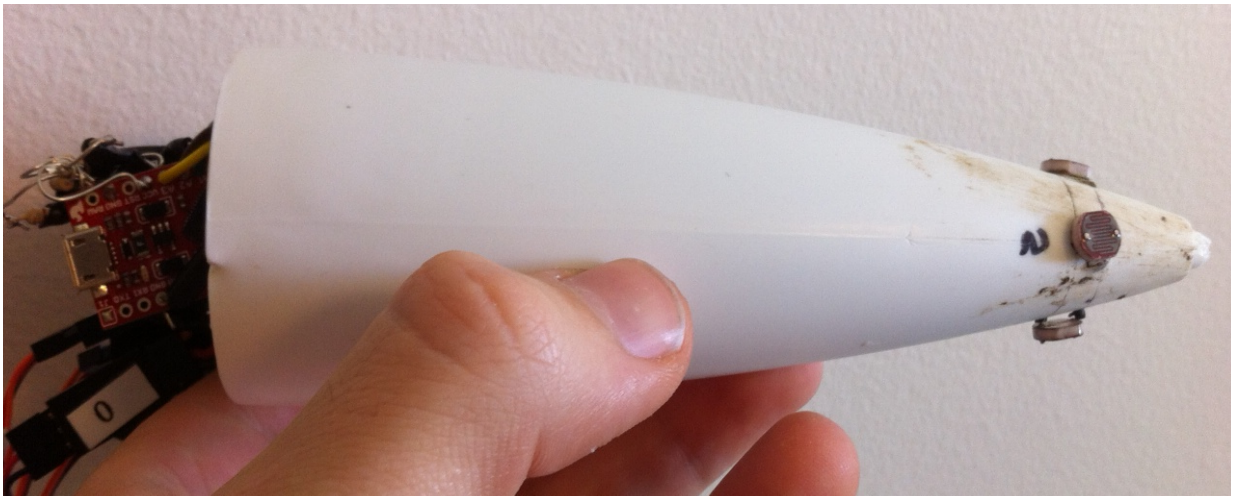

Figure 3. Sensor Tip. Four photoresistors are attached to the nose. Also shown, removable servo connectors, leading to the rest of the payload bay, which houses the actuators.

Actuator

Choosing the Actuator Style

Although fins seemed to be the best way to steer the rocket from the beginning, we decided to research methods of control before assuming this was the case. Generally, the considerations in choosing a method of control include: slew rate/response time, efficiency, maximum steering force, complexity and cost. In our case complexity and cost are primary considerations and slew rate is a secondary consideration. We compared 5 types of steering methods: two kinds of Thrust Vectoring Control (TVC), Fins, Control Gyroscopes and Momentum Wheels. The results of this comparison are summarized in table 1. For our complete detailed analysis, see Appendix B.

Method	**Complexity**	**Cost**	**Slew Rate**
TVC, exhaust deflection	Low, approx equal to fins	High due to expensive materials	Good, limited by servo speed
TVC, nozzle control	High, requires heavy mods on rocket	Low/Medium	Good, limited by servo speed
Fins	Low	Low, requires 4 servos and plastic fins	Good, limited by servo speed
Control gyroscopes	Medium/High, simpler with out-of-box gyro	Medium	Awful, meant for slow fine movements
Momentum Wheels	Medium	Low/Medium, would require decent motors	Awful, meant for slow fine movements

Table 1. A comparison of common rocket control methods. For a low-budget, high-speed rocket, adjustable fins are a clear winner.

We were not too surprised to find fins to be the most attractive options as they are very simple and relatively easy to add to our platform. They are also by far the cheapest and provide an acceptable slew rate.

Actuator Design

The requirements for our actuator are the ability to fit in the nose of our platform and the ability to provide a torque sufficient for fighting against the force of the wind. Servos come in packages as small as a centimeter long and provide relatively large torques while moving precisely to position setpoints. As such they were the natural choice for an actuator.

In order to decrease complexity we used one servo per control fin and directly attached the fin to the servo's horn. We used four servo plus fins for control, evenly spaced radially around the nose. Since our sensor uses four photoresistors in the same pattern this allows us to decouple the two dimensions of freedom the system has control over. This means we are able to write one simple control function based on principles derived in a 2d environment which we run two instances of to extend our rocket control into the 3 dimensions of the real world.

We were able to find servos with a 12 ms transit time at 60 degrees, which, given a conservative burn time of 1 second, should suffice (our actuator should never be moving 60 degrees at a time). These servos have a body length of under 1 inch and by mounting them from the outside of the nose we are able to fit all four servos at the same height in the nose. This provides a couple of benefits, including modularity of fins should any of them break and consistency between fins of the resulting force.

The servos we chose, like most, require a PWM input whose duty cycle determines the position. This works well for our system which revolves around an arduino which has pwm outputs available. Thus with the ability to interface with our processing unit as well as interface with the mechanical housing, our actuator was convenient and well-suited to our requirements.

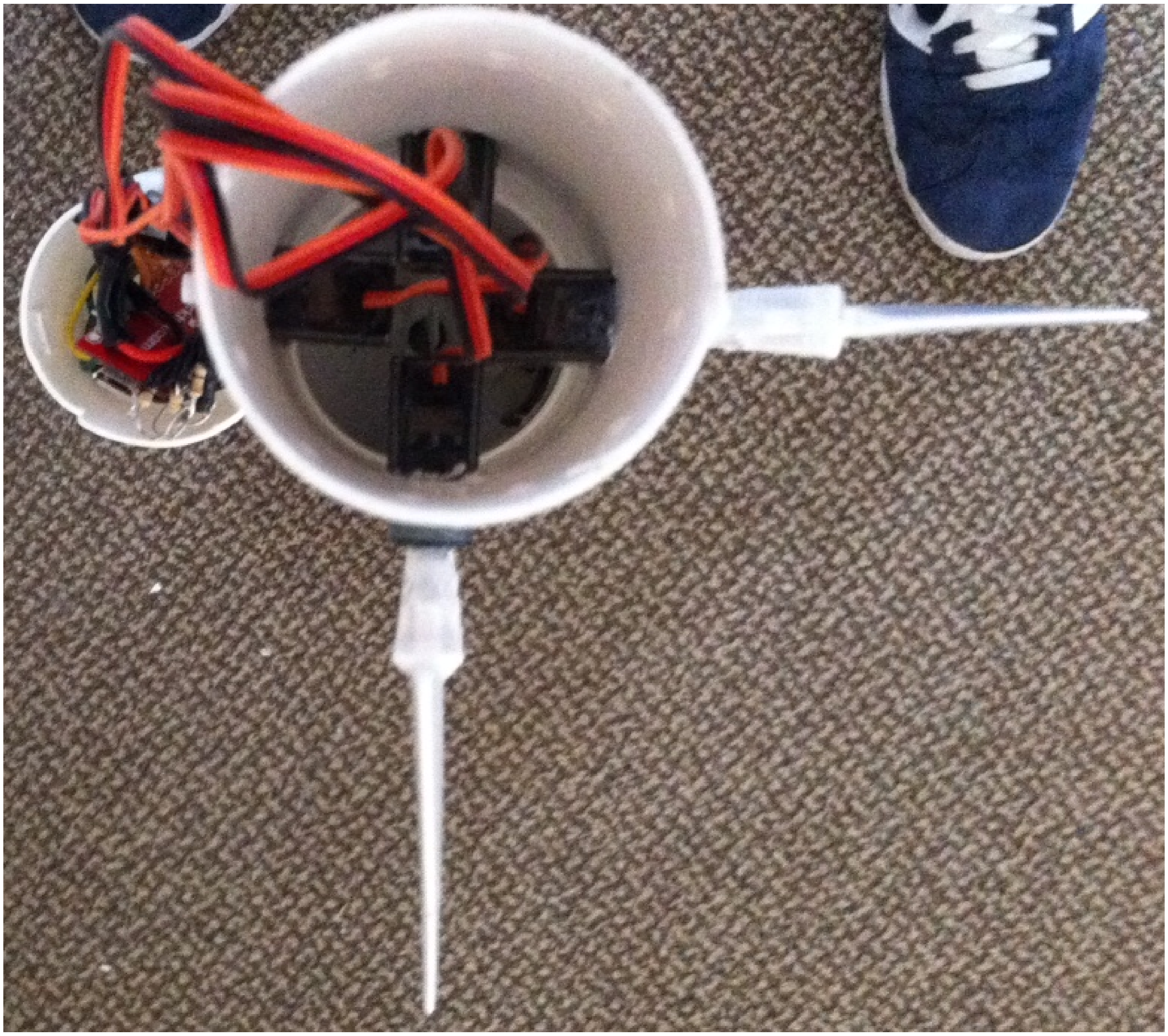

Figure 4. Servo Mounts. Four servos were mounted, level with each other. The horns stuck through the rocket body, where they were glued to nose fins. The nose fins are modified versions of the rocket's tail fins.

Figure 5. Rocket Nose Fin. The nose fins were build from the Estes Tomahawk's tail fins. Each fin was cut to reduce its area. The mount was also cut down and replaced with a servo horn, which snaps into a servo in the rocket body.

Dynamics and Simulation

Understanding Rocket Propulsion

We left the propulsion system of the rocket untouched in our design, as it met our requirements on its own. However, to best understand our model and the forces involved, we conducted background research on rocket propulsion physics. Using Robert A. Braeunig's overview of rocket propulsion, we gained a familiarity with the basic physics and design of both liquid and solid rockets, as well as various other basics of rocket propulsion.

This background research confirmed that we could leave our propulsion system unmodified. Additionally, it gave us confidence in our simulation, which models thrust as a force parallel with the rocket, with magnitudes taken from the D-12 datasheet [10].

The entirety of the research can be found in Appendix C.

Passive Rocket Stability

An important concept for developing a stable guided missile is the idea of passive stability.

That is, how rockets with no active control manage to fly straight up. This feature comes from the center of pressure being below the center of mass of the rocket (The center of pressure is the average position of the drag forces on the rocket, loosely correlated with the geometric centroid of the rocket, since drag is proportional to area). If the center of pressure is below the center of mass, then when the rocket rotates to the right of the wind, the center of pressure rotates to the left, and the resulting torque restores the rocket to be inline with the wind. The torque gets stronger as the rocket approaches perpendicular to the wind.

When the nose fins are pointed straight, we want to maintain the rocket's passive stability. That is, despite the nose fins' added surface area at the top of the rocket, the center of pressure should remain below the center of mass. However, a center of pressure that is nearer to the center of gravity is preferable in a steered rocket since this allows us to improve the response time. The sine of the fins' angle of attack and the sine of the rocket's angle of attack will not change proportionally as the rocket turns; the fin will increase at a lesser rate than the rocket. These two provide opposing torques that when balanced result in a steady turning radius for any fin actuation angle, at least in theory. We ignore this effect however, for 3 reasons. The first reason is that the angles for which this effect is relevant are larger than we think should happen with a good controller. The second is that we will see that this is not true in our linearized model. Finally, we are suspicious that other effects such as rotational drag and nonlinear lift will affect the system in this edge case enough that our understanding will break down. Therefore, as far as we will concern ourselves, when turning the fins enough to steer the rocket, we temporarily leave the realm of passive stability, requiring active control to maintain a bounded angle.

Characterizing the Aerodynamics

To get an accurate model of the forces on our rocket, we attempted to derive the forces that drag would impart on the rocket. We first researched aerodynamic drag, and then conducted tests in a wind tunnel. We used this knowledge and data to inform our simulation and our control parameters.

There are three main categories of aerodynamic drag. Wave drag comes from waves around a moving body, and is usually only relevant near or above the speed of sound. Lift-induced drag is the loss of forward velocity that is a side-effect of purposely redirecting air molecules perpendicularly to create lift. [11]. Because our rocket will be traveling at small angles of attack, its lift is small. The most significant form of drag on our rocket is parasitic drag, which comes from moving any body through a fluid. There are three types of parasitic drag: Skin friction is caused by the viscosity of the fluid, which creates friction along the entire surface of the rocket; Pressure drag comes from displacing the air in front of the rocket. It is proportional to the cross-sectional area of the rocket, but can be reduced by streamlining the rocket; Base drag comes from the vacuum behind the rocket as it moves forward. This creates turbulence around the rocket's tail, disrupting airflow, which worsens the other causes of drag. The gas ejecting from the burning engine reduces this effect, however, it is significant after powered-flight [9].

Each of these types of drag has empirically derived constants. Rather than determining individual constants and calculating forward, we used the Olin College wind tunnel to empirically measure the effects that are important to us, and lump the equations and their constants into useful functions of velocity and nose fin angle.

In order to get data for our model, we will divide drag into three measurable and useful effects. There is some force in the direction of the rocket's motion, which we will call parasitic drag. The force perpendicular to this, we will call lift. Finally, there is a torque exerted by drag, because the average point where the drag force is imparted is not located at the center of mass. Because we are trying to control our rocket's velocity, and the velocity depends on its angle, each of these effects is relevant. However, because our rocket is cylindrical and will not be traveling with large angles of attack, we decided to neglect the effect of lift on the rocket's velocity.

This leaves us with two values to calculate: the force that is directly subtracted from the rocket's speed, and the total torque due to drag on the rocket. Each of these depends on the rocket's speed, its orientation angle relative to its velocity bearing, and the angle of the nose fins of the rocket. To gather information about these effects, we mounted the rocket in the wind tunnel, shown in Figure 6. The mount consisted of a torque meter, mounted on the center of mass of the rocket. By taking data at different wind velocities, different rocket angles, and different nose fin angles, we characterized the torque due to drag on the rocket. Because this torque is a direct result of drag, if we can calculate the center of pressure of the rocket, we can use the same data to calculate the drag on the rocket.

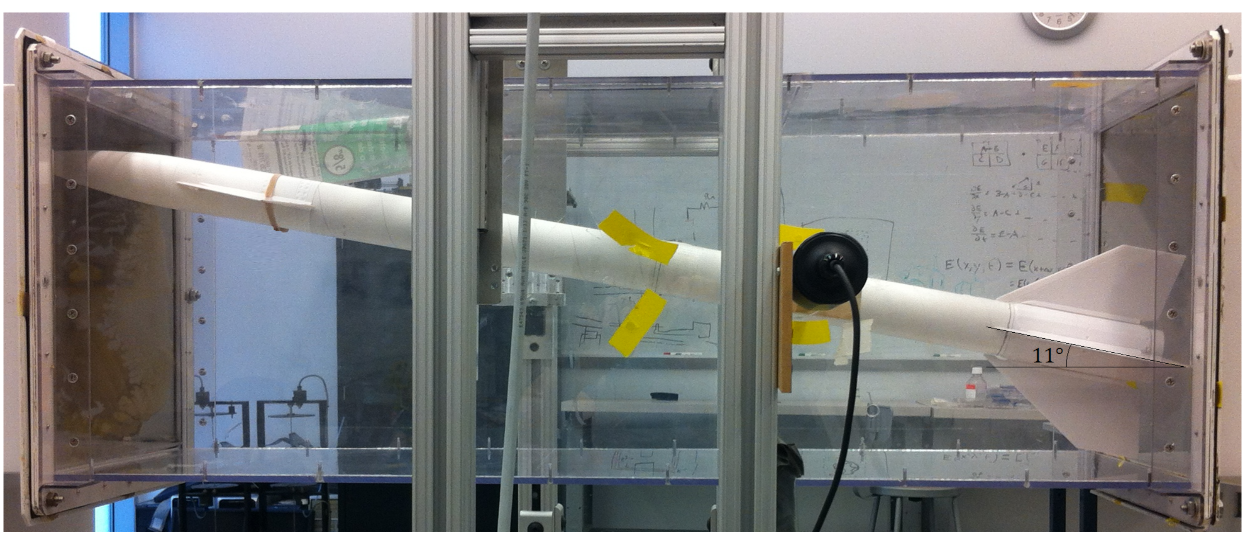

Figure 6. The rocket, with unactuated nose fins, mounted in the wind tunnel. With this setup,

the center of mass was too low, and as a result, the rocket was not passively stable. Significant data for this setup was not recorded.

Wind Speed	Torque (Kg-cm) 6°	Torque (Kg-cm) 10°	Torque (Kg-cm) 11°
17	-.5	-1.05	-1.3
21.8	-.75	-1.7	-2.1
26.6	-1.2	-2.75	-3.2
31.8	-1.7	-3.6	-4.3
37.8	-2.05	-5.0	-5.7
40.5	-2.4	-5.9	-6.6

Table 2: Torque without nose fins

This table shows how apparent wind speed and direction affect the torque on our rocket. As the table shows, the torques always push the nose of the rocket back towards the wind. This effect is stronger at greater angles and greater wind speeds.

Fin Angle	Torque (Kg-cm) 11° Rocket Angle
-20	-3.7
-15	-3.1
-10	-2.2
-5	-1.2
0	-.6
5	-.1
10	.3
15	1.1
20	1.5
25	2.2
30	2.8

Table 3: Torque with nose fins, Rocket Mounted at 11 degrees, Wind Speed 30mph

In this table, you can see that the restoring force and nose fin force are initially facing the same way. At about 6 degrees, the torque is zero, showing that for 6 degrees of nose fin angle, the rocket should eventually settle to 11 degrees off of the wind.

As these tables show, the effect of nose fin angle is significant, indicating that steering the rocket is possible. As the nose fins turn away from the wind, the net torque also increases in that direction. If the nose fins are straight, the force is towards the wind, indicating that passive stability is maintained. The total torques with and without nose fins are also of similar magnitude, which suggests that our resulting torques are of the correct magnitude to steer the rocket without sending it out of control.

Deriving the Equations of Motion and a Transfer Function

In order to develop our transfer function and ultimately the control system, we found and linearized the equations of motion of our system. The rocket has essentially 4 forces working on it: gravity, thrust, drag, and lift. Although we will soon show that we must ignore translational motion in order to linearize our system, we work out a complete set of equations for both our simulation, which we do not linearize, as well as for our internal understanding of the system so that we may be able to identify invalid assumptions better.

The equations describing the above mentioned four forces are written Equations 1 - 8. A diagram for visualizing the variables of interest is provided in Figure 7.

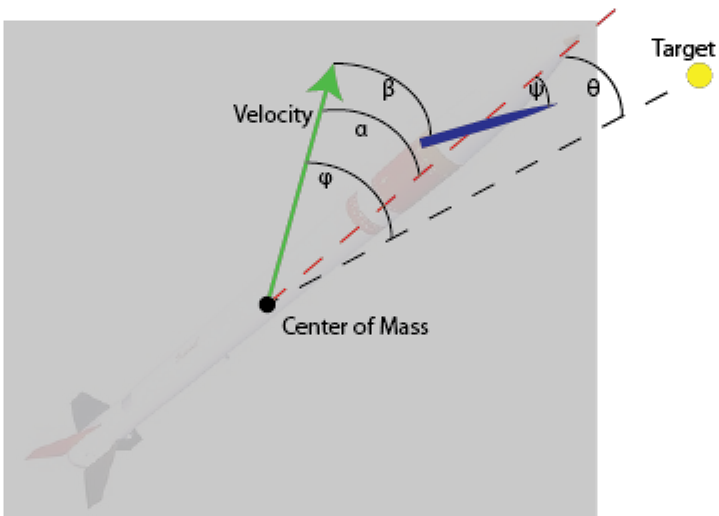

Figure 7. Rocket, with angles of interest. The blue arrow represents the rocket's nose fins, the green arrow represents its velocity.

$$F_{lift} = \alpha * \tfrac{1}{2}\rho|v|^2 K_{lr}$$ *Equation 1*

$$F_{drag} = sin(\alpha) * \tfrac{1}{2}\rho|v|^2 K_{dr}$$ *Equation 2*

$$F_{noselift} = \beta * \tfrac{1}{2}\rho|v|^2 K_{lf}$$ *Equation 3*

$$F_{nosedrag} = sin(\beta) * \tfrac{1}{2}\rho|v|^2 K_{df}$$ *Equation 4*

$$T = sin(\alpha)(F_{drag}D_{drag} + F_{nosedrag}D_{nose}) + cos(\alpha)(F_{li} \quad)$$ *Equation 5*

$$\frac{d^2\theta}{dt^2} = \frac{T}{I}$$ *Equation 6*

$$F_{centripetal} = F_{lift} + F_{noselift} + sin(\alpha) * F_{thrust}$$ *Equation 7*

$$\frac{d\varphi}{dt} = \frac{F_{centripetal}}{m|v|}$$ *Equation 8*

We implemented a forward Euler solver of these equations in MATLAB for initial values of angle, position and velocity. The full MATLAB code is provided in appendix D. In addition to storing a vector with the values of a variables of interest for plotting later, we draw a representation of the rocket in a MATLAB graph, a screen capture of which is shown in Figure 8. This simulation allows us to both validate our equations of motion and after linearizing and developing a control system, test our control system in a simulated environment.

One of the first realizations we had was the relative size of the thrust term in the equation for the centripetal force on the rocket (i.e. the net force responsible for changing the direction of the rocket's velocity). This allows us to make significant simplifications such that drag and lift only affect theta and phi is only affected by thrust from the engine. This also implies that our control function must have a settling time of less than the engine's burn time. Otherwise the rocket will not have steering ability. This actually fits with our intuition that we cannot steer very much without the engine providing the necessary turning force.

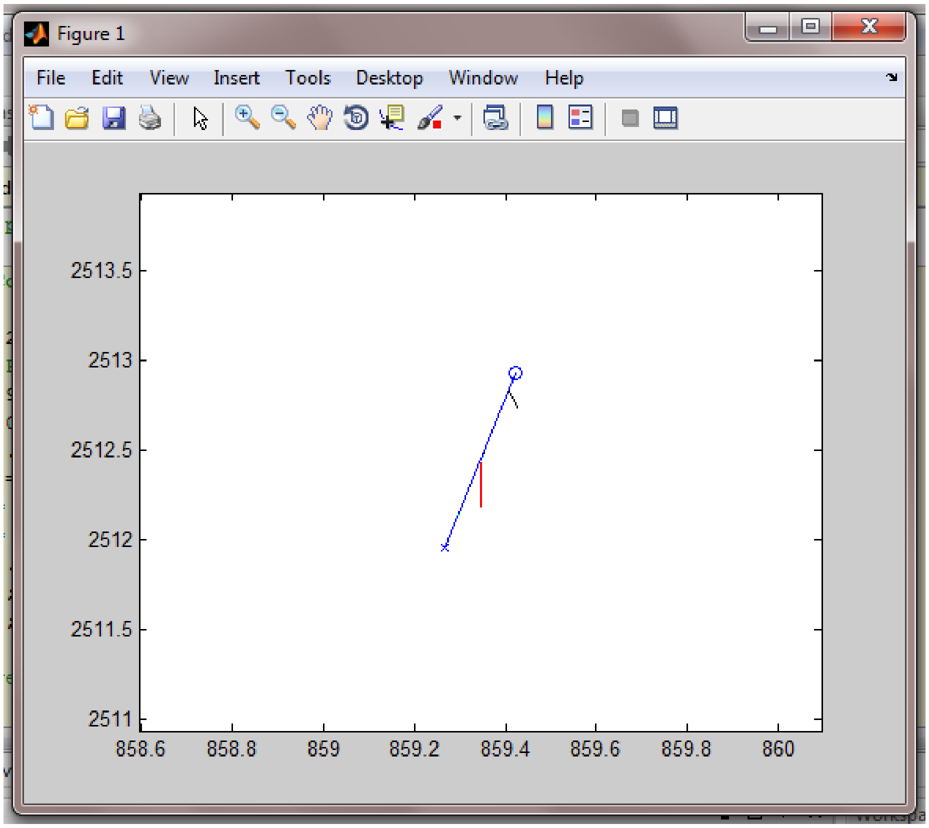

Figure 8. Screen capture of rocket with gravity and drag forces shown.

Our input to the system is a desired phi, which represents the angle of the velocity vector which is why we will work in a polar coordinate system. This also means that we need to develop our linearized model around operating points for translational terms such as the thrust vector and the magnitude of velocity. What we do for these is assume a constant value, develop a control system around this value and if we would like to adjust for the real values we can implement our control system such that these values may dynamically change the control function. In other words, our control system should have dynamic parameters that correspond to these values. We then need to perform a feed-forward estimation of the values of thrust and velocity to provide to our control function.

We also make the small angle assumption around the sine and cosine terms (sin(x) = x and cos(x) = 1). Although this assumption may vary by as much as 10% near points that we expect our rocket may operate under (any of our angles may be as much as 45 degrees) so long as the signs of the angle values and their derivatives are correct (i.e. sin is positive increasing from 0 to 45, negative decreasing from 0 to -45 and cos is positive for -45 to 45) this assumption will not dramatically change the results of our control system.

The last simplification we make is the grouping of the aerodynamical terms. These include the coefficients of drag and lift, location of the centers of drag, lift and the nose fins, and the area of the aerodynamic surfaces. These are lumped into two constants: Kr for all terms related to the rocket without control surfaces and Kf for all terms related to the nose fins. These coefficients are gains on alpha and beta respectively before being multiplied by the blocks that convert this into a torque and ultimately a rotational acceleration of the rocket.

After linearizing and simplifying, the complete block diagram of our system is shown in Figure 9. Since our sensor measures theta yet our desired input is phi, we manipulated the block diagram to better fit a typical control system topology. The resulting block diagram and corresponding transfer functions are shown in Figure 10 and table 4 respectively.

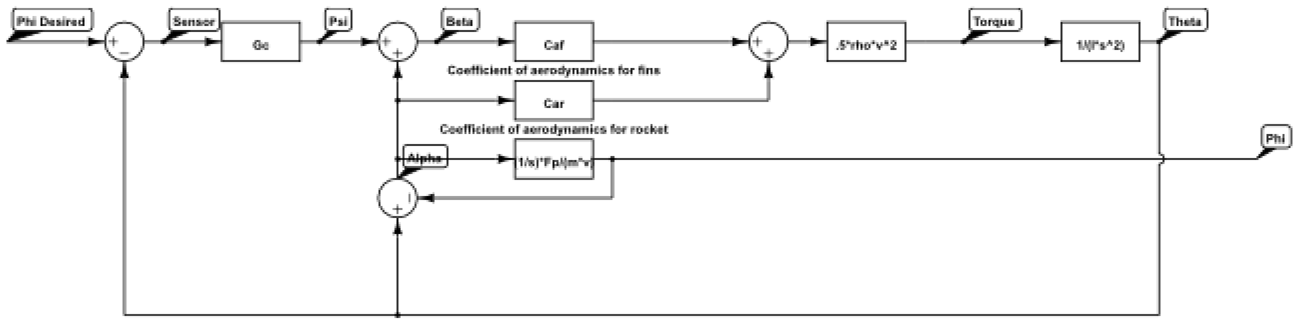

Figure 9. Block Diagram of Rocket. This block diagram shows the physics and feedback of our system.

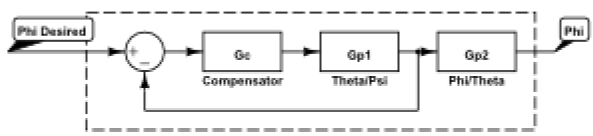

Figure 10. Reorganized Block Diagram. This block diagram shows more clearly what is of interest in our system. It is evident here that phi has no feedback, but is estimated based on our system model and theta, which is controlled via feedback.

$Gp1 = \frac{\frac{\rho v^2}{2I} K_f (s+\frac{F}{mv})}{s^3+\frac{F}{mv}s^2-\frac{\rho v^2}{2I}(K_f+Kr)s}$
$Gp2 = \frac{\frac{F}{mv}}{s+\frac{F}{mv}}$
$G = Gp1 * Gp2 = \frac{\frac{F\rho v}{2mI} K_f}{s^3+\frac{F}{mv}s^2-\frac{\rho v^2}{2I}(K_f+K_r)s}$
$H = \frac{1}{Gp2} = \frac{s+\frac{F}{mv}}{\frac{F}{mv}}$

Table 4. System Transfer Functions

Control System Design

Proportional Navigation

Proportional Navigation (PN) is a commonly applied guidance law in heat seeking missiles. PN is particularly applicable in situations where the target is moving. PN takes advantage of the fact that two objects will collide if the line of sight between the two objects does not change orientation. Based on this fact, PN says that the acceleration of the rocket perpendicular to velocity (what we call the “centripetal acceleration” above) should be proportional to the rate of change of the line of sight angle. This results in a very good control system for short range rockets.

Ostensibly, our control system would do well to be designed with PN’s principles. However we found a significant problem with our application which is the fact that our line of sight with the sun remains essentially unchanged as the rocket moves. This is simply due to the fact that the sun is many millions of miles away. If our system were to be targeting a shorter range target, or actually traveling to the sun, PN may be a reasonable approach however since this is not the case, we need to develop our own approach based on the equations we found earlier.

Uncompensated System

We first analyze the Bode plot of the uncompensated system (psi equal to theta). The resulting closed loop plot is shown in Figure 11. Clearly the system, although stable, could do much better with a well designed compensator. It seems that while we would like to increase or at least maintain the bandwidth of the system, lead compensation is not particularly useful since either the phase margin is <90 or we will be pushing our gain margin, which is already extremely small. Thus we conclude that actually lag compensation in order to both increase the gain at lower frequencies and increase the gain margin would be a good form of compensation. Our future steps for improving the system’s response would be to implement such a compensator in the actual system and test its merits in the real world.

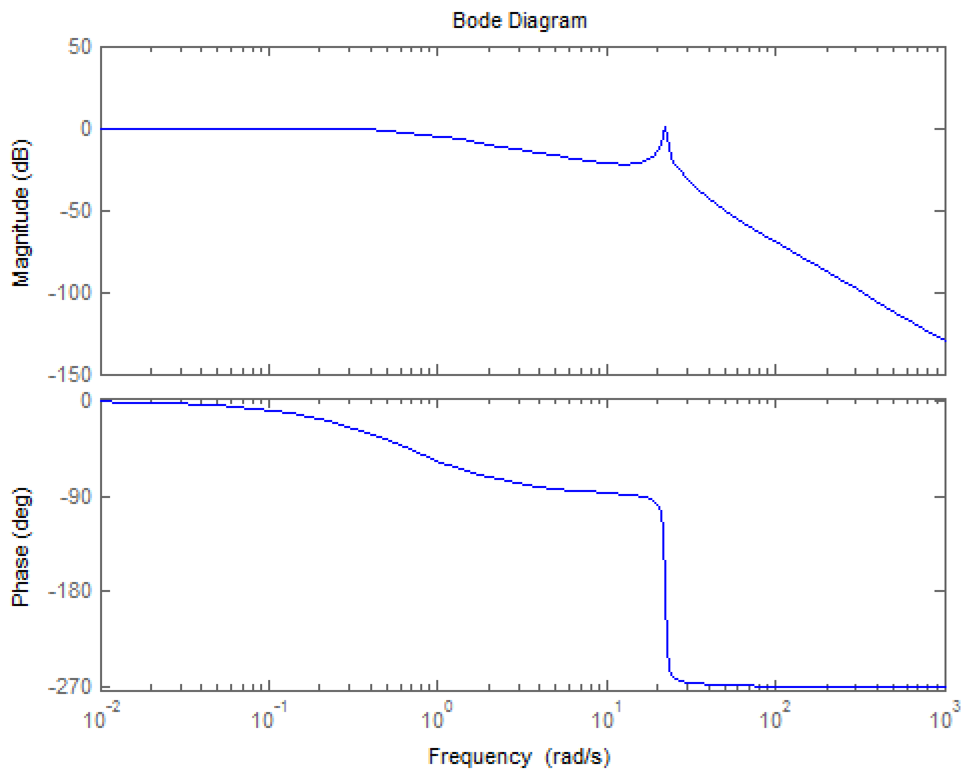

Figure 11. Bode Plot of the Uncompensated Closed Loop Transfer Function.

PD Compensated System

As a simpler pass at compensation, we worked on a simple PD (proportional and derivative) controller to control overshoot and oscillations. Using Kd=Kp=1, the resulting bode plot is shown in Figure 12. This simple compensator successfully increased the problematic gain margin while giving us room enough to significantly increase the gain.

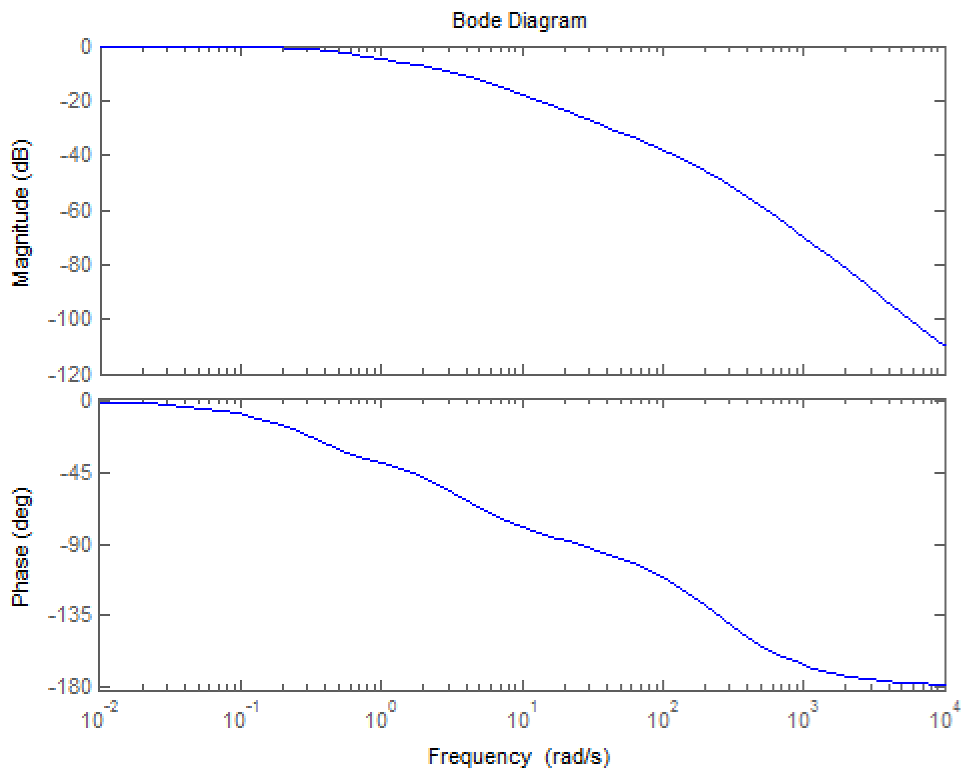

Figure 12. Bode Plot of the PD-Compensated Closed Loop Transfer Function.

Results

Control Algorithm

Because it is difficult to measure data about the flight of the rocket, we chose to keep our tests simple, with more easily interpreted results. This meant starting out with only basic stability tests, learning as much as possible from them, and then moving onto a full-system test.

Building on our tests, in the wind tunnel, which suggested that the rocket would be just barely passively stable with its fins straight, we launched the rocket with the simple mission of steering in an S-shaped curve. With a 1.7 second burn time, we aimed for .5 seconds of vertical flight, followed by switching the fins between 25 and -25 degrees from center every .5 seconds. We started with steering in one plane, leaving one pair of fins static.

We had moderate success with this test. The rocket had a stable start. At the expected time, the fins steered the rocket as hoped, giving it a horizontal flight path. However, when the time came to reverse rotation, the reversal was very slight. This left the rocket flying horizontally until the next reversal, which caused it to steer below horizontal, and eventually into the ground. We attributed this failure to either an asymmetry in mechanical design, or simply the lack of enough wind tunnel data and integration of that data into our simulation. What is heartening is that the control did have an effect, and with more tests and adjusting gains on the fin angles, it could be tuned to be controlled.

Structural Integrity

One of the large challenges of this project was modifying the rocket in such a way that it could be controlled, using inexpensive parts. The use of Duco Plastic Cement was effective at holding the fins on the rocket during flight, however, as could be expected, when the rocket nose-planted, the front fins were knocked off the rocket. This failure mode makes repeated tested very difficult, due to the long gluing process required after the failure. A possible modification would be to use press fit pins to hold the fins onto the servo horns, however, this risks snapping the servo or servo horns themselves.

As for the electronics bay, nothing was harmed during the crash-landings of the rocket, and the modifications to hold the electronics did not interfere with the parachute recovery process.

Conclusion

This project suggested that it is indeed possible to build a low-budget guided rocket. We were able to get accurate values from our sensor to locate the sun. By adjusting the center of mass and center of pressure, we were able to adjust the dynamics of our rocket to create the right torques to steer the rocket. We were also able to create an accurate model to help us understand the dynamics. However, without enough test flights to sufficiently optimize and validate our model, we were not able to strike the fine balance between control and stability that is required to steer a rocket.

Acknowledgements

Drew Bennett
Chris Lee
Kent Lundberg
Jessica Townsend

Works Cited

[1] http://exploration.grc.nasa.gov/education/rocket/rktstab.html
[2] http://www.ibiblio.org/pub/archives/rec.models.rockets/ARTICLES/guidanceFAQ.txt
[3] http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1080&context=smallsat

[4] Sutton, George. Biblarz, Oscar. Rocket Propulsion Elements, 7th Edition. Wiley, 2001.
[5] http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19700032883_1970032883.pdf
[6] https://sites.google.com/site/airwavershr/Home/guided-rocket
[7] Zarchan, Paul. Tactical and Strategic Missile Guidance. Progress in Astronautics and Aeronautics, Volume 124. 1990.
[8] http://www.braeunig.us/space/propuls.htm
[9] http://dynlab.mpe.nus.edu.sg/mpelsb/mdts/Aero%20L3n.pdf
[10] http://www.nar.org/SandT/pdf/Estes/D12.pdf
[11] http://www.grc.nasa.gov/WWW/k-12/airplane/ldrat.html
[12] ATK Space Propulsion Products Catalog, May 2008, p. 30
[13] Estes 2011 Catalog www.acsupplyco.com/estes/estes_cat_2011.pdf

Appendices

Appendix A. Legality

Hobby rocketry has several restrictions, defined by the National Association of Rocketry, and listed at: http://www.nar.org/NARmrsc.html. Most of the requirements are met by our system from our lack of modification.The sections below are relevant to the components of the rocket that we plan to modify, or the flight-test methods that we plan to use.

1. Materials. I will use only lightweight, non-metal parts for the nose, body, and fins of my rocket.

5. Launch Safety. [...] If I am uncertain about the safety or stability of an untested rocket, I will check the stability before flight and will fly it only after warning spectators and clearing them away to a safe distance.

6. Launcher. I will launch my rocket from a launch rod, tower, or rail that is pointed to within 30 degrees of the vertical to ensure that the rocket flies nearly straight up, and I will use a blast deflector to prevent the motor's exhaust from hitting the ground. To prevent accidental eye injury, I will place launchers so that the end of the launch rod is above eye level or will cap the end of the rod when it is not in use.

7. Size. My model rocket will not weigh more than 1,500 grams (53 ounces) at liftoff and will not contain more than 125 grams (4.4 ounces) of propellant or 320 N-sec (71.9 pound-seconds) of total impulse.

8. Flight Safety. I will not launch my rocket at targets, into clouds, or near airplanes, and will not put any flammable or explosive payload in my rocket.

Appendix B. Overview of Stabilization and Steering

Instability in a rocket generally means oscillations or uncontrolled rotation during flight. Instability is highly undesirable as it increases drag unnecessarily, can cause problems for onboard sensors and can lead to tumbling which will mean failure for getting into orbit or hitting a target.

A rocket can be made passively stable in pitch and yaw by keeping the center of gravity in front of the center of pressure. Usually this is accomplished with fins at the back of the rocket to bring the center of pressure toward the tail. Alternatively weight can be moved or added to the front to bring the center of gravity toward the head. For stability in roll (excessive roll instability can cause oscillations along other axes as well) passive rollerons can be used.

Active stabilization methods can be used alongside passive methods. Common active methods, all of which we explore in this lab, include actuated control surfaces, directional control of thrust and gyroscopic methods. For any active stabilization method an appropriate sensor must be included to determine the error from the desired orientation.

Although passive methods are simple and easy to implement, they inherently fight against the ability to steer the rocket. For this reason, an actively steered rocket should be passively stable but only barely (i.e. the center of pressure barely behind the center of gravity). This optimization can be particularly difficult to design for because the center of pressure and the center of gravity may change during flight, particularly the latter.

In addition to the above considerations, a rocket designer must take into account the type of engine, how sharp the turns must be, cost, complexity and efficiency. In this lab we will explore a number of the most common stabilization and steering methods and review their respective advantages and disadvantages in the above categories and comment on any nuances related to their application.

Table: summary of researched stabilization methods

Stabilization Method	**Common Applications**	**Resources**
Thrust Vectoring	Big rockets; jet planes;	Chapter 16 of Rocket Propulsion Elements http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19700032883_1970032883.pdf (compares 3 methods we already liked. It's from 1970 but hey so is the sidewinder.)
Fins	Simple, low cost applications; Fish;	http://exploration.grc.nasa.gov/education/rocket/rktstab.html http://en.wikipedia.org/wiki/Center_of_pressure_(fluid_mechanics)#Aircraft_aerodynamics

		http://www.ibiblio.org/pub/archives/rec.models.rockets/ARTICLES/guidanceFAQ.txt
Control Gyroscopes	Satellites;	http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1080&context=smallsat
Momentum wheels	Satellites;	http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1080&context=smallsat

Thrust Vectoring:

Thrust Vectoring is an umbrella term for any method which can adjust the net direction of the rocket's thrust. Thrust Vectoring Control (TVC) mechanisms fall into 4 categories: mechanical adjustment of the nozzle, exhaust deflection, fluid injection and auxiliary thrusters [4]. As a whole TVC methods are particularly suited for large systems that require rapid and powerful adjustments. Although they fall on the complex and more difficult side, they can operate outside of the atmosphere (unlike control surfaces) and can provide larger forces than systems using gyroscopic effects. A notable disadvantage is that TVC mechanisms are only effective while the propulsion system is operating.

Nozzle directional control methods

This category includes systems that have some form of hinged nozzle that can be oriented as needed. In liquid propellant rockets usually the combustion chamber together with the nozzle is rotated while systems in solid propellant rockets rotate just the nozzle. These methods see minimal losses in efficiency at low angles away from directly vertical (known as the deflection angle) and have the largest range of deflection angle. The primary disadvantage however is the large actuation forces required to orient the nozzle and sometimes combustion chamber, especially when a fast slew rate is desired

Exhaust deflection

Exhaust deflection methods use heat resistant components at the end of the nozzle that divert the exhaust in order to adjust the direction of the thrust. These mechanisms are relatively simple and easy to implement however this comes at the expense of efficiency since they necessarily obstruct the flow thus decreasing thrust. Systems may use 'jet vanes', 'jet tabs' or a 'jetavator' to divert the exhaust.

Fluid injection

This category of systems works by injecting fluid into the side of nozzle. Essentially this flow displaces the exhaust flow in the direction that more thrust is desired. Because valves respond quickly this can provide good slew rates with little loss in efficiency. One of the primary issues with fluid injection systems is that the better performing liquids are toxic which is problematic for handling and maintenance safety. Furthermore, the effective deflection angle is relatively small for these mechanisms.

Auxiliary thrusters

Essentially these mechanisms use propulsion systems (often smaller rocket engines) at different angles which can provide a sideward thrust, turning the rocket. They are the most complex to implement but very efficient and depending on the implementation can be very compact. Overall these methods have mostly fallen out of favor.

Fins:

Almost all rockets and missiles use fins. It is possible to create a stable finless rocket by other means of stability, but the effectiveness, low weight, and low power use, are almost always worth the small downside of added drag (from RPE ch 16). The primary utility of fins are passive/active stability and guidance. In addition, if roll is desired, fins can be fixed at an angle to cause the rocket to spin, which creates gyroscopic restoring forces and rotates any imperfections, creating an effect as if one flaw was radially symmetrical. (RPE 4.10)

Passive Stability

The passive stability created by rear-mounted fins comes from simply moving the center of pressure backwards. The center of pressure is the point at which torque is applied by aerodynamic forces, which is dependent on where drag is applied. The fins make sure that drag is applied in the back of the rocket, so that if the back starts to rotate towards the apparent wind, it is pushed back away from the wind until the fins are parallel to the wind vector. Note that fins align the rocket only with the apparent wind, so the upward velocity of the rocket must be much greater than the wind to cause the rocket to face straight up.

Active Stability and Guidance

Active fins simply rotate to create lift in the direction that torque should be applied. These fins can go on the front or the back of the rocket, and desired torques are in opposite directions for each case. However, there are other differences between the two cases.

Active fins in the back are more powerful than those in the front when it comes to turning upwind, because to steer upwind, the fins turn downwind- the same direction as the rocket. The rocket's angle and the fins angle relative to the rocket will sum to produce the angle of attack of the fin. This is perhaps best illustrated by the opposite case: front-mounted fins turn upwind to steer upwind, but since the rocket is facing downwind, there is a point where, even with some rotation, the fins are merely facing straight into the wind, and producing no lift. This tradeoff depends on the situation- sometimes it is preferable to have a less touchy actuator when you're hurtling through the sky and your motors are not fast enough to control large oscillations.

Another concern with front-mounted fins is that, if they fail to actuate, all they are left doing is bringing the center of pressure forward. For this reason, it is preferable to have at least passive fins in the rear no matter what- however, there is a tradeoff between how passively stable the rocket should be and how easy it is to steer.

In addition to aerodynamic considerations, there are construction and practical concerns, and front-mounted active fins tend to be better in these respects. This is because

1. They are at a safe distance from the rocket motors

2. They are not squeezed in with the rocket motors
3. Wiring is smaller and simply when the fins are closer to electronics and sensor payloads
4. The additional weight in the front moves the center of gravity forward, helping passive stability [2]

Gyros and Momentum Wheels:

The ability of gyroscopes and momentum wheels to store angular momentum is useful in many aspects of rocket control. Because the gyroscopic force is perpendicular to the acceleration on the gyroscope, these tools cannot be used for passive stability. However, both of these tools can be actuated to exchange momentum between itself and the aircraft body.

Each of these tools uses a different method to impart a torque on the the aircraft body. In the case of a momentum wheel, the wheel is accelerated, causing an opposite torque on the aircraft. For a gyroscope, the already rotating gyroscope is rotated, and the restoring force creates a torque on the aircraft. Because these systems are relatively complicated and large for the amount of torque that they create, they are much less common in rockets than in satellites, where their long lifetimes and energy efficiency can be preferable to external thrusters, particularly for the slow, fine movements that are often necessary on satellites.

Shown below is a graph of which method is preferable given different parameters. The spacecraft moment of inertia is important because as torque requirements increase, gyroscope power use goes up several factors slower (increasing the ship’s edge length by 2 increases its moment of inertia by 32, requiring 2x the power from a gyroscope and 32x the power from a momentum wheel.) The slew angle makes some difference, due to mechanical inefficiencies in CMGs (Control Moment Gyroscopes) that hurt gyroscopes when large torques are not required, but rather large angular momentum capacity.

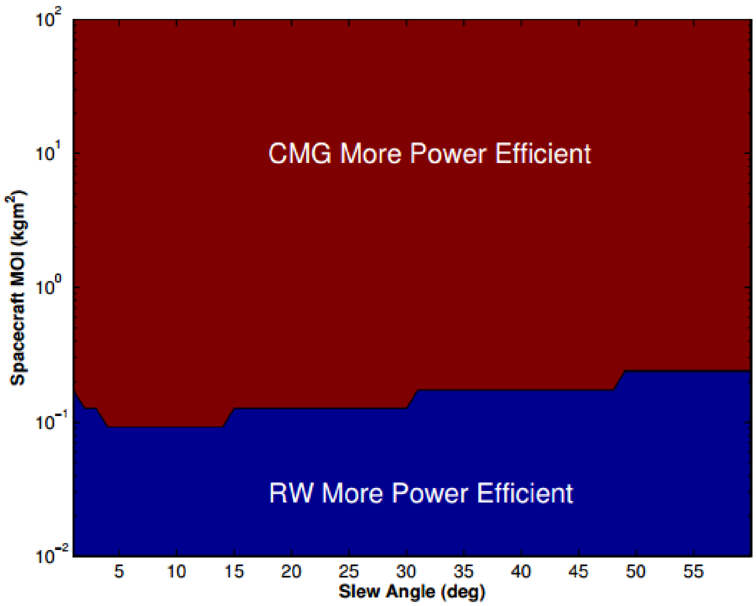

Figure from [3]

Gyroscopes are also useful in rockets as sensors. They can be used to transmit rotational data to a computer, as well as being used as direct controllers of actuators such as fins. [3]

Appendix C. Fundamentals of Rocket Propulsion

We left the propulsion system of the rocket untouched for our system, as it met our requirements on its own. However, to best understand our model and the forces involved, we conducted background research on rocket propulsion physics. Using Robert A. Braeunig's overview of rocket propulsion, we gained a familiarity with the basic physics and engine design of liquid rockets, the geometry of solid rockets, and various other basics of rocket propulsion.

Most large rockets use multi-part liquid fuels. The design of these systems makes use of the pressure on the rocket nozzle that results from heating the fuel and ejecting it at high speed from the rocket. Because momentum is conserved in the system, the momentum added to the rocket is equal to the momentum of the ejected fuel as it leaves the rocket. The measure of efficiency for a rocket engine, related to its exhaust velocity, is specific impulse. Specific impulse is the ratio of the force exerted on the rocket to the mass flow rate of propellant. In combustion reactions, high specific impulse is achieved through high temperature and pressure and low molecular weight. Electric thrusters can achieve much higher specific impulses, however, the total thrust that they can produce is less. The specific impulse of our Estes D-12 rocket engine is 81.86s, as compared to large solid rocket boosters (250s), and ion thrusters (21,400) [12, 13].

Liquid rocket engines contain three sections: the chamber, the nozzle, and the injector. The chamber is where the fuel reacts. It is designed to be able to handle the high pressures and temperatures of this reaction, and feed the hot fuel into the nozzle. The shape of the chamber is generally spherical or cylindrical. The nozzle turns the heat energy of the gas into mechanical energy. It begins with a small “throat” (Figure 1.4), which pressurizes the gas. As the gas moves to the exit of the nozzle, it is accelerated and depressurized. The goal is for the gas to reach outside pressure at the exit of the nozzle, so that all of the gas’s expansion is done in the nozzle, and it is restricted to expanding parallel to the rocket, exerting all of it’s force in that direction. The desire to meet the outside pressure is a main reason that different nozzle geometries are used in space than in the lower atmosphere for this reason. Historically, cone-shaped nozzles were common, however bell-shaped nozzles have been found to maximize efficiency. Exact geometry is largely determined by the fuel-type used.

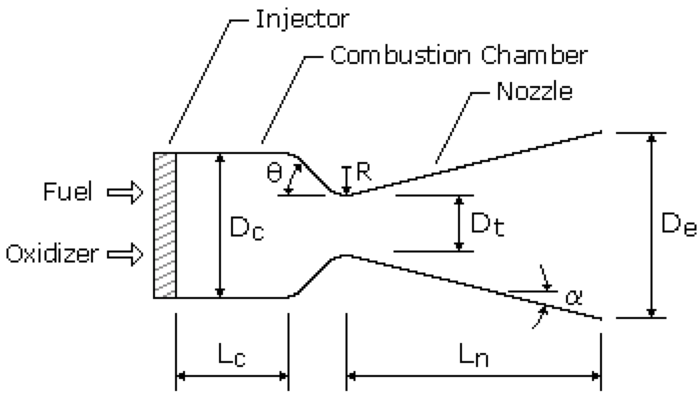

Figure 1.4

The most important component for efficiency is the fuel-injector, which can cause huge variances in the systems efficiency. Well-designed systems can convert nearly 100% of the combustion energy into thrust. The injector's goal is to provide a good mixture, fine atomization, and avoid combustion instability, where pressure oscillations in the chamber are amplified, causing damage and inefficiency.

Liquid rockets use many variations of power cycles to make use of their fuel. Most power cycles are designed to put fuel into the combustion chamber without wasting much fuel accomplishing this. The systems include valves to control the ratios, but often pumps are required to transport the fuel. These pump can be powered by small amounts of the rocket fuel, which can then be recycled into the engine. The power cycles also employ clever tactics to recycle heat and unburned exhaust, as well as to cool the engine with unburnt fuel while simultaneously

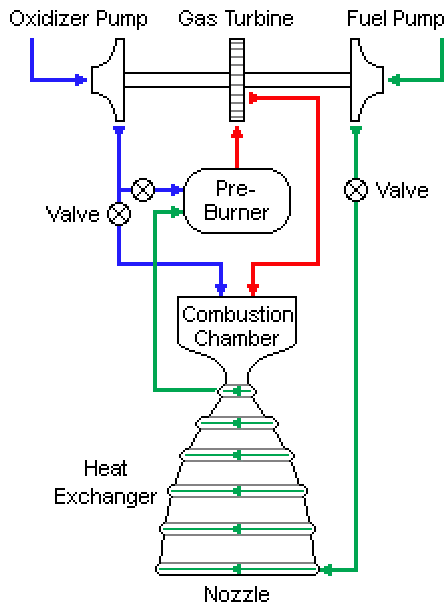

Fig. 1.9 - STAGED COMBUSTION

preheating the fuel for easier burning.

Solid Rocket Motors

Solid rockets, like the ones used in hobby rockets, are often used for their simplicity. However, they have the downsides of not being as efficient, and being uncontrollable once ignited. There are several types of solid rocket fuel grain geometries. The amount of thrust depends on the amount of fuel burning, which depends on the surface area that is exposed. One type of rocket uses a flat surface that constantly burns from the nozzle upwards. The more common style is to have a prismic area of empty space in the engine, whose shape determines the thrust evolution over time of the rocket, as the shape changes. The rate of surface recession is called the "burn rate." Some rocket geometries and their thrust curves are shown in Figure 1.14. Solid rockets use similar nozzles to liquid rockets (wp solid-fuel rocket).

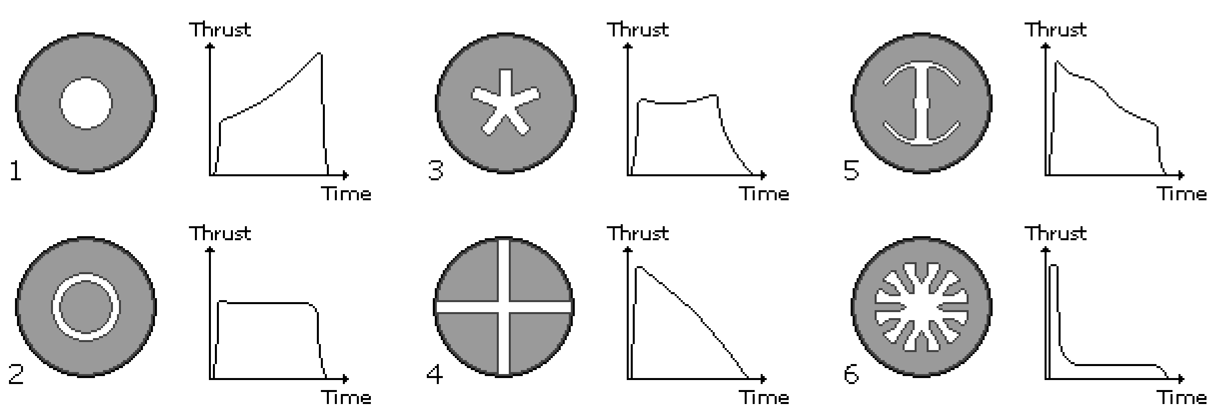

Figure 1.14

The final type of propulsion system we studied was hydrazine monopropellant engines. These engines do not have enough power to perform large maneuvers, but are easy to control and store. For these reasons, they are very common in spacecraft attitude and velocity adjustment, and were also used to control the Mars Curiosity Rover's descent and landing on Mars. The process consists of hydrazine, a highly unstable compound, being released onto an iridium catalyst that accelerates its decomposition to a rate which produces high enough temperatures to generate a large amount of thrust through a nozzle.

Appendix D. MATLAB Code

```
-------- START 'simulate1.m' --------
%notes: projected altitude = 229m,

%% Set Constants
g = 9.8; %m/s^2
rho = 1.2041; %kg/m^3
%Rocket Parameters:
rlen = .986; %m
rdia = .046; %m
rmass = .26; %kg
rmoment = rmass*rlen^2/12; %kg*m^2
rareat = pi*rdia^2/4; %m^2 (top area)
rareaf = rlen*rdia; %m^2 (frontal area)
farea = .046^2; %m^2 (fin area)
Cd = .08; %unitless (pressure only)
Cl = .05; %unitless (lift coefficient)

%positive means toward nose, relative to Cg
Dn = .5; %m (dist from Cg to nose)
Dp = -.3; %m (dist from Cp to Cg)
Dl = .05; %m (dist from Clift to Cg)
Df = .4; %m (dist from Cg to nose fins)

D12Curve = [0 0.049 0.116 00.184 00.237 00.282 00.297 00.311 00.322 00.348 00.386 00.442
0.546 0.718 0.879 1.066 1.257 1.436 1.590 1.612 1.650;
        0 2.569 9.369 17.275 24.258 29.730 27.010 22.589 17.990 14.126 12.099 10.808 9.876
9.306 9.105 8.901 8.698 8.310 8.294 4.613 0.000];

D12Curve(2, :) = D12Curve(2, :) * 1/2;

%% PrepareSim
axes = [0 20 0 230];
rel_axes = [-1 1 -2 2]*3/4;
```

```
ts = 0:.01:20;

start_theta = .15; %rad; 0 = up, pi/2 = right, -pi/2 = left
sprintf('Start angle: %.2f', start_theta);
start_omega = 0; %rad/s
start_pos = [0 0]; %[m, m]
start_vel = 0; %m/s
start_phi = 0; %rad

%% Run Sim
theta = start_theta;
omega = start_omega;
pos = start_pos;
vel = start_vel;
phi = start_phi;
psi = 0;
target = 0;
perr = 0;

pt = 0;

Fgravity = [g*rmass, pi];
Kthrust = [0, 0]; %[+thrust(t), +theta]
Kdrag = [.5*rho*Cd, pi]; %[*A*vel^2, +phi]
Klift = [.5*rho*Cl, pi/2]; %[*A*alpha*vel^2, +phi]
angles = [];
vels = [];
for t = ts

    dt = t - pt;

    alpha = wrapToPi(theta-phi);
    cosa = cos(alpha); sina = sin(alpha);

    Fthrust = Kthrust + [interpolate(D12Curve, t), theta];

    vel2 = vel^2; %speed up
    Fdrag = Kdrag .* [abs([cosa, sina])*[rareat; rareaf]*vel2, 1] + [0, phi];
    if(abs(alpha) > pi/4)
        Flift = Klift .* [0, 1] + [0, phi];
```

```
else
   Flift = Klift .* [rareaf*alpha*vel2, 1] + [0, phi]; %may have negative magnitude
end

err = target-theta;
psi = controlFunc(psi, err, (err-perr)/dt, 0);
perr = err;
beta = wrapToPi(alpha+psi);
Ffindrag = Kdrag .* [sin(beta)*farea*vel2, 1] + [0, phi];
if(abs(beta) > pi/3)
   Ffinlift = Klift .* [0, 1] + [0, phi];
else
   Ffinlift = Klift .* [farea*beta*vel2, 1] + [0, phi];
end

%Fc = centripetal, Fl = linear (ie direction of phi)
[Fcnet, Flnet, Tnet] = split(...
   convertPolarForce(Fgravity, [0 pi], phi) +...
   convertPolarForce(Fthrust, [0 theta], phi) +...
   convertPolarForce(Fdrag, [Dp theta], phi) +...
   convertPolarForce(Flift, [Dl theta], phi) +...
   convertPolarForce(Ffindrag, [Df theta], phi) +...
   convertPolarForce(Ffinlift, [Df theta], phi));

vel = vel + Flnet/rmass*dt;
if(vel < 0)
   vel = -vel;
   phi = phi + pi;
end
if(vel ~= 0)
   phi = wrapToPi(phi + atan(Fcnet/rmass*dt/vel));
end
omega = omega + Tnet/rmoment*dt;
theta = wrapToPi(theta + omega*dt);

pos = pos + vel*[sin(phi) cos(phi)];

pt = t;

%% plot real-time
```

```
    %rocket
    plot([Dn, Dn-rlen]*sin(theta)+pos(1), [Dn, Dn-rlen]*cos(theta)+pos(2));
    hold on
    plot((Dn-rlen)*sin(theta)+pos(1), (Dn-rlen)*cos(theta)+pos(2), 'x');
    plot(Dn*sin(theta)+pos(1), Dn*cos(theta)+pos(2), 'o');

    %forces
    Fscale = .1;
    plot([0, Fscale*Fgravity(1)*sin(Fgravity(2))]+pos(1), [0,
Fscale*Fgravity(1)*cos(Fgravity(2))]+pos(2), 'r');
    plot([0, Fscale*Fthrust(1)*sin(Fthrust(2))]+pos(1), [0,
Fscale*Fthrust(1)*cos(Fthrust(2))]+pos(2), 'r-');
    plot([Dp, Dp]*sin(theta)+[0, Fscale*Fdrag(1)*sin(Fdrag(2))]+pos(1), [Dp, Dp]*cos(theta)+[0,
 Fscale*Fdrag(1)*cos(Fdrag(2))]+pos(2), 'g');
    plot([Dl, Dl]*sin(theta)+[0, Fscale*Flift(1)*sin(Flift(2))]+pos(1), [Dl, Dl]*cos(theta)+[0,
Fscale*Flift(1)*cos(Flift(2))]+pos(2), 'k');

    %fins
    Lfin = -.1;
    plot([Df, Df]*sin(theta)+[0, Lfin*sin(theta+psi)]+pos(1), [Df, Df]*cos(theta)+[0,
Lfin*cos(theta+psi)]+pos(2), 'k');

    hold off
    %axis(axes)
    axis([pos(1) pos(1) pos(2) pos(2)] + rel_axes)
    drawnow

    pause(.05)

end
-------- END ‘simulate1.m’ --------
-------- START ‘convertPolarForce.m’ --------
function [res] = convertPolarForce( F, loc, basis )
%F = [mag in N, radians], loc = [mag in m, radians], basis = radians
%   Detailed explanation goes here

    T = loc(1)*F(1)*sin(F(2)-loc(2));
    Fc = F(1)*sin(F(2)-basis);
    Fl = F(1)*cos(F(2)-basis);

    res = [ Fc, Fl, T ];
```

```
end
-------- END 'convertPolarForce.m' --------
-------- START 'split.m' --------
function [varargout] = split(a)
for k = 1:length(a)
  varargout{k} = a(k);
end
-------- END 'split.m' --------
-------- START 'wrapToPi.m' --------
function [ aout ] = wrapToPi( ain )
    aout = ain - floor(ain/(2*pi) + .5)*2*pi;
end
-------- END 'wrapToPi.m' --------
```