DEVELOPMENT OF A REAL-TIME PERFORMANCE PREDICTOR AND AN INVESTIGATION OF A RETURN TO POINT VEHICLE FOR HIGH ALTITUDE BALLOONING

By

JOSEPH P. CONNER JR.

Bachelor of Science
Oklahoma State University
Stillwater, Oklahoma
1995

Master of Science
Oklahoma State University
Stillwater, Oklahoma
2000

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2009

DEVELOPMENT OF A REAL-TIME PERFORMANCE PREDICTOR

AND AN INVESTIGATION OF A RETURN TO POINT

VEHICLE FOR HIGH ALTITUDE BALLOONING

Dissertation Approved:

Dr. A.S. Arena

Dissertation Adviser

Dr. J. Jacob

Dr. G. Young

Dr. C. DeYong

Dr. A. Gordon Emslie

Dean of the Graduate College

ACKNOWLEDGEMENTS

I would like to express my appreciation to my wife whom has supported me throughout this process with nothing but love and patience. I would like to express my thanks and appreciation to my major advisor, Dr. Andrew S. Arena, Jr., for his unending patience, constructive guidance, and his dedication and love to the field of teaching and research. I would also like to extend my thanks to my other committee members Dr. Gary Young, Dr. Jamey Jacob, and Dr. Camille DeYong whose guidance, encouragement and knowledge were an invaluable part of my education. I would like to thank all of my lab colleagues for their advice and assistance, especially Mr. S. Lee , Mr. D. Grismore, Mr. N. Moffit, and Mr. C. O'Neill,. At this time I would like to take the time to thank all my friends for their support throughout the years. Major thanks also go to my parents for their support and guidance. They allowed me to grow and learn on my own, while still being there to help support me when I needed help. Finally I would like to thank all the professors and staff members that I have worked with throughout the years and for all the opportunities that they provided for to learn both in and out of the classroom.

ABSTRACT

The Atmospheric and Space Threshold Research Oklahoma (ASTRO) program was started in 2004 to provide access to the near space environment for both educational and research opportunities. As the ASTRO project has evolved, the need for two major additions became apparent, the need for a real-time predictive tracking and control over the descent phase.

Before the start of this project, the software that was available for high altitude ballooning programs allowed real-time tracking but did not have the ability to perform real-time predictive tracking that could adjust to the performance indices of the system during the mission. In fact, most of the tracking programs available did not have any ability to predict the track of the system at all. Currently, the descent phase utilizes a circular parachute and, as such, there are no means of controlling the landing location of the vehicle and payloads. Without control, the direction the parachute takes is dependent upon the winds aloft which can allow payloads to land in undesirable locations. At times, if the risk of a long or difficult recovery is predicted the flight must be cancelled before it even begins. Without direct control of the descent phase, the chance exists that the payloads could land in areas that are not recoverable, such as heavily wooded areas, steep and rocky terrain, or hard to access private property. In order to improve control over the descent phase and the landing zone, an investigation into a Return to Point Vehicle (RPV) is being proposed. The major goals of this project are to develop software that will allow real-time predictive tracking and to investigate the feasibility of an autonomous system to return to a predestinated landing zone.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

A	Cross-Sectional Area
a	Acceleration
APRS	Automatic Position Reporting System
ASTRO	Atmospheric and Space Threshold Research Oklahoma
B	Gross Lift (Buoyancy Force)
C_d	Coefficient of Drag
C_l	Coefficient of Lift
COTS	Commercially Off The Shelve
D	Drag Force
GFS	Global Forecast System
K_P	Proportional Gain
L	Lift Force
g	Gravity
m	Mass of Balloon/Payload system
P	Pressure
Re	Reynolds Number
READY	Real-time Environmental Applications and Display sYstem
RTMP	Real-Time Mission Predictor
RPV	Return to Point Vehicle
T	Temperature
V	Velocity
~~V~~	Volume of Balloon
W	Weight

N, E, U	Coordinates in local tangent plane
X,Y,Z	Coordinates in Earth Center Earth Fixed Frame
u, v, w	Velocity components in body fixed frame
x, y, z	Coordinates in body fixed frame
λ	Longitude
ψ	Latitude
h	Altitude
S	Surface Area
α	Angle of Attack
ϕ	Rigging Angle
γ_d	Glide Angle
$\gamma_B, \gamma_W, \gamma_D$	Uncertainty in Buoyancy, Weight, and Drag
ρ	Density
φ_{bearing}	Bearing
$\varphi_{\text{heading}_{\text{GND}}}$	Ground Heading
ϵ_φ	Heading Error

INTRODUCTION

The Atmospheric and Space Threshold Research Oklahoma (ASTRO) program was started in 2004 to provide access to the near space environment for both educational and research opportunities by use of high altitude balloons. High altitude balloons have several major advantages in providing access to this near-space environment when compared to aircraft or sounding rockets. High altitude balloons routinely carry payloads to altitudes that can reach above 99.5% of the Earth atmosphere. At this altitude, payloads can be exposed to environmental conditions similar to a low orbital flight, with the exception of still experiencing 1-G. While sounding rockets can also allow access to the near-space environment, they are not able to provide the durations that are available with balloons. Altitudes achievable by typical balloons exceed common aircraft operational ceilings and are far less costly when compared to the operational cost of an aircraft.

As the ASTRO project evolved the need for two improvements arose; a method to accurately predict performance of the mission in real-time and control of the descent phase to a desired landing zone.

One of the major problems that occurred in the past was the ability to accurately predict the landing zone during a mission. Currently, the only tracking software available has little to no ability to determine the systems performance indices during a real mission. At most, the current software can adjust for the winds aloft. The objective of the new software is to have the capability of not only updating the winds aloft during the mission, but to also adjust the system parameters in real time.

In order to address the need for more control during the descent phase, an investigation into a Return to Point Vehicle (RPV) has been initiated. The current plan is to design a vehicle that has the ability to ‘learn’ the winds during the ascent phase of the mission for use during descent. Once the vehicle has reached the desired altitude, usually over 80,000 ft, it is released from the balloon. When the RPV has been released, it will compute a trajectory which returns the vehicle to a desired landing zone.

BALLOON DYNAMICS

Ascent Rate of balloon

Since one of the major objectives of this project is for the RPV to learn about the winds aloft during the ascent phase of the flight, it is necessary to understand the governing dynamics during this phase. During the ascent phase, the dynamics of the balloon governs the overall performance of the mission. Figure 1 shows the basic forces that are acting on the balloon and payload during the ascent phase. In development of the governing equations, the following assumptions have been made.

- Payload and descent craft are assumed to be a single mass
- Payload lift lines do not stretch

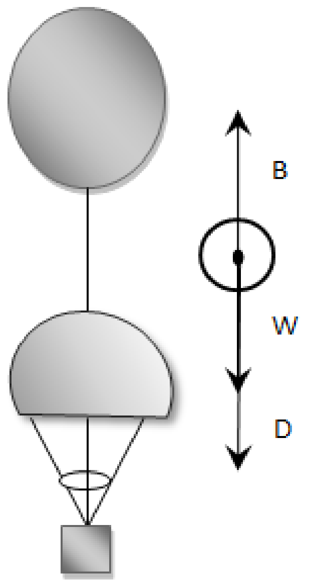


Figure 1 Free Body Diagram for Balloon and Payload

Using this information, one is able to develop the following governing equations. With the free body diagram, we can see that the ascent phase is governed by three major forces; weight of the entire system (*W*), drag force (*D*), and gross lift on the system (*B*). The governing equation for the simplified balloon/payload system is:

$$B - W - D = ma \qquad [1]$$

or:

$$\forall(\rho_{air} - \rho_{He})g - W - \frac{1}{2}\rho C_d A V^2 = m\frac{dV}{dt} \qquad [2]$$

Now that the governing equation for the balloon and payload have been developed, one needs to determine the maximum altitude that the system can traverse before the balloon has increased its volume to the point of bursting.

Balloon Pressure

The maximum volume that a balloon can increase is governed by the design and construction of the balloon. For the purpose of this paper we are going to assume that the information provided by the manufacturer of the balloon is correct. To determine the expansion of the balloon it will be assumed that it is governed by the ideal gas law. Since the volume, pressure, and temperature are known at the time of launch, one can then determine the conditions where the balloon will reach its maximum diameter as specified by the manufacturer. In order to test the assumption that the balloon can be modeled using the ideal gas law, we will need to verify that the balloon is near, or has the behavior of, a zero-pressure balloon. A zero-pressure balloon is designed to expand such that the pressure difference between the inside and outside of the balloon remains the same and thus there is a zero-pressure difference between the inner and outer walls of the balloon. An experiment with the aim of verifying the zero-pressure assumption was conducted by monitoring the pressure difference of a Kaymont KCI 30 balloon during inflation. During the experiment the fill volume of air was monitored by a gas sampling test meter which allow measurement of the volume of air to a resolution of 0.1 cubic feet. The pressure was monitored by use of a gas pressure sensor, which has a pressure range 0 to 210 kPa with a resolution of 0.05kPa. Finally the size of the balloon was recorded on camera with a 48” metal ruler in the foreground. The experimental setup is shown below in Figure 2.

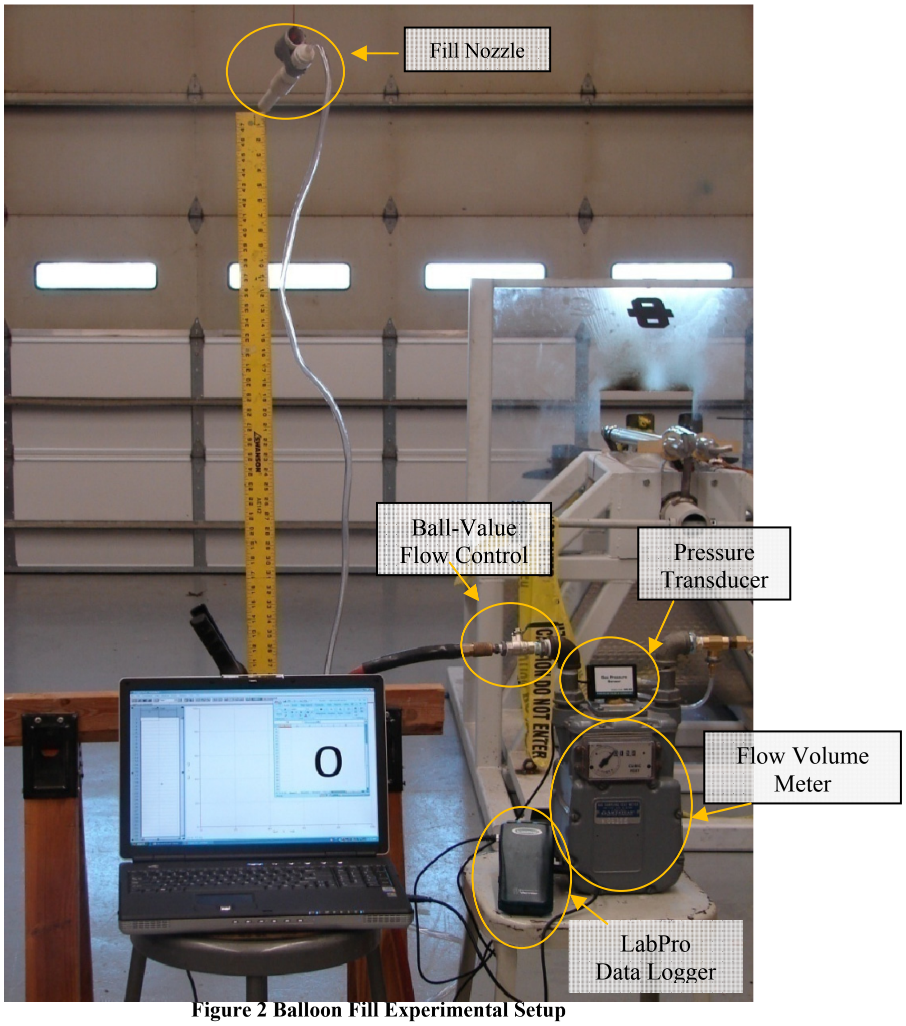


Figure 2 Balloon Fill Experimental Setup

The objective of this experiment was to inflate the balloon and then record the fill volume of air into the balloon and internal pressure at fixed fill volumes until burst. Once the balloon had reached a fix volume, the fill process was paused and the interior pressure as well as a picture of the balloon was taken. The fill then proceeded, stopping at fixed intervals until balloon burst. In order to establish the fill volume intervals, it was decided that on the first balloon that the fill volume would be monitored from empty to burst while incrementally increasing the volume of the balloon. The results of the incremental test, shown in Figure 3, indicated that while the pressure did vary as the balloon was filling, the change in pressure as a whole did not seem to vary dramatically. One item of interest was that as the balloon was starting to fill the pressure increased as would be expected but after the volume had increased above approximately 5 cubic feet the pressure actually decreased. This behavior was discussed by Fox (1) in an article written for Physics Education. Fundamentally what occurs is that as the balloon is inflated its skin becomes thinner. As the skin becomes thinner it exerts less pressure on the air inside and thus it will actually take less pressure to fill the balloon the larger the balloon's volume becomes. This is one of the main reasons why when you first start to inflate a balloon you have to blow very hard to stretch the rubber and as you continue to inflate balloon, it becomes easier.

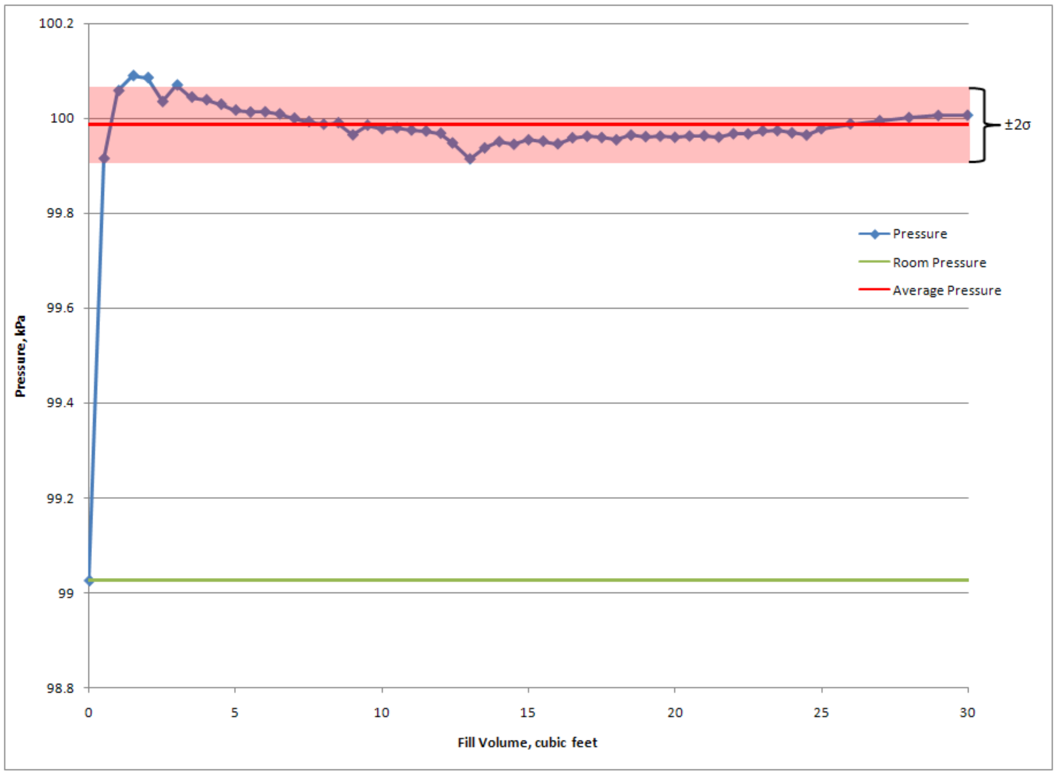
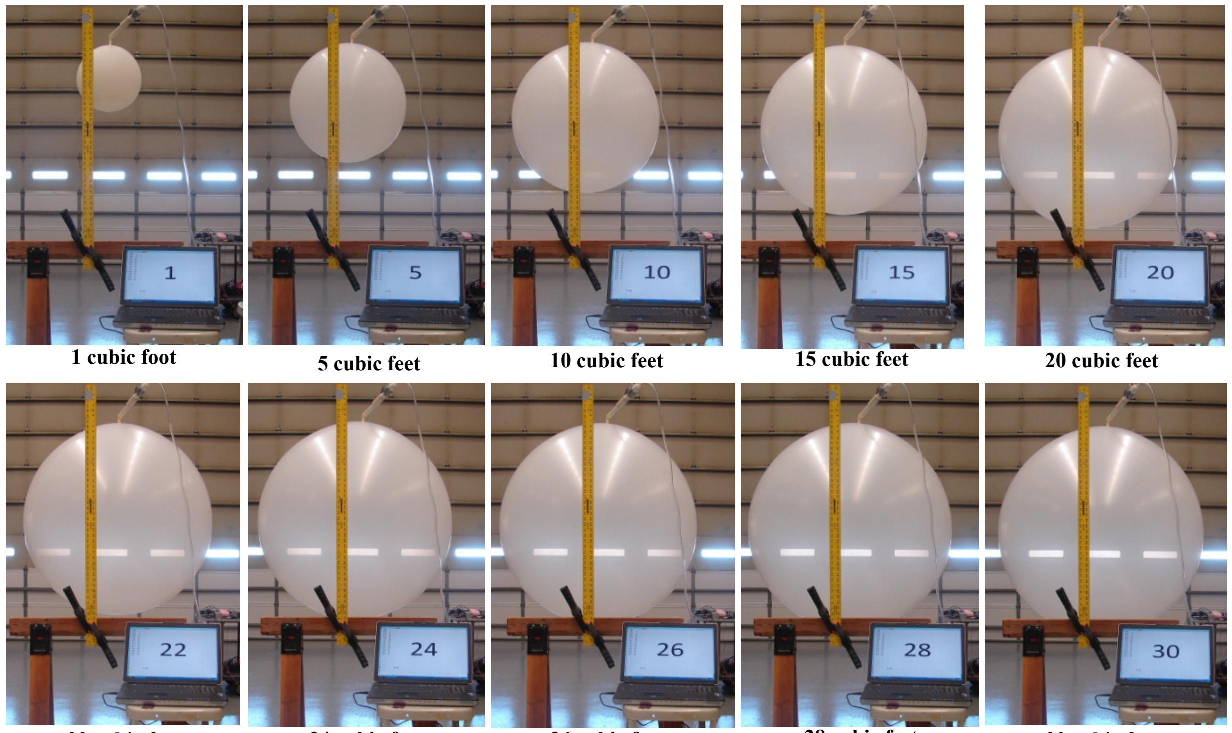


Figure 3 Extended Pressure Fill Test for a Kaymont KCI 30

An example of a typical run with the different fill volumes is shown in Figure 4. One item that should be noted is that during inflation the balloon takes on a spherical shape and then continues to maintain this shape until burst.


1 cubic foot | 5 cubic feet | 10 cubic feet | 15 cubic feet | 20 cubic feet

22 cubic feet | 24 cubic feet | 26 cubic feet | 28 cubic feet | 30 cubic feet

Figure 4 Images of Balloon During Pressure Test

After repeating this experiment for a total of five times, the data was processed and it was observed that the interior pressure of the balloon remained at or near 99.98 kPa (14.5 psi) with a standard deviation of 0.074 kPa, which results in a pressure differential between interior and external pressures of approximately 0.95 kPa (0.139 psi). The results of the five runs shown in Figure 5 illustrate the typical internal pressures seen during inflation. One problem with this figure is that the results are plotted with respect to absolute pressure, and as can be seen in the zero volume state, there was a difference in atmospheric pressure over the several days the experiments was run. To alleviate this problem it was decided to plot the differential pressure between the interior and exterior of the balloon. The resulting differential pressure, shown in Figure 6, indicates that the pressure difference of the balloon remained at or near 0.938 kPa (0.136psi), with a standard deviation of 0.0485kPa. This small difference supports the assumption that the balloon can be approximated as a zero-pressure balloon. Since the balloon can now safely be modeled as a zero-pressure balloon, it's expansion rate should be governed by the Ideal gas law.

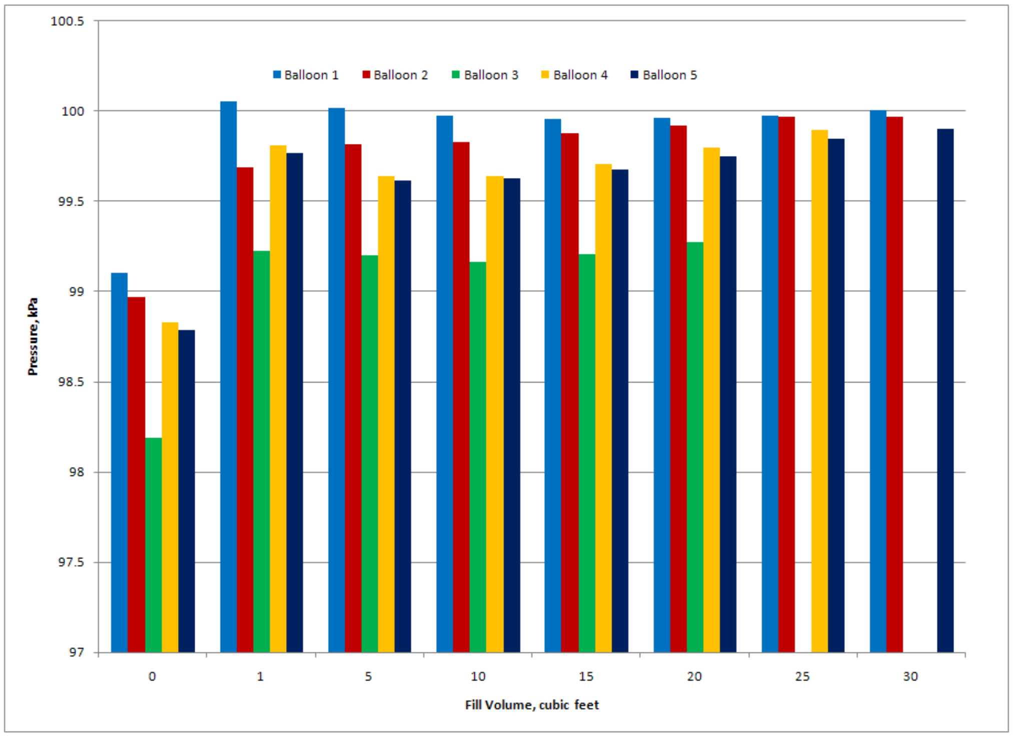


Figure 5 Pressure Fill Test for Several Kaymont KCI 30 Balloons

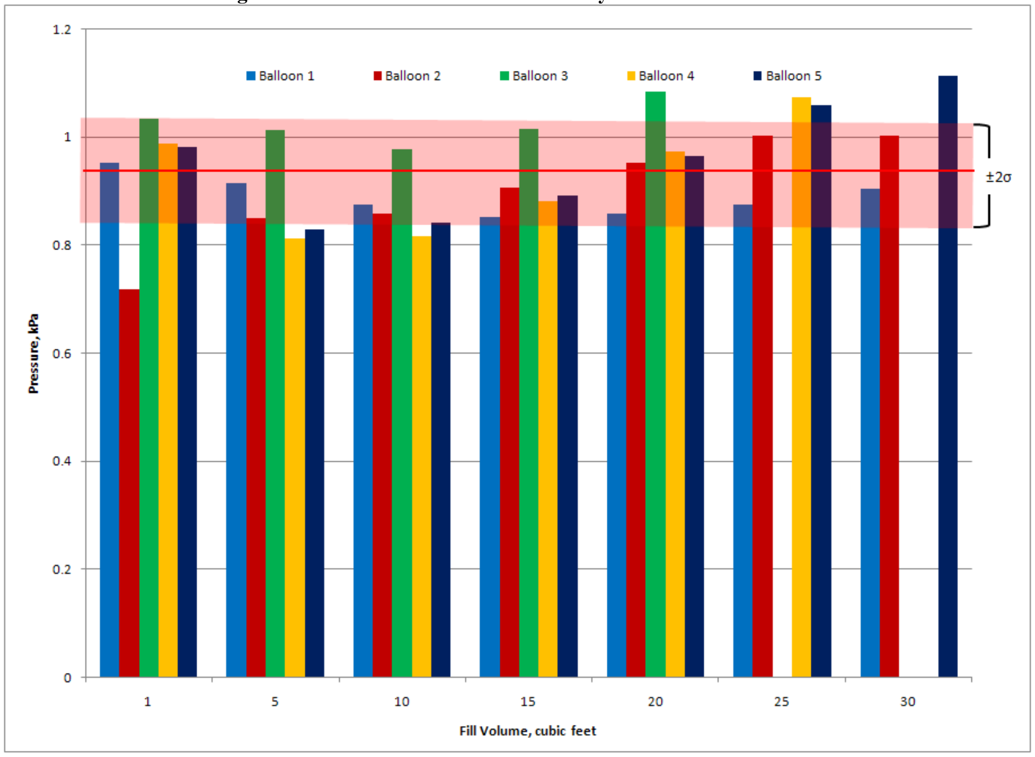


Figure 6 Differential of Interior and Exterior Pressure Test

Balloon Volume Expansion

Occurring to the Ideal gas law, the volume of a gas is a function of its pressure, and temperature according to the equation:

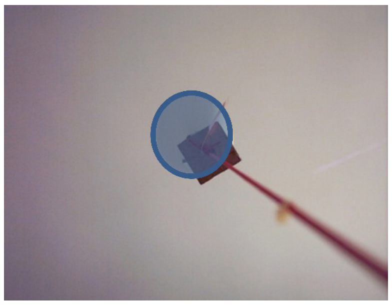
$$P\forall = nRT \qquad [3]$$

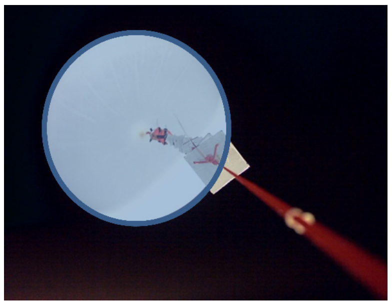
Once the balloon is filled with a known volume of helium, the volume of the balloon at altitude can be found with:

$$\forall_{alt} = \forall_{gnd}\left(\frac{P_{gnd}}{P_{alt}}\right)\left(\frac{T_{alt}}{T_{gnd}}\right) \qquad [4]$$

Using equation [4], along with the standard atmospheric model (2), a graph of balloon expansion as a function of altitude was determined. In order to obtain the results, it was assumed that the balloon was filled with approximately 550 cubic feet of helium at launch and that the helium expands at a rate governed by the ideal gas law.

To determine how well this method works, information obtained from a camera on ASTRO-04 was used to determine the size of the balloon during the flight. To accomplish this, a payload was designed to hold a digital camera which was focused toward the balloon. Using the resulting pictures shown in Figure 7, a comparison between results obtained with the equation developed above and the pictures, were created to indicate how the balloon's diameter increased as a function of altitude. The results, shown in Figure 8, indicate that the model of balloon expansion, developed above, matches well with data obtained from the ASTRO-04 flight.


~1000 ft
Diameter from
Picture 10 ft
Ideal Gas Law 10.12 ft


~60,000 ft
Diameter from
Picture 23 ft
Ideal Gas Law 23.51 ft

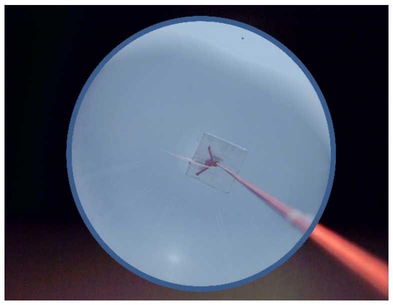

~90,000 ft
Diameter from
Picture 33 ft
Ideal Gas Law 35.64 ft

Figure 7 Expansion of Balloon during ASTRO-04

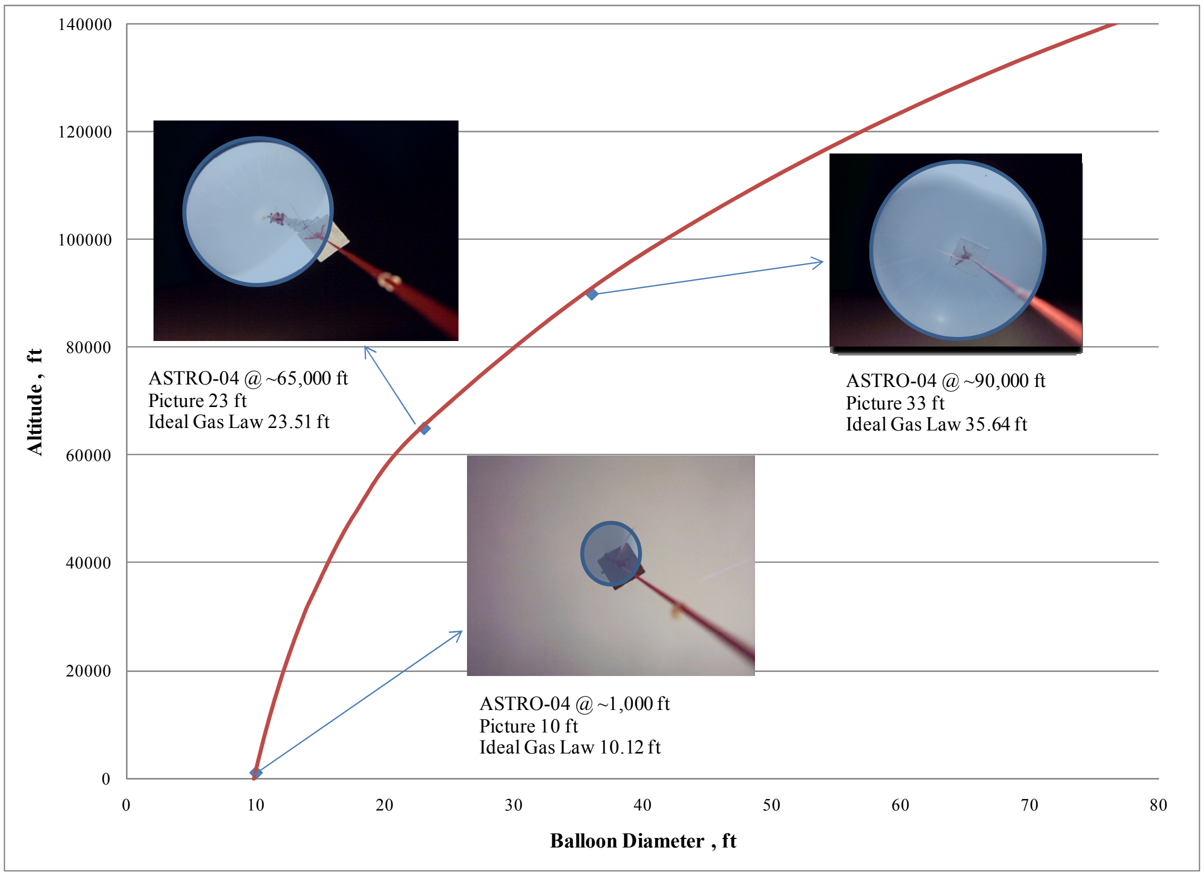


Figure 8 Balloon Expansion as a Function of Altitude

While it may appear that the balloon will continue to ascend, eventually the balloon will do one of two things; either the balloon will stop expanding, or it will reach its elastic limit. In the first case, the latex of the balloon will only expand as long as the pressure exerted by the helium is greater than the elastic forces of the latex. As pressure decreases, the elastic forces will equalize with the pressure forces and the balloon will not be able to expand any more. Once the balloon is no longer able to expand, it will be displacing a fixed volume of air which will not change as the balloon continues to rise. Since the air above it is still continuing to drop in air pressure, eventually the static volume of the balloon will lead to decreasing gross lift and the balloon will begin to descend. Ultimately, the balloon will stabilize at an altitude where the gross lift and weight are equal. Once the balloon reaches this static case, it becomes what is known as a 'floater'. The balloon will remain floating near this altitude until either the latex deteriorates due to solar effects, or the helium leaks out.

For the second case, the balloon will continue to expand until the latex is stretched beyond its elastic range and bursts. For most balloon systems, this is the way the system transfers from the ascent phase to the descent phase. Since this phase involves the balloon physically bursting, some ballooning programs will perform a commanded release of the balloon to avoid the forces that are encountered during burst. One of the other reasons that some will perform this commanded release is to avoid the descent vehicle from becoming fouled with the remains of the balloon. The results of a severe fouling can be seen in Figure 9 and give an indication to the amount of damage that can be caused by commanding a release after burst. This entanglement resulted in two main problems. The first is that the remains of the balloon and cabling became entangled in the chords of the parachute resulting in the chute not fully opening and thus leading to a faster descent velocity. The second problem was that most all the antennas on board became fouled and ultimately were either damaged or broken during the descent phase. Fortunately due to multiple layers of tracking, the system was still able to be tracked and located.

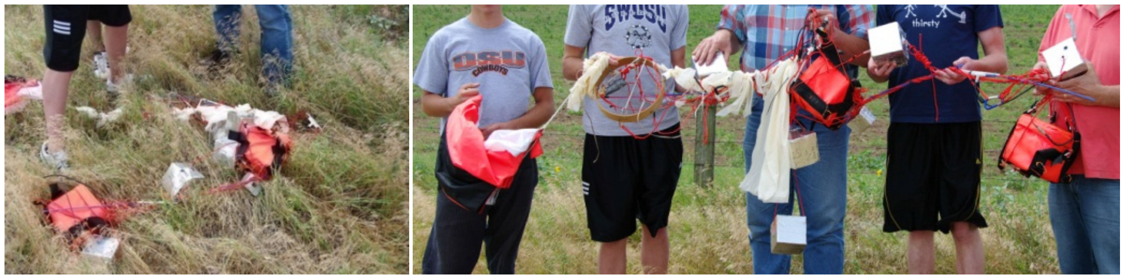


Figure 9 Balloon Fouling Damage (ASTRO-14)

Gross Lift

In ballooning, there are three different categories of lift that act on the balloon; gross lift, nozzle lift, and free lift. Gross lift is total lifting force acting on the balloon by the helium. Nozzle lift is gross lift minus the weight of the balloon and would be the force measured at the nozzle of the balloon before the addition of payload. The final type of lift that is acting on the balloon is free lift. Free lift is the useful lift acting on the balloon after the addition of the payload weight. It is this free lift that determines the ascent rate of the system. The three different types of lift are shown in Figure 10.

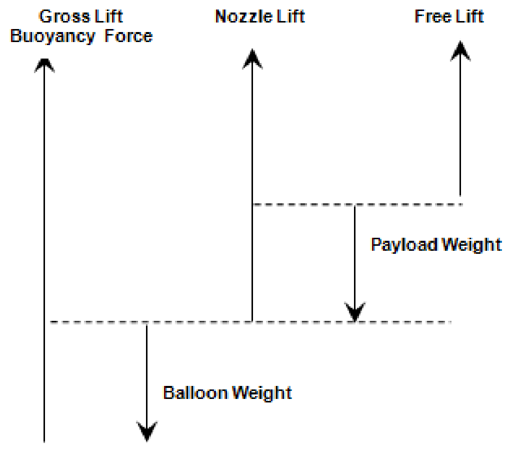


Figure 10 Lift Categories in Ballooning

Gross lift is determined by the relationship between the amount of helium and air that is displaced. The following equation is used to determine gross lift:

$$B = \cancel{V}(\rho_{air} - \rho_{He})g \qquad [5]$$

However, since the density of both air and helium change as a function of altitude, the gross lift must be solved for as the balloon's altitude changes. In order to determine the change in density of air, the standard atmospheric model for the Earth was used to determine pressure and temperature of air. Using the pressure and temperature found from the atmospheric model, the density of air and helium was found with use of the Ideal gas law. Using the Ideal gas law the density of air and helium can be found with:

$$\rho_{alt} = \frac{P_{alt}}{P_{gnd}}\frac{T_{gnd}}{T_{alt}}\rho_{gnd} \qquad [6]$$

This result can then be substituted back into equation [5] to obtain:

$$B_{alt} = \cancel{V}_{alt}\frac{P_{alt}}{P_{gnd}}\frac{T_{gnd}}{T_{alt}}\left(\rho_{air_{gnd}} - \rho_{He_{gnd}}\right)g \qquad [7]$$

Using the results from equation [4] and substituting them into the equation above, one obtains:

$$\begin{aligned} B_{alt} &= \cancel{V}_{gnd}\left(\frac{P_{gnd}}{P_{alt}}\right)\left(\frac{T_{alt}}{T_{gnd}}\right)\left(\frac{P_{alt}}{P_{gnd}}\right)\left(\frac{T_{gnd}}{T_{alt}}\right)\left(\rho_{air_{gnd}} - \rho_{He_{gnd}}\right)g \\ &= \cancel{V}_{gnd}\left(\rho_{air_{gnd}} - \rho_{He_{gnd}}\right)g \end{aligned} \qquad [8]$$

From this result, it can be seen that the gross lift acting on a balloon is not a function of altitude, but instead is only a function of the initial volume of helium.

Ascent Drag Model

Now that there is an understanding on how the balloon’s diameter changes with respect to time, one can now establish how the drag changes as a function of altitude. The total drag on the system can be determined with:

$$\boldsymbol{D = \frac{1}{2}\rho_{air}C_d A V^2} \qquad [9]$$

In order to determine the drag coefficient acting on the balloon, it was assumed that at any given time a quasi-steady solution to equation [2] would be satisfied.

$$\boldsymbol{B - W - \frac{1}{2}\rho C_d A V^2 = 0} \qquad [10]$$

In order to verify that a quasi-steady solution is valid, the flight results for several flights were used to determine the acceleration during the ascent phase of the mission. The results obtained from nine missions indicate that while there is some acceleration of the system during the ascent phase the amount seen is small as can be seen in Figure 11, where the maximum magnitude of acceleration did not exceed 0.008 m/s^2. With the acceleration being so low, the system can be safely assumed to be a quasi-steady solution.

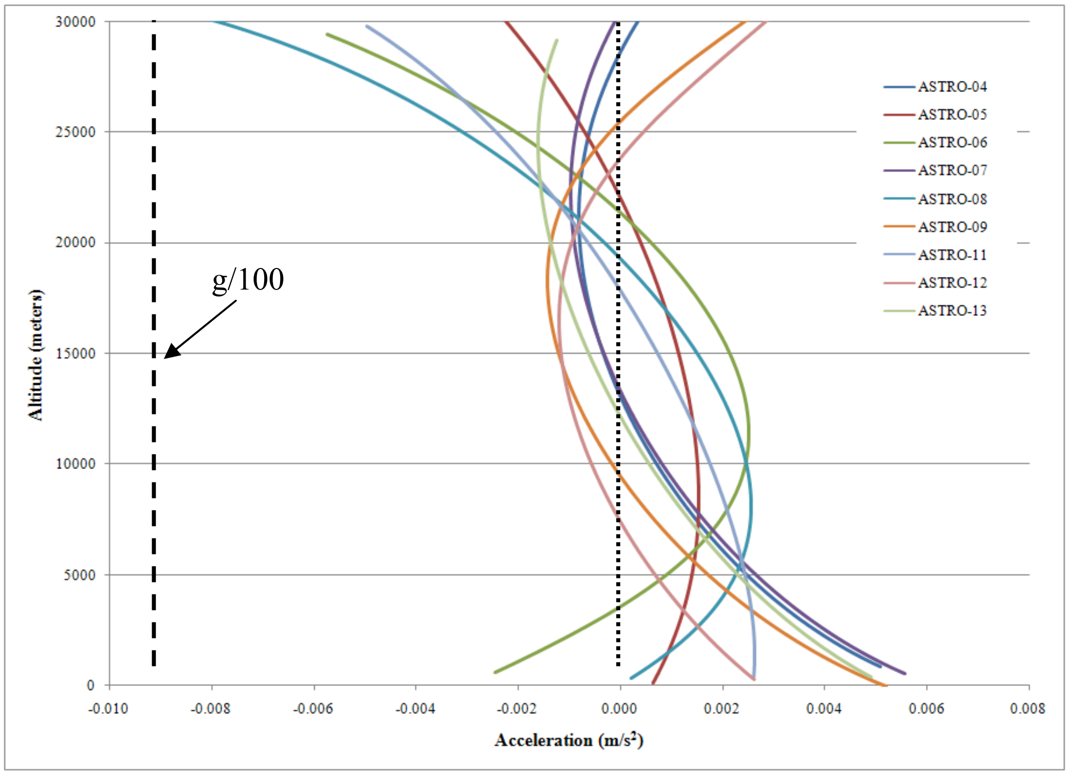
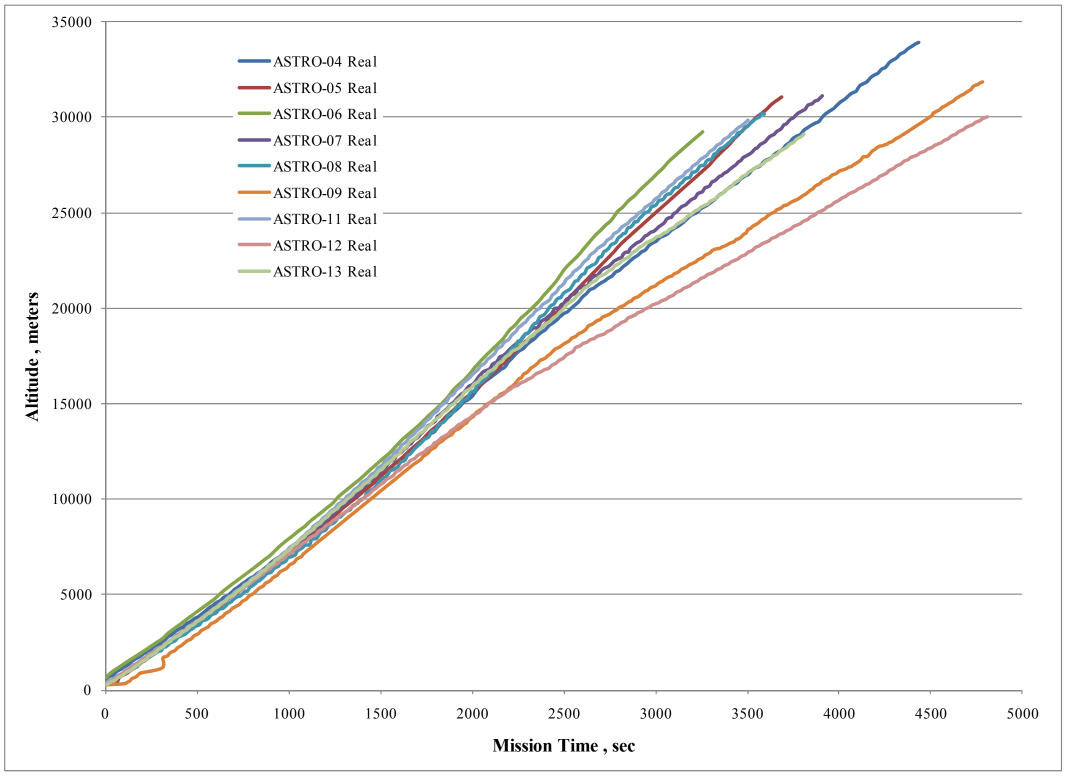
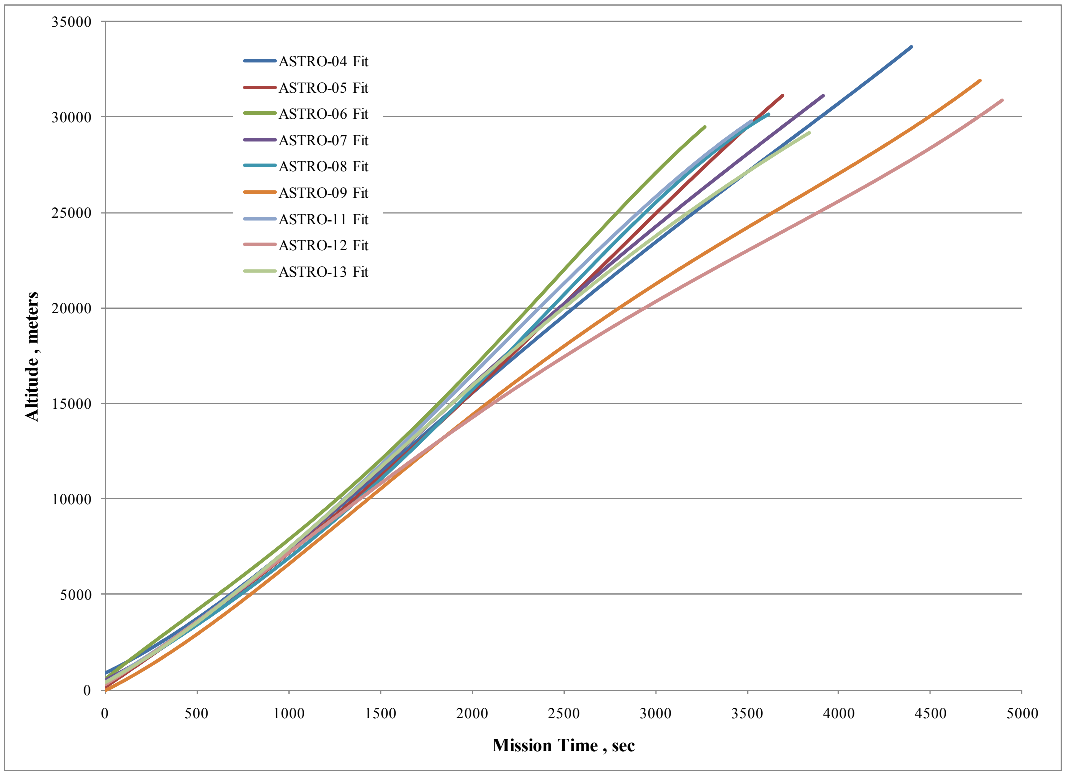


Figure 11 Acceleration as a Function of Altitude

The acceleration profiles used above where not directly obtained from flight data, the information was acquired from fitting a 4th order polynomial line through mission data of altitude as a function of mission time, as seen in Figure 12.



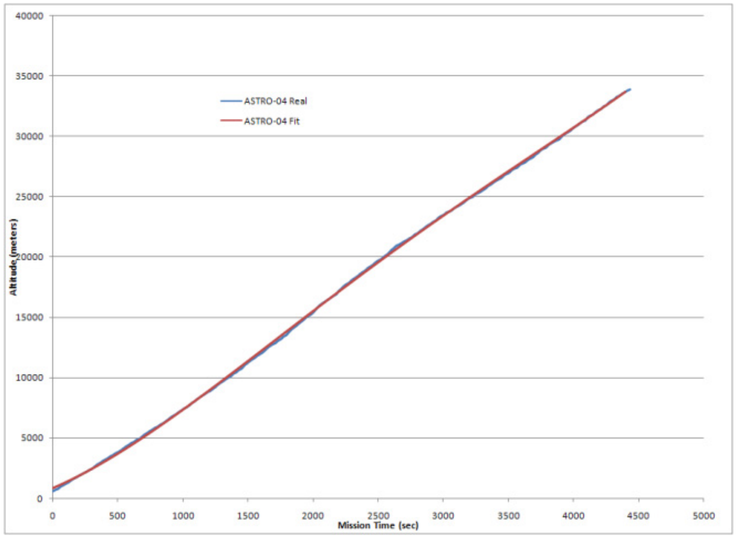
Raw Mission Data



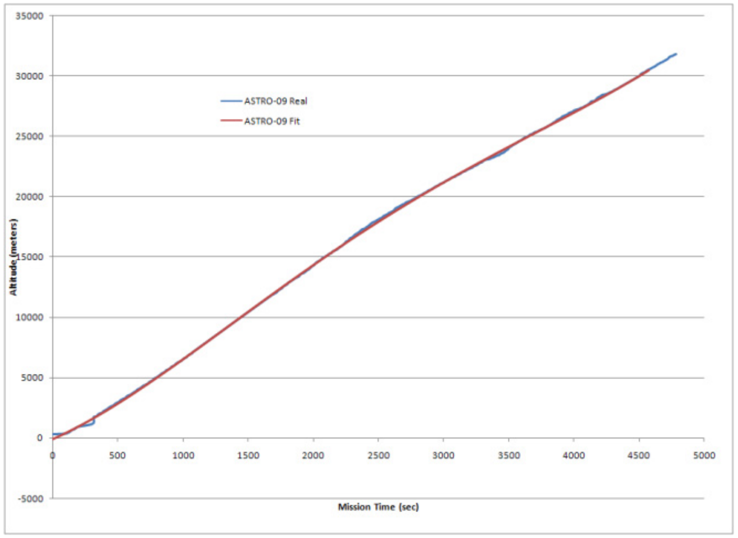
Fitted Mission Data
Figure 12 Altitude as a Function of Mission Time

In order to obtain the altitude information, a packet radio along with a GPS receiver onboard the balloon was used. This radio transmitted the location of the balloon on a 13 second interval. The signal was decoded and stored for use in real-time tracking as well as post flight analysis. Once this data was obtained a 4th order polynomial was fit to the data and then plotted against the real data.

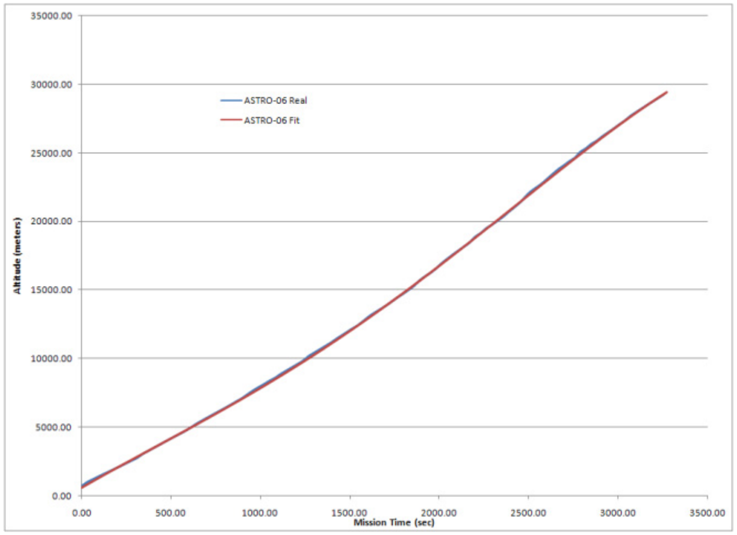
The flight data was processed in this manner for two reasons. First was that the original data set did not contain evenly spaced data sets, due to loss of packets during the mission, and thus would be very difficult to obtain velocity information from as most central difference methods require an evenly spaced data. The second reason this method was selected is that since the results would be a polynomial equation it could be readily differentiated to obtain velocity and acceleration that occurred during the flight. Examples of this curve fitting are shown in Figure 13 and show the results of the curve fit compared to real flight data for several flights.



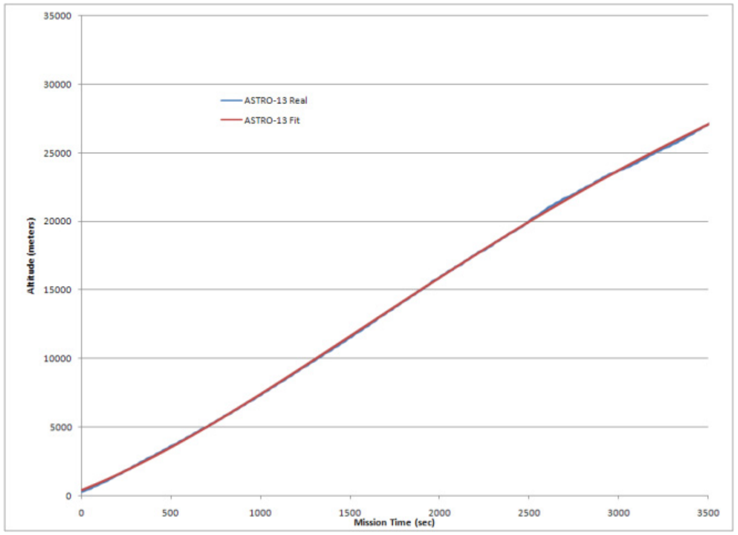
ASTRO-04



ASTRO-09



ASTRO-06



ASTRO-13

Figure 13 Altitude Real Data and Fit Data as a Function of Mission Time

The polynomial equations obtained from the curve fitting, where then differentiated to obtain velocity as a function of altitude and then differentiated once more to obtain the acceleration graphs discussed above. With the capability of determining velocity and balloon diameter during the ascent phase, we can focus on determining the drag coefficient as a function of either altitude or Reynolds number. If we solve equation [10] for drag

$$C_d = 2\frac{B - W}{\rho A V^2} \qquad [11]$$

Using equation [11], along with flight data from nine different ASTRO flights drag coefficient as a function of Reynolds number, was found. The drag coefficient results of the flights were then fit with a 4th order polynomial to obtain a model of the drag coefficient over the flight regimes that had been encountered; this result is then used during the ascent phase to aid in the prediction of flight performance during the ascent phase. By using the data from nine flights, the balloon drag model fit with a 4th order polynomial with the following form. The developed model, along with the experimental drag coefficient from the flights, is shown in Figure 14.

$$\begin{gathered} C_d = \sum_{0}^{4} \mathbf{c(i)} * \mathbf{Re^i} \\ \textbf{where:} \\ \begin{aligned} \mathbf{c(0)} &= 7.119\mathrm{E}-01 \\ \mathbf{c(1)} &= -2.568\mathrm{E}-06 \\ \mathbf{c(2)} &= 4.707\mathrm{E}-12 \\ \mathbf{c(3)} &= -4.040\mathrm{E}-18 \\ \mathbf{c(4)} &= 1.309\mathrm{E}-24 \end{aligned} \end{gathered} \qquad [12]$$

The resulting balloon drag model is shown in Figure 15, along with the classic spherical drag, presented by Hoerner (3). The results show that while the balloons drag coefficient does exhibit the behavior of transition from low to high as Reynolds number decreases it does so with a smoother transition. Where the classic drag on a sphere has a major transition from approximately 0.10 to 0.4 over a Reynolds number range of 250,000 to 400,000 the balloon transition over this same drag coefficient range occurs in a Reynolds number range of 200,000 to 1,200,000.

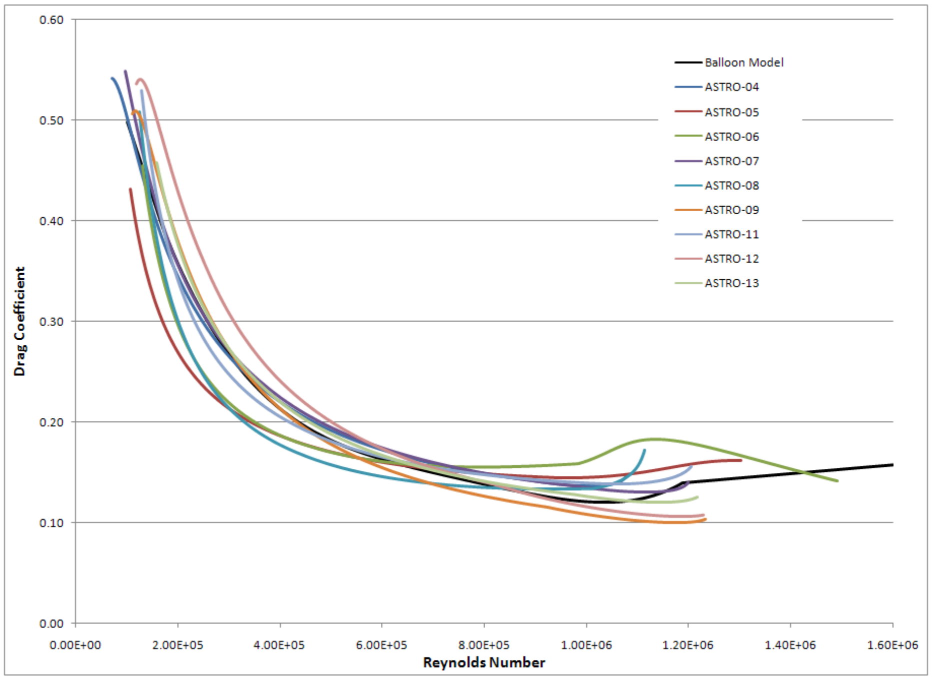


Figure 14 Balloon Drag Model Compared to Flight Results

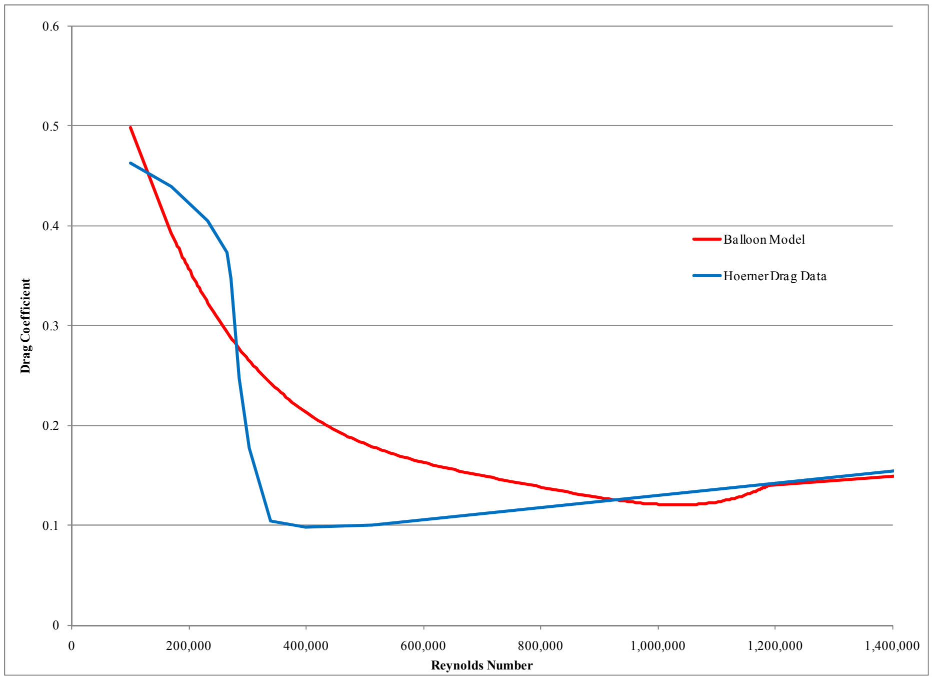


Figure 15 Classic Spherical Drag Compared to Balloon Drag

Once the drag coefficient was found, a full numerical simulation of the ascent phase could be performed using the governing equation [2]. In order to determine how well this numerical simulation works, actual results from several ASTRO flights where used. It was decided to plot the results of ASTRO-06 and ASTRO-09 against numerical simulations of both flights. The reasoning behind selected these two flights is that with information collected to date ASTRO-06 represents one of the fastest ascent rates and ASTRO-09 one of the slowest.

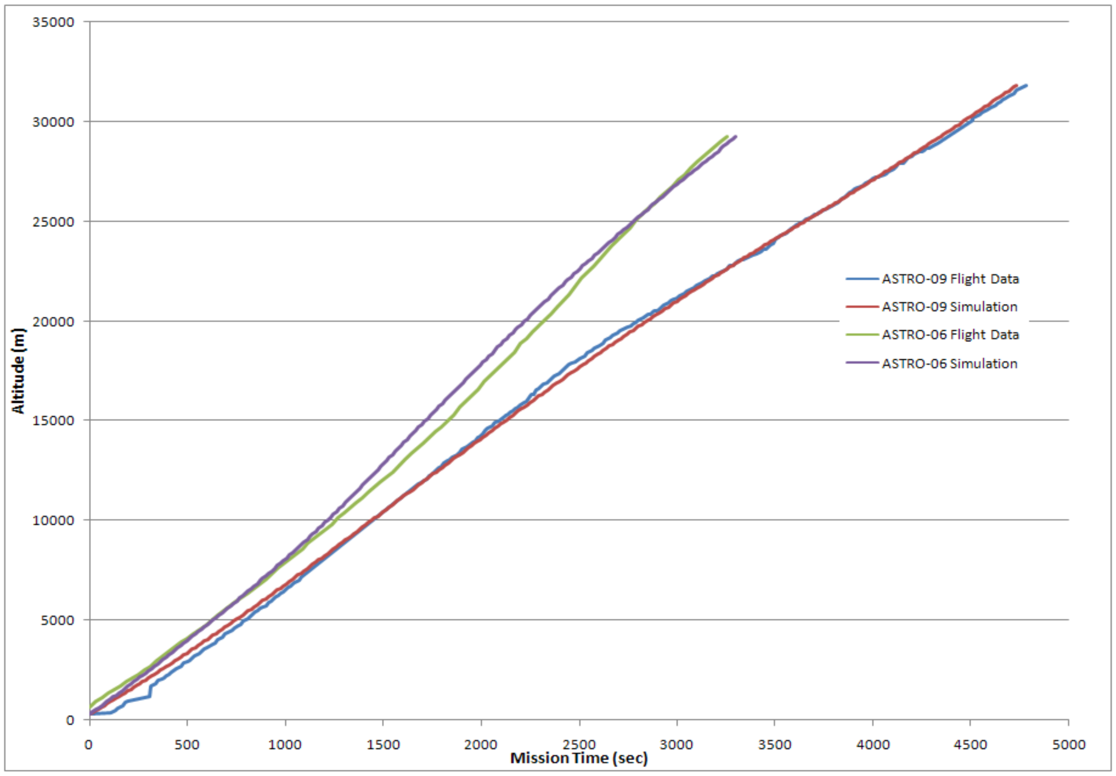


Figure 16 Numerical Simulation of Ascent Phase compared to Actual Flight Data

As can be seen in Figure 16, the results of the simulation and the flight data agree. One interesting note is that while the slower flight, ASTRO-09 appears to be a constant ascent rate this is actually not the case. In fact, what happens is near 20,000 meters, the ascent rate actually slows down. This decrease in flight speed has been noted by other ballooning programs but few have taken the time to fully understand why this happens. If we take a look at drag coefficient as a function of altitude, we note that as the balloon is approaching 20,000 ft, the drag coefficient increases rapidly. This can be seen in Figure 17.

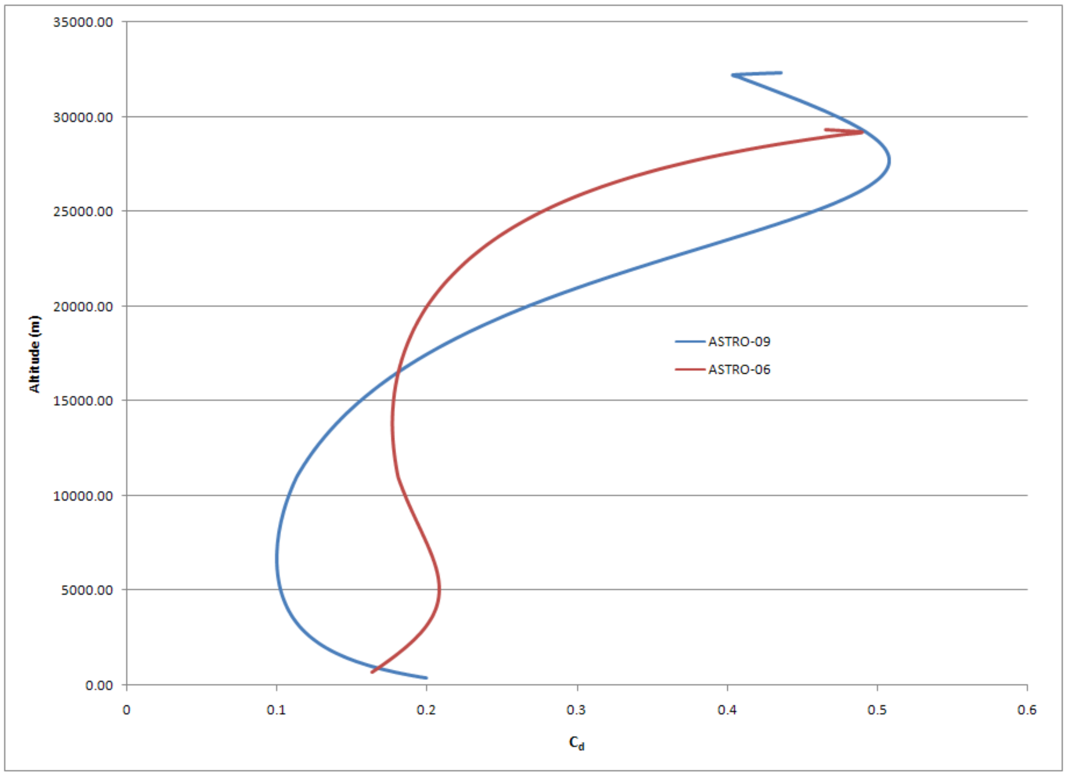


Figure 17 Drag Coefficient Versus Altitude

Ascent Rate Performance

With an understanding of the major parameters of the balloon's dynamic equation now able to be determined, we can take a look at the performance of the balloon with varying payload weights. Figure 18 shows the results of increasing the weight of the payload on the maximum altitude that can be obtained before bursting. It should be noted that during this study, the balloon (KCI 3000), was held constant and only the payload weight and fill volume of helium was changed. This result, shown in Figure 18, would be expected as an increase in helium would be required in order to increase the overall gross lift. As was discussed earlier, the amount of helium in the balloon is directly related to the initial volume of the balloon. An interesting detail from this result is that while an increase in weight does decrease the maximum altitude, this decrease is not a one-to-one relationship. For example, at 1000 ft/min, a 10 lb payload can reach a predicted altitude of 108k feet, while a 20 lbs payload is predicted to reach 99.2k feet. At first this might seem erroneous, but one has to recall that the gross lift on the balloon is determined by the volume of helium used and that it is this initial volume that determines how large the balloon is as a function of altitude.

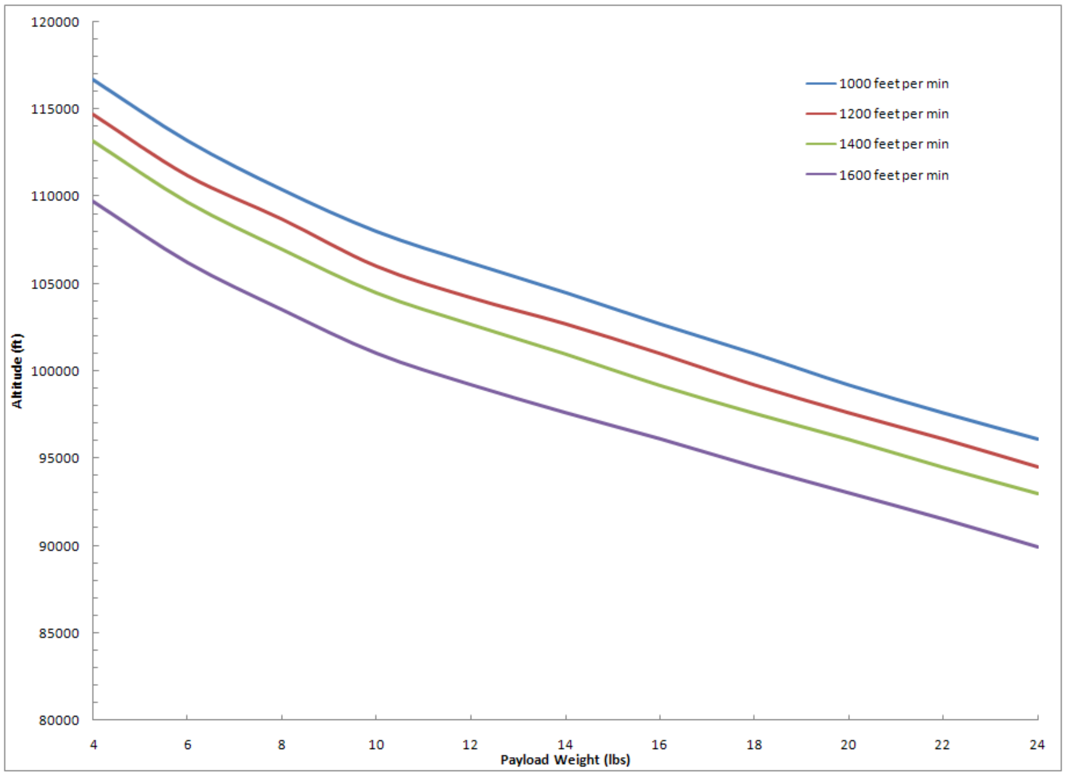


Figure 18 Payload Weight Affects on Ascent Rate

Finally, in order to get an understanding of how the amount of Helium affects the performance of the system, it was decided to look at the performance of the balloon with varying initial fill volumes. In order to conduct this, it was decided to keep the type of balloon, as well as the payload weight constant, and only vary buoyancy.

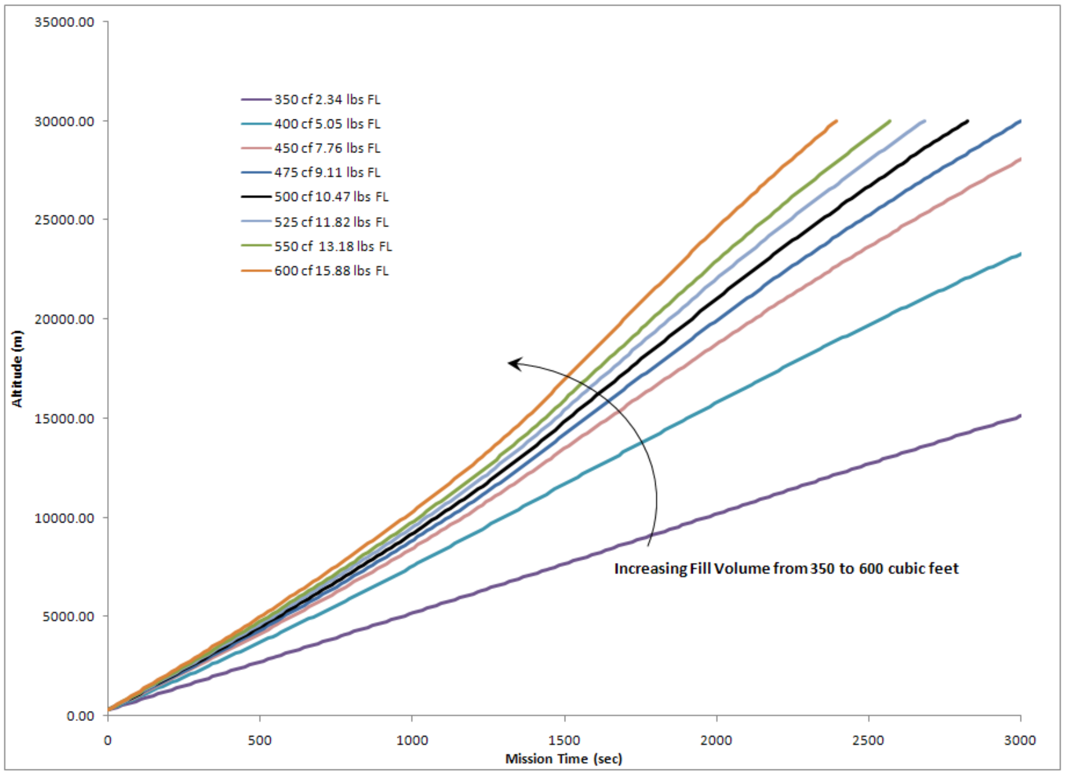


Figure 19 Fill Volume Affects on Ascent Rate

From the results, it was seen that the relationship between fill volume and ascent rate is by no means linear. As expected, as the fill volume increases the ascent rate also increases. What is somewhat surprising is that the higher volume balloons actually increase ascent rate as the altitude increases. One of the other items of interest is how rapidly this effect takes place. For example, the 400 cubic foot case has an ascent rate that remains nearly constant, the 450 cubic foot case has an increase in ascent rate as altitude increases and the 350 cubic foot case has a slight decrease in ascent rate, with an increase in altitude.

Descent Phase

Once the balloon either bursts or a cut-down command is issued, the system transitions to the descent phase. During the descent phase, the system can take on one of the following modes; free-fall, spherical chute, or parafoil mode. Since the free-fall and spherical chute modes are basically the same they will be the first to be developed and analyzed.

Spherical Parachute

During the descent phase, the major objective is to slow the descent of the system to an acceptable impact velocity. In development of the equations, the following assumptions have been made.

- Payload lift lines do not stretch
- System does not rotate about its centroid
- Parachute and payloads produce no lift

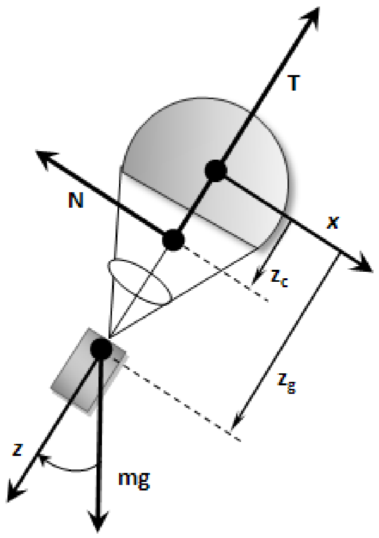


Figure 20 Free Body Diagram of Parachute and Payload

With the free body diagram, we can see that the descent phase is governed by several major forces; weight of the entire system (*mg*), the normal aerodynamic force (*N*), and a tangential aerodynamic force on the system (*T*). Resolving the forces along the body axis one obtains:

$$mg * \cos(\theta) - T = ma_z \quad [13]$$

$$mg * \sin(\theta) - N = ma_x \quad [14]$$

$$M_G = -N(z_g - z_c) = I\alpha \quad [15]$$

For this case we are going to assume that the system does not rotate about its centroid, therefore equation [15] is set to zero.

$$M_G = -N(z_g - z_c) = 0 \quad [16]$$

Since $(z_g - z_c)$ is non-zero then the only way for equation [16] to be satisfied is for the normal aerodynamic force (N) to be zero, and equations [13]and [14] can be rewritten as:

$$mg * \cos(\theta) - T = ma_z \quad [17]$$

$$mg * \sin(\theta) = ma_x \quad [18]$$

With this set of equations, we can now see that the only forces acting on the parachute/payload is the weight and tangential aerodynamic force. Under typical conditions, the tangential aerodynamic force is composed of the drag on the parachute (D_s) and payload (D_p).

$$T = (D_P + D_s) = \frac{1}{2}\rho V^2 \left(C_{D_s}S_s + C_{D_p}S_p\right) \quad [19]$$

Using this result equation [17] becomes:

$$mg * \cos(\theta) - \frac{1}{2}\rho V^2 \left(C_{D_s}S_s + C_{D_p}S_p\right) = ma_z \quad [20]$$

Now with the system of equations found, the only remaining unknowns are the coefficients of drag on the parachute and the payloads. The coefficient of drag on a parachute with low porosity has been reported as 1.2 for Reynolds numbers greater than 100,000 and the typical parachute used in ASTRO missions has an effective cross-section area of approximately 1.5 m^2 or a total drag area ($C_{D_s}S_s$) of 1.8. The payloads that are typically flown on the flights are 0.1 m (4 inch) cubes; the cube normally has a coefficient of drag of 1.07. On most flights, the total number of payloads flown is twelve student payloads, total area of 0.12 m^2, and two tracking payloads, 0.05 m^2 of area, that total to 0.18 m^2 of frontal area for payloads. The frontal area of the payloads along with the coefficient of drag makes for a total payload drag ($C_{D_p}S_p$) of 0.19. Since the total payload drag is significantly less than that of the parachute equation [20] can be simplified to:

$$mg * \cos(\theta) - \frac{1}{2}\rho V^2(C_{D_s}S_s) = ma_z \qquad [21]$$

Using this equation along with information from ASTRO flights 4 through 7, several numerical simulations were conducted and are shown in Figure 21. The results indicate that an effective chute diameter between 1.27 and 1.4 meters (50 and 55 inches) appears to match the results obtained from ASTRO flight 4, 6, and 7 extremely well. The ASTRO-05 flight, however, does not exhibit the same behavior as the other flights. Instead of traveling along a line of constant effective chute diameter as the other flights, ASTRO-05 appears to actually change its behavior in mid-flight. When looking at the data for the ASTRO-05 flight, it appears that this change in behavior occurs around 20,000 meters. In order to simulate this effect, the numerical simulator was edited to allow a change in effective chute diameter at a given altitude. Using the simulator it has been proposed the following happened during the ASTRO-05 flight. After burst, the system began to descend as normal and the chute opened to an effective diameter of 1.27 meters (50 inches). As the system approached an altitude of 20,000 meters, the descent profile suddenly changed to an effective diameter of 0.762 meters

(30 inches). Support for this assumption was made during recovery, as it appears that at some time during the descent the balloon became entangled in the parachute rigging.

Normally one would expect the parachute to reach a terminal velocity after falling for a short while, but this does not appear to be the case in either the real flight data or the numerical simulation. In both cases, the system does decelerate rapidly. At first as it appears to decelerate from 65 m/s to 20 m/s in 420 seconds and then for the remaining 1245 seconds it slows from 20 m/s to nearly 8 m/s. This reaction can be seen in both the simulated flight data as well as real flight data from ASTRO-04 to ASTRO-07, as shown in Figure 21 below.

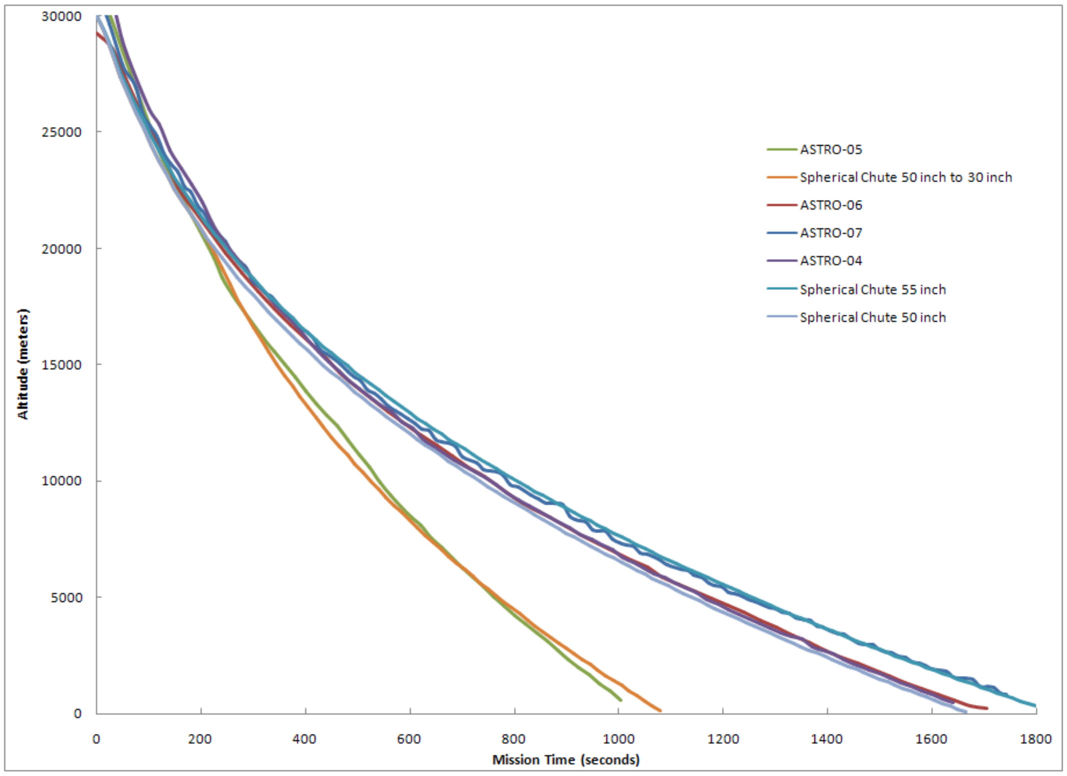


Figure 21 Numerical Simulation of Descent Phase compared to Actual Flight Data

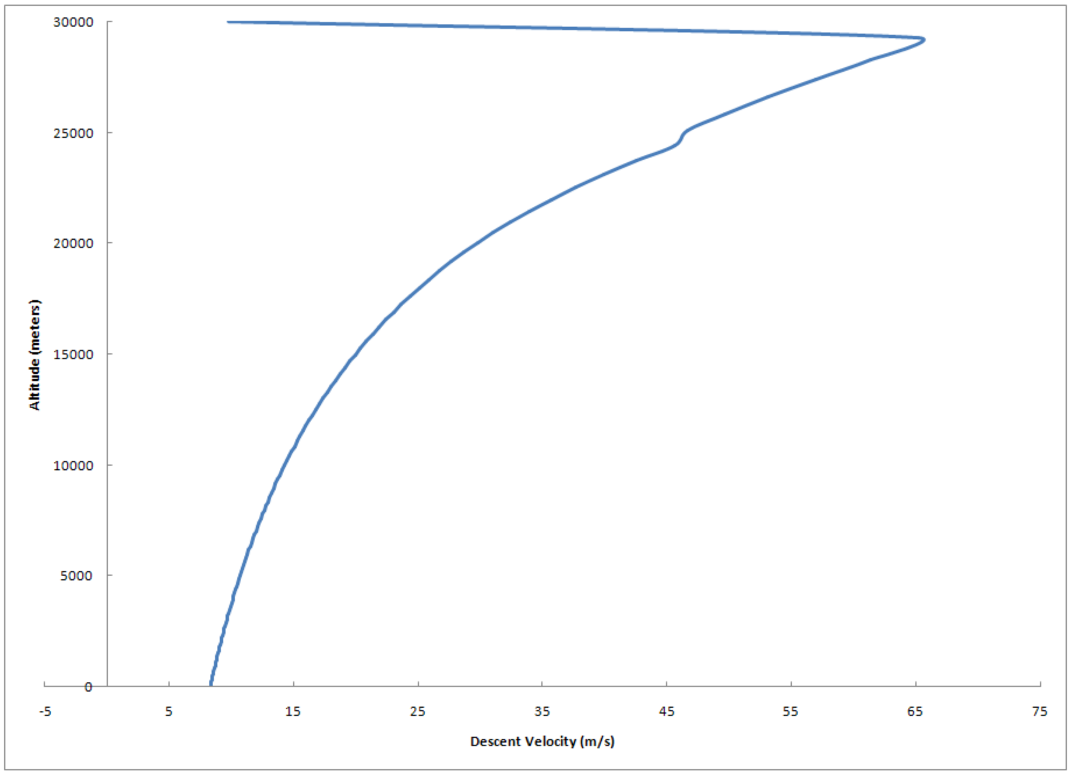


Figure 22 Descent Velocity as a Function of Altitude

Parafoil Dynamics

One of the major disadvantages of using a spherical chute during the descent phase is that this type of chute cannot produce a driving force, which means no significant quantities of lift are produced. Without the ability to generate a driving force in the horizontal plane, the system is controlled only by the winds aloft. However if a parafoil, gliding parachute, is used then a driving force can be generated. Since a parafoil has the ability to generate a driving force in the horizontal plane, then it also has the ability to control its direction with respect to the wind. This control in the horizontal direction is what is of major interest to this project. In order to understand how much control there can be, a basic kinematic analysis of a generic parafoil will be developed and then analyzed. In development of the equations, the following assumptions have been made.

- Payload and rigging lines do not stretch
- System does not rotate about its centroid
- Payloads produce negligible amounts of lift

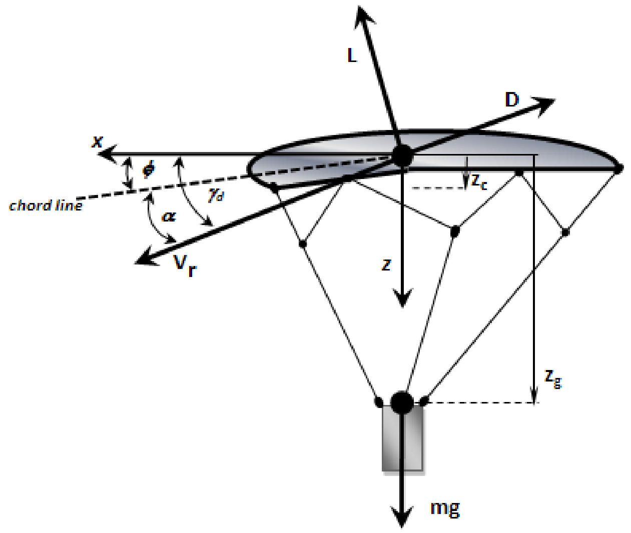


Figure 23 Free Body Diagram of Parafoil and Payload System

With the free body diagram we can see that the dynamics of the parafoil are governed by several major forces; weight of the entire system, (*mg*), aerodynamic forces lift (*L*), and drag (D). Resolving the forces along the body axis one obtains:

$$mg - L\cos(\gamma_d) - D\sin(\gamma_d) = ma_z \tag{22}$$

$$L\sin(\gamma_d) - D\cos(\gamma_d) = ma_x \tag{23}$$

$$M_G = -(L\sin(\gamma_d) - D\cos(\gamma_d))(z_g - z_c) = I\alpha \tag{24}$$

If it is now further assumed that the parafoil will reach a state of equilibrium for a given altitude, then the equations above will simplify to the following:

$$mg - L\cos(\gamma_d) - D\sin(\gamma_d) = 0 \tag{25}$$

$$L\sin(\gamma_d) - D\cos(\gamma_d) = 0 \tag{26}$$

$$M_G = -(L\sin(\gamma_d) - D\cos(\gamma_d))(z_g - z_c) = 0 \tag{27}$$

Using this set of equations, the angle of descent, or glide angle (γ_d), can be found. The glide angle can be determined by using equation [26] and solving for (γ_d).

$$\tan(\gamma_d) = \frac{D}{L} = \frac{1}{L/D} \qquad [28]$$

From this result it can be seen that the glide angle is a function of the lift to drag ratio, as this ratio, increases the glide angle becomes shallower. The glide angle has a direct effect on two parameters of importance when it comes to controlling the landing location of the parafoil and payload; total time to descend and range traveled before reaching landing zone. Before we can determine the glide angle, we have to be able to determine the lift and drag that are acting on the parafoil. To determine the lift and drag, the following equations are used.

$$\mathrm{L} = \frac{1}{2}\rho \mathrm{V_r}^2 \mathrm{S C_L} \qquad [29]$$

$$\mathrm{D} = \frac{1}{2}\rho \mathrm{V_r}^2 \mathrm{S C_D} \qquad [30]$$

As can be seen from the equations above, at a fixed lift and drag coefficients, the lift and drag that are produced are functions of velocity and the current density at altitude. However, the parameter of interest is the velocity of the parafoil. For a parafoil in equilibrium, equation [25] can be rewritten as:

$$mg = L\cos(\gamma_d) + D\sin(\gamma_d) \qquad [31]$$

Equation [31] along with equations [29] and [30] can now be rewritten as:

$$mg = \frac{1}{2}\rho \mathrm{V_r}^2 \mathrm{S}(\mathrm{C_L}\cos(\gamma_d) + \mathrm{C_D}\sin(\gamma_d)) \qquad [32]$$

This equation can then be further simplified with a bit of geometry that relates the drag and lift with respect to glide angle and weight.

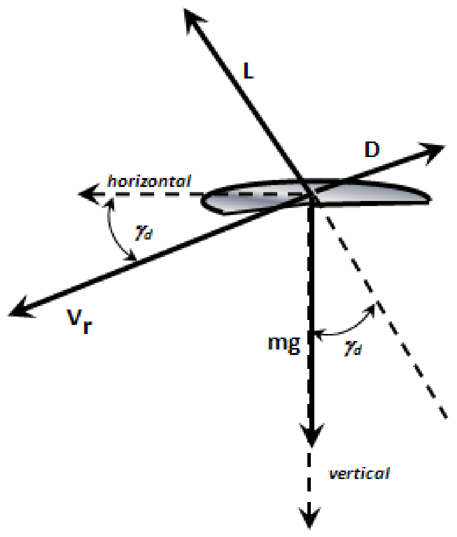


Figure 24 Lift and Drag in Steady State Glide

Figure 24, shows the relationship between lift and drag and can be used to solve for the following equation.

$$\mathrm{mg} = \frac{1}{2}\rho V_r^{\,2} S\left(\frac{C_L}{\cos(\gamma_d)}\right) \tag{33}$$

Solving for glide velocity:

$$V_r = \sqrt{\frac{2\cos(\gamma_d)}{\rho C_L}\frac{mg}{S}} \tag{34}$$

From this equation, we can see that the glide velocity depends on three factors; wing loading, density, and glide angle. Under normal flight conditions, wing loading will not change during a mission, however, the remaining two parameters can change. Density does change as a function of altitude, in fact as the vehicle descends its glide velocity will decrease as the density increases. The last variable that can change is the glide slope angle. As can be seen in equation [28], the glide slope is directly a function of the lift and drag that is being produced by the system. If control over the rigging angle (ϕ) is allowed, then the effective angle of attack of the parafoil can be changed as well.

The relationship between the glide angle and angle of attack (α) is shown in Figure 23 as:

$$\gamma_d = \alpha + \phi \qquad [35]$$

From equation [35], it can be seen that the purpose of the rigging angle is to vary the angle of attack, which in turn affects the glide angle. By allowing control of rigging angle, it becomes possible to vary the lift and drag ratio of the parafoil while. Experimental results reported by Nicolaides (4), for varying parafoils of different aspect ratio are shown below in Figure 25 and Figure 26 for lift and drag coefficients respectively.

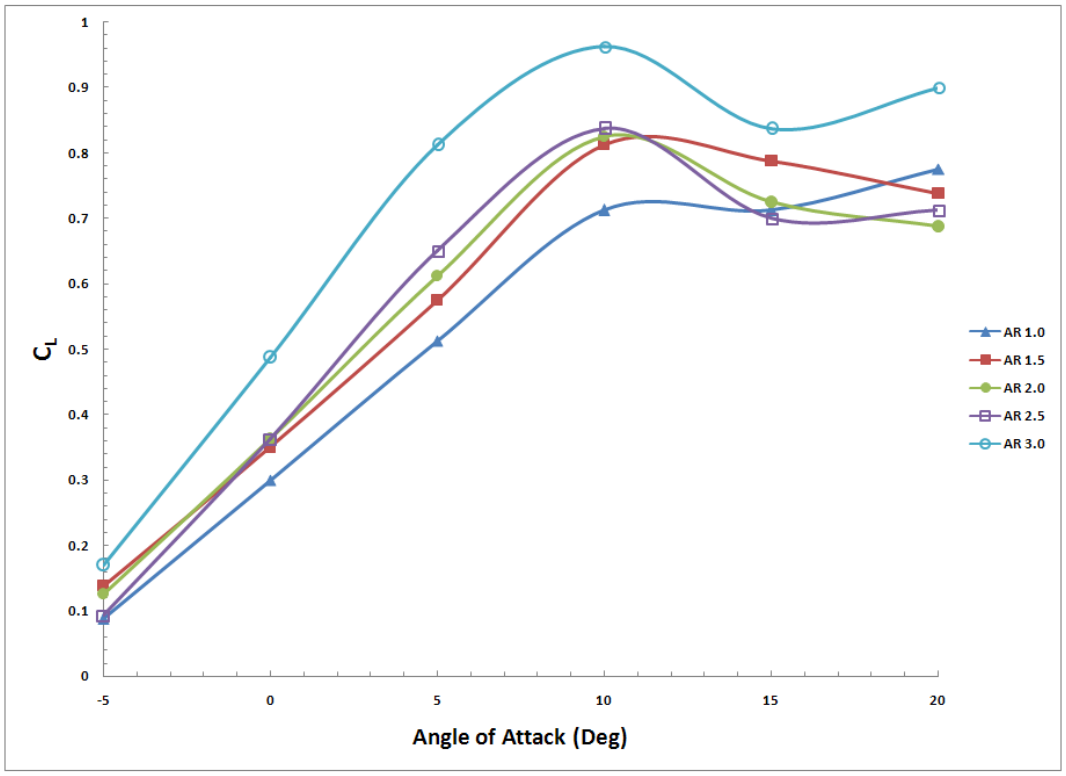


Figure 25 Lift Coefficients Versus Angle of Attack for Parafoils of Varying Aspect Ratios

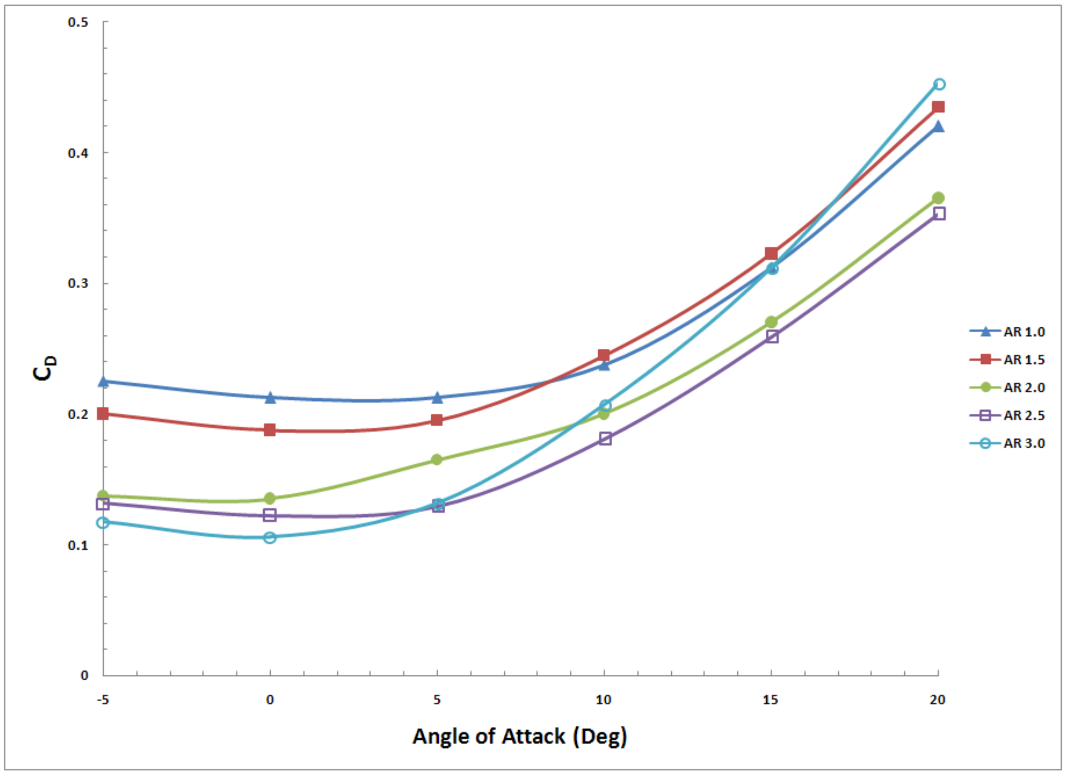


Figure 26 Drag Coefficients Versus Angle of Attack for Parafoils of Varying Aspect Ratios

While the information presented in the figures above is useful to determine the performance of the parafoil, it is not as important as the lift to drag ratio. As seen previously, it is the lift to drag ratio that determines the glide angle during descent. The resulting lift to drag ratio for the figures above are shown in Figure 27.

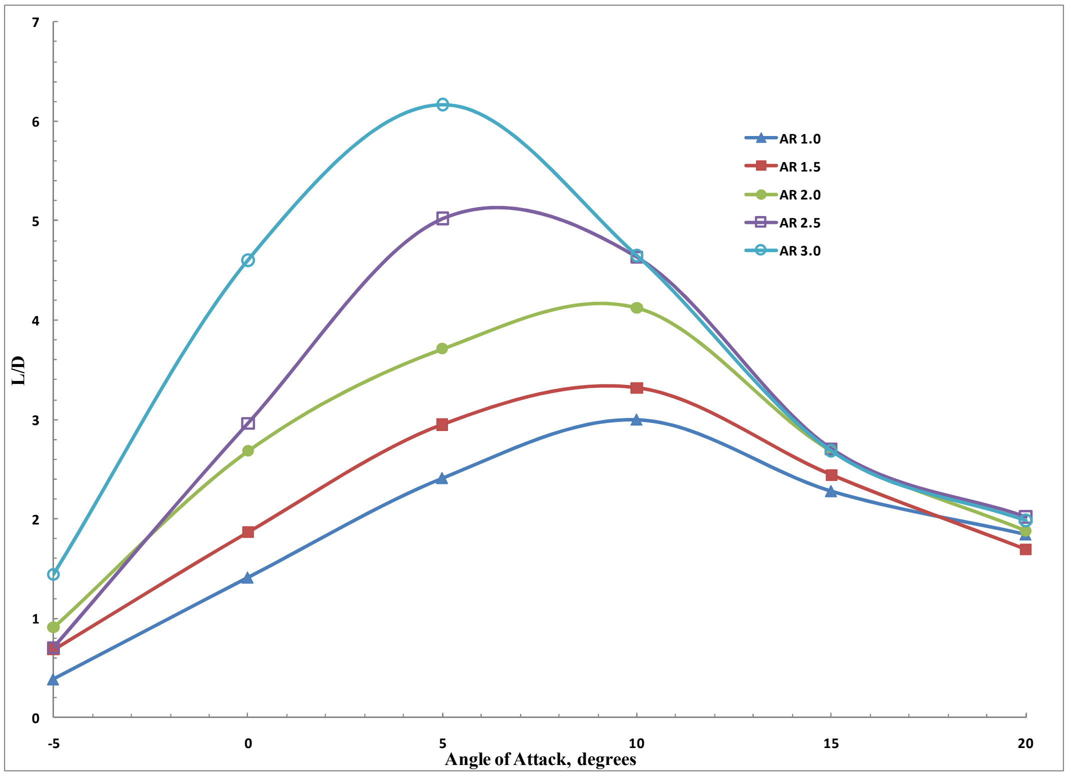


Figure 27 Lift to Drag Ratio versus Aspect Ratio Summary

If we assume that the parafoil to be used will be a commercial-off-the-shelf (COTS) product then there are only a limited number of aspect ratios that are readily available. Typical COTS parafoils have published aspect ratio between two and three, which indicates that lift to drag ratios between three and six are possible.

Now that we have developed the equations that govern the angle of descent as well as the glide velocity, a new variable can be solved. This new variable is the rate of descent, or sinking speed, of the parafoil. This rate of descent can be solved by resolving the glide velocity into its vertical and horizontal components. From Figure 24, the vertical and horizontal components of the parafoils resulting velocity can be found to be the following:

$$\mathbf{u = V_r \cos(\gamma_d)} \quad \text{[36]}$$

$$\mathbf{w = V_r \ \mathit{sin}(\gamma_d)} \quad \text{[37]}$$

Substituting equation [34] and applying geometry shown in Figure 24 the equations above become:

$$u = \sqrt{\frac{2}{\rho C_L}\frac{mg}{S}\left(\frac{C_L}{\sqrt{C_L^2 + C_D^2}}\right)\frac{C_L}{\sqrt{C_L^2 + C_D^2}}} \tag{38}$$

$$w = \sqrt{\frac{2}{\rho C_L}\frac{mg}{S}\left(\frac{C_L}{\sqrt{C_L^2 + C_D^2}}\right)}\frac{C_D}{\sqrt{C_L^2 + C_D^2}} \tag{39}$$

Simplifying the above equations we obtain:

$$u = \sqrt{\frac{2}{\rho}\frac{mg}{S}}\frac{C_L}{k} \tag{40}$$

$$w = \sqrt{\frac{2}{\rho}\frac{mg}{S}}\frac{C_D}{k} \tag{41}$$

Where: $k = \sqrt[3/4]{C_L^2 + C_D^2}$

Now that we can determine the horizontal and vertical components of velocity, we need to consider how to control them. As was seen, the vertical velocity can be controlled by changing the rigging angle, which in turn changes the angle of attack of the parafoil. Once a new angle of attack is established the parafoil will transition to a new descent angle. For the purpose of this project, change of the rigging angle will not be investigated. Instead, a fixed rigging angle will be set during mission preparation. The main focus of this project will be controlling the parafoil in the horizontal plane. To control the parafoil in the horizontal plane, its risers are pulled in the direction of the turn desired. By pulling risers a yaw rate is established which allows the parafoil to change its heading in the horizontal plane.

The equations for motion in the horizontal plane along with the effects of wind are developed in the following equations.

$$\frac{dx}{dt} = V_r \cos\omega + V_W \cos\varphi \tag{42}$$

$$\frac{dy}{dt} = V_r \sin\omega + V_W \sin\varphi \tag{43}$$

Using the two equations above, along with equations [22] and [23], a numerical simulation was written to test the feasibility of using a parafoil to return payloads to a landing zone. In order to determine the 'range' that is possible for the parafoil and system, no controller was used, instead a numerical simulation was run for constant headings. The simulations where conducted with the following conditions: winds aloft information that was collected during the ASTRO-13 mission, and initial conditions that allowed the ascent rates to match results obtained during ASTRO-13. The results for the parafoil descent where then obtained from three different altitudes; 50k, 75k, and 100k feet and are shown in Figure 28.

From the results seen in Figure 28, the longest range of the parafoil is in the direction of the prevailing winds, however it is possible for the system to penetrate upwind. One item found to be of interest is the prospect that the total number of landing locations may be kept small as it appears that under typical flight conditions the possible landing zones that the higher altitude zones contain, also contain the landing zones of lower altitudes

.

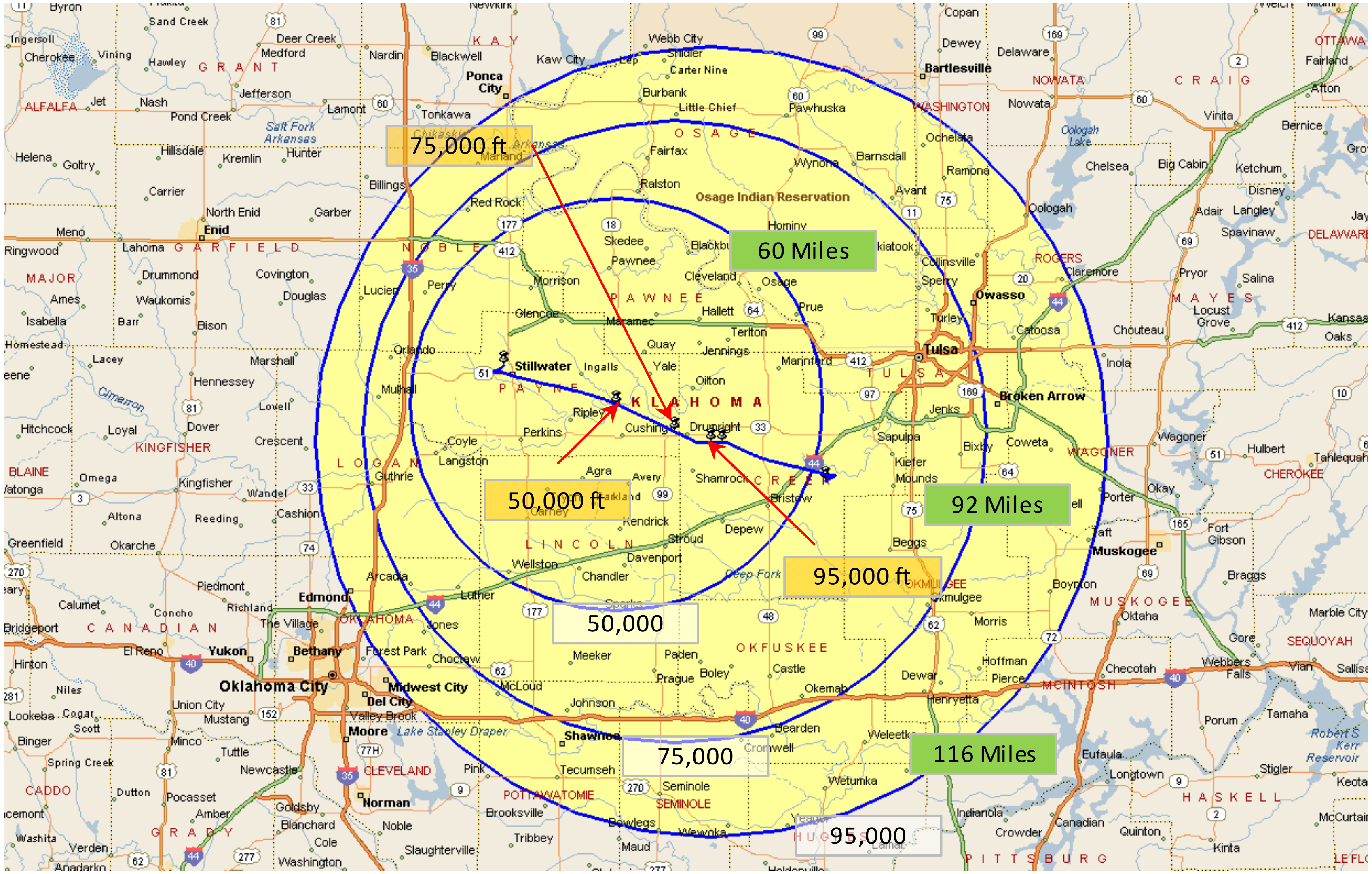


Figure 28 Constant Heading Reachable Landing Zones

REAL-TIME PREDICTION OF SYSTEM PERFORMANCE

With an understanding of both the ascent and descent phases of the mission, we are able to predict a mission before it is flown to see in general how the mission will perform. Unfortunately, no matter how much care is taken there will all ways be differences between how the mission really performs and the behavior of the predicted mission. In order to get a better solution to the location of the landing-zone, some method of real time tracking and performance correction is required.

In order to correct for the behavior of the system in real-time, a method of communication had to be established. One method of communication that has been well established is currently used by amateur radio operators around the world to report the position of mobile stations during field operations. This method is called Automatic Position Reporting System (APRS) and was originally designed by Bob Bruninga, WB4APR, and introduced in 1992 at the TAPR/ARRL Digital Communications Conference. (5)

Fundamentally, APRS is a packet communications protocol that is used for disseminating live data to everyone on a network in real time. One of the major features of APRS is that it allows quick dissemination of an operator's position, course, and speed to any other operators monitoring. APRS is actually composed of two different message formats compressed and uncompressed both formats contain current status of longitude, latitude, altitude, ground speed, and ground heading. The major difference between the two is that the uncompressed format does not contain a time stamp as to when the information was transmitted, but this can be overcome by time stamping the packet when it arrives and ensuring that we are not operating in an environment where packets can be rebroadcast or digipeated[1].

W2OSU-11>APT311,WIDE3-3:/145744h3559.06N/09701.31WO318/000/A=000928
Uncompressed APRS Format
W2OSU-11>3U5Y0V,WIDE3-3:`}Y;l!HO/"7%}
Mic-E APRS Format

Figure 29 APRS Formats

Both packets shown in Figure 29 contain the same information with the exception of the time stamp. The compressed format has an advantage in that it is shorter and therefore takes less time to transmit so there is a smaller chance of a packet error to occur. One advantage of the uncompressed format is that one could decode this packet easily by hand if needed, however, since the APRS message is being decoded by software this does not really factor into the picture.

Regardless of the format, one of the major pieces of information that is not directly available from APRS is ascent and descent velocity. Since the ascent and descent velocities are one of the major system parameters that are needed in order to determine how the system is behaving, a method to indirectly obtain this information was needed.

Since most APRS packets contained altitude and time stamps, it is possible to obtain the velocity information using a central difference method. The problem with this method is that there is no guaranty that the packets will arrive and be decoded successfully at even time

[1] Digipeating is the process of retransmit a packet radio transmission, but unlike a repeater that receives on one frequency and transmits on another frequency, the packet is transmitted on the same frequency as it was received.

intervals. In order to resolve the problem of uneven time intervals, it was decided to use the method of Least Squares to produce a linear curve fit over a small moving flight window. To fit the data to this line, the current altitudes, as well as a time stamps, are required. Since it is not always clear as to when the packets might arrive, it was decided to record the time duration between each packet as well as determine the current time that has occurred since the start of the mission. The result was an equation that took on the following form:

$$\boldsymbol{altitude = m(mission_{time}) + const} \qquad [44]$$

If we take a closer look at equation [44], we can see that m is actually the currently ascent or descent velocity and thus allows us to determine velocity as a function of altitude and mission time. This information was then used to create a similar profile in the simulation of both the ascent and descent phases of the mission. When the system switches from ascent to the descent phase the method of least squares was reset to zero. Now that we are able to determine the velocity of the real mission and that of the simulation, we can focus on how to correct the physical parameters of the simulated system so that we can more closely predict the behavior of the system. This will allow us to more accurately predict the landing zone. In order to do this, we are going to have to look at potential sources of error that govern the ascent and descent phases of the mission.

Ascent Phase Uncertainty

The ascent phase is governed by three major forces: Weight of the entire system (W), the drag force (D), and the gross lift on the system (B). The governing equation for the simplified balloon/payload system is:

$$\boldsymbol{B - W - D = ma} \tag{45}$$

If we look at equation [46], we can see directly there are three major sources of error that can occur during the ascent phase: weight, buoyancy, and drag. If we assume that each of these variables can have an error associated with them, then the equation can be written as:

$$\boldsymbol{\gamma_B B - \gamma_W W - \gamma_D D = ma} \tag{46}$$

The weight can also be written as:

$$\boldsymbol{\gamma_B B - \gamma_W W - \gamma_D D = \frac{\gamma_W W}{g} a} \tag{47}$$

If we now assume that at any given time a quasi-steady solution is valid, then the equation above can be reduced to:

$$\boldsymbol{\gamma_B B - \gamma_W W - \gamma_D D = 0} \tag{48}$$

Since weight is only parameter that can be directly measured, we are going to normalize and solve the equation as follows:

$$\boldsymbol{\frac{\gamma_B B}{\gamma_W W} - 1 - \frac{\gamma_D D}{\gamma_W W} = 0} \tag{49}$$

The problem with the equation above is that we will still have to determine the correction factors so that we can then predict the behavior of the system in real-time without direct access to any of the three. In order to solve this problem we, will need to solve equation [49] with respect to known variables.

$$\boldsymbol{B - W - \frac{1}{2}\rho C_d A V^2 = 0} \tag{50}$$

Solving for velocity the equation becomes:

$$\boldsymbol{V^2 = 2\frac{B - W}{\rho C_d A}} \qquad [51]$$

Now using the equations above, we can see that the ascent velocity is governed by five parameters. Buoyancy (*B*) is directly controlled by the amount of Helium that is currently in the balloon. Weight (*W*) is simply the total weight of the system, which includes the weight of the balloon, Helium, and payloads. Density (*ρ*) is a function of altitude and for this project is assumed to follow standard atmospheric trends. Area (*A*) is the cross sectional area of the balloon and is a function of the amount of Helium in the balloon as well as atmospheric conditions, pressure and temperature, at altitude. Finally, drag coefficient, as seen in the previous sections, is a function of ascent velocity and cross sectional area of the balloon. The main problem that is now encountered in equation [51] is that it can be difficult to resolve the currently velocity if we are not able to accurately determine buoyancy, weight, density, balloon volume and drag as the balloon ascends. However with care the only parameter that cannot be directly measured before the start of the mission is the drag term.

$$\boldsymbol{D = \frac{1}{2}\rho C_d A V^2} \qquad [52]$$

To measure the buoyancy force, we must recall from earlier discussions that there are three different categories of lift that act on the balloon; gross lift, nozzle lift, and free lift, all of which can be determined at time of launch. Gross lift, which is also known as total buoyancy, while not directly measureable can be inferred from nozzle lift. If we can determine the amount of nozzle lift at the time of launch then we can indirectly determine the buoyancy force acting on the system by adding the balloons weight to nozzle lift. This information then allows equation [48] to reduce to the following:

$$B - W - \gamma_D D = 0 \tag{53}$$

If we now expand this equation we obtain:

$$B - W - \gamma_D \frac{1}{2}\rho C_d A V_{sim}{}^2 = 0 \tag{54}$$

Once again we solve for velocity and the result is:

$$V_{sim}{}^2 = \frac{2}{\gamma_D}\frac{B - W}{\rho C_d A} \tag{55}$$

If we now back substitute equation [51] into the equation above:

$$V_{sim}{}^2 = \frac{V_{real}{}^2}{\gamma_D} \tag{56}$$

With this result, we now see that with the ability to determine the velocity during ascent from data obtained in real-time. We are also able to determine a correction factor γ_D that can be used in the simulation to adjust the model to a behavior that more closely represents the real system with the correction factor being found by:

$$\gamma_D = \frac{V_{real}{}^2}{V_{sim}{}^2} \tag{57}$$

With the correction factor now found, we can correct for any changes in the system's performance caused by an error in drag. To implement this correction factor the real-time performance predictor uses equation [46] in the following form:

$$\left(\frac{B}{W} - 1 - \gamma_D \frac{D}{W}\right) g = a \tag{58}$$

To see how well this correction factor works, we ran a simple benchmark test, where the only parameter that would change was the drag coefficients. We looked at the effects of four cases and compared to a baseline case. The four cases selected where 80%, 90%, 110%, and 120% of the baseline balloon drag coefficient model. The results of this test are shown in Figure 30 and indicate that as the error in the drag coefficient increases the time required to reach

30,000 meters also increases. Since we now had results of several different cases, we wanted to see how the correction factor changed as a function of altitude. The results, shown in Figure 32, demonstrate that they correction factor is not constant. From the results at lower altitudes, typically below 10,000 meters, the correction factor is the lowest and is nearly constant. Between 10,000 and 15,000 meters, the correction factor increases rapidly and then above 15,000, meters returns to a near constant value.

In order to see the effect of the correction factor on the results, we decided to apply the correction factor to the 120% case. The selection of this case was primarily chosen due to this case having one of the largest errors in time to reach 30,000 meters, as can be seen in Figure 31.

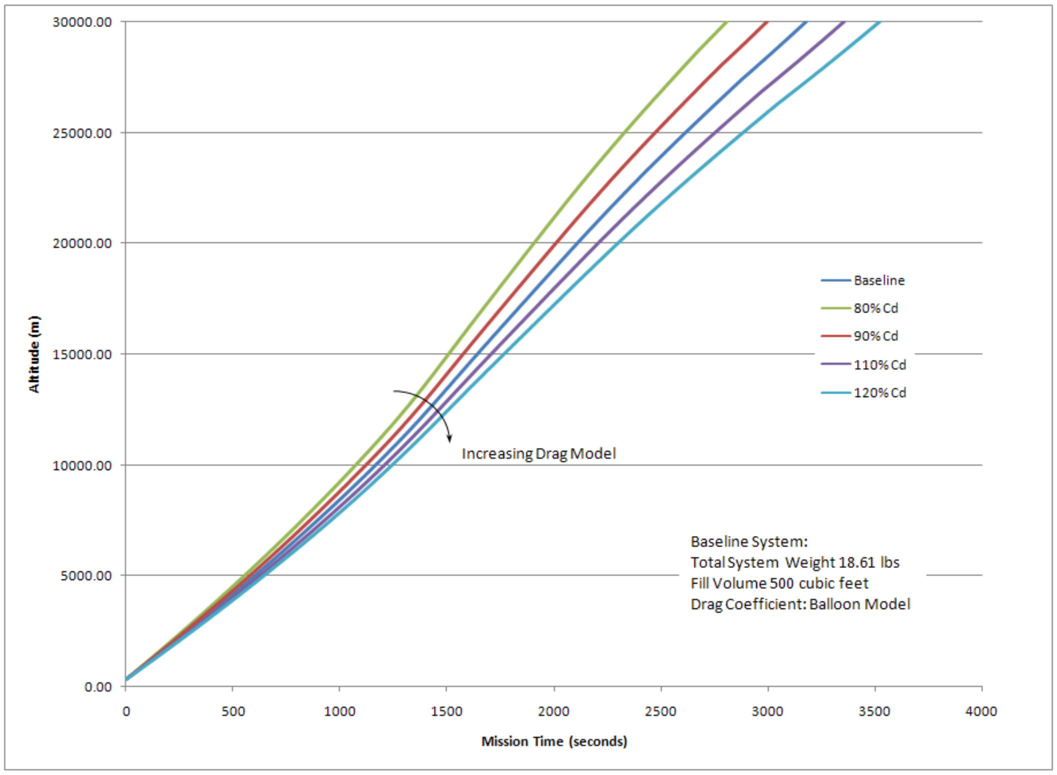


Figure 30 Effect of Drag Model on Ascent Phase Profile

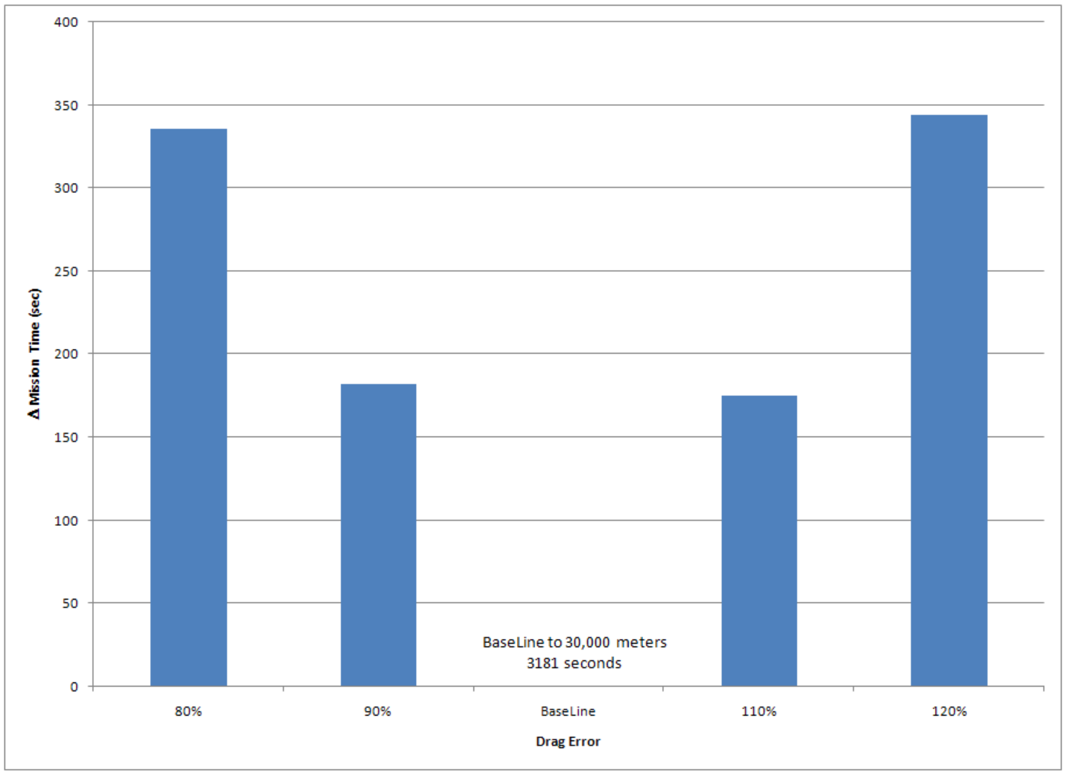


Figure 31 Differnce in Time to 30,000 meters

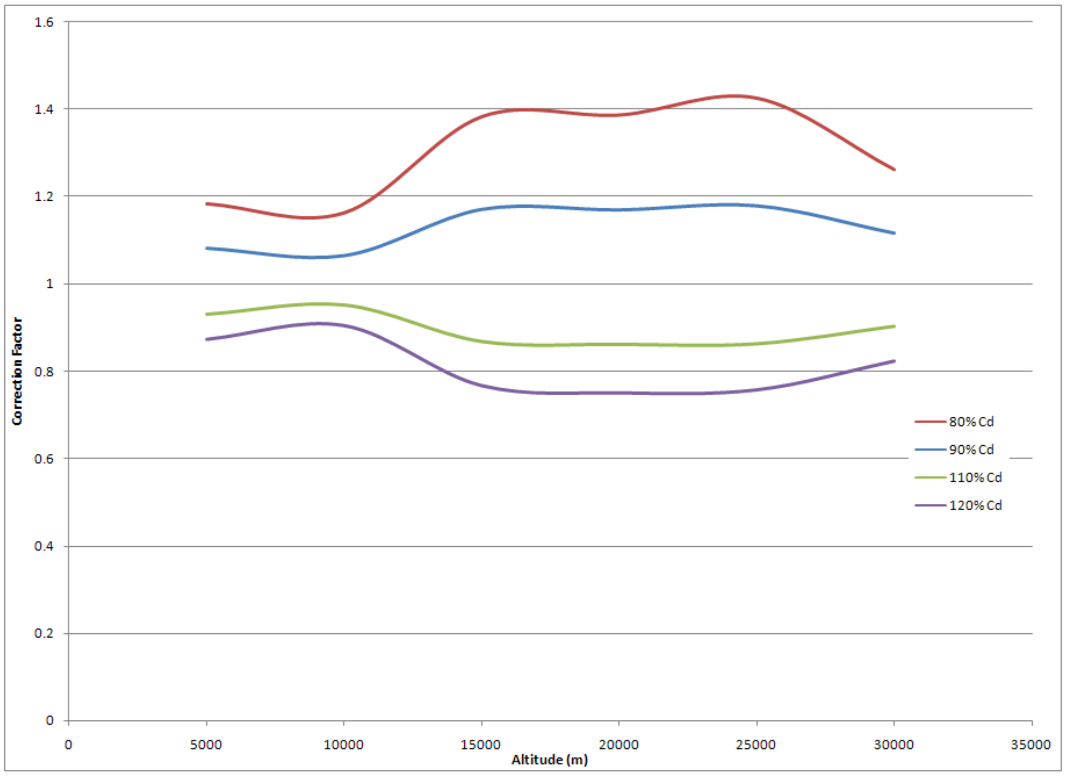


Figure 32 Variation of Correction Factors as Function of Altitude for the varying Cd

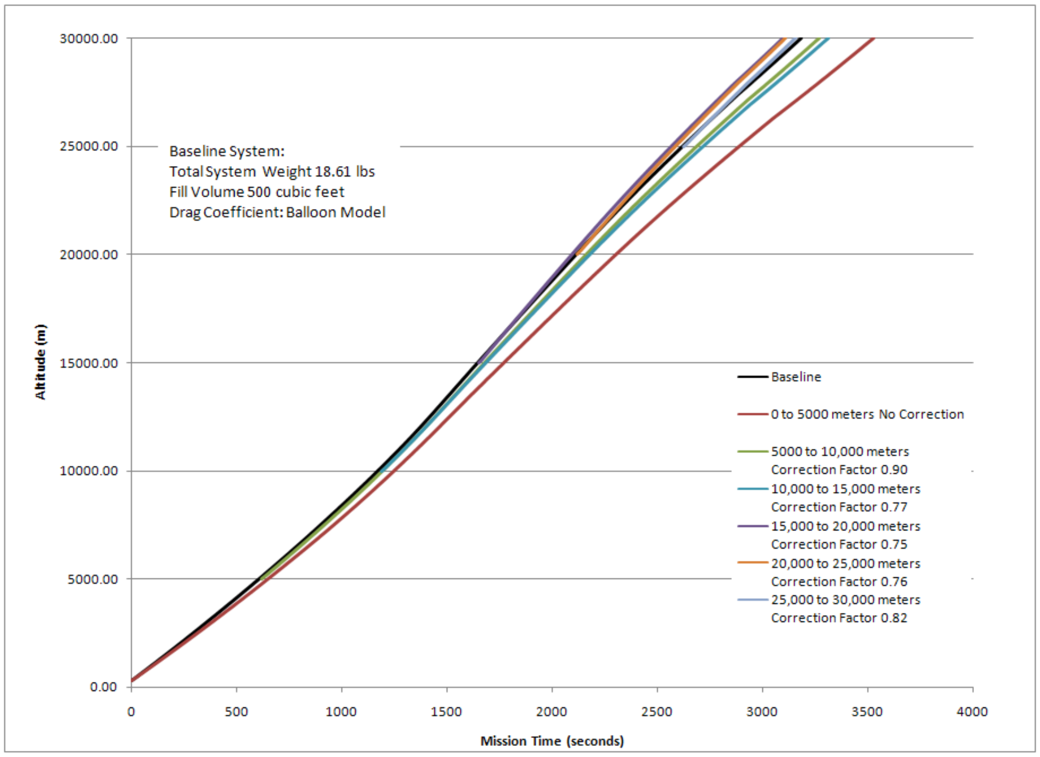


Figure 33 Effect of Drag Model on Ascent Phase Profile with Correction Factor

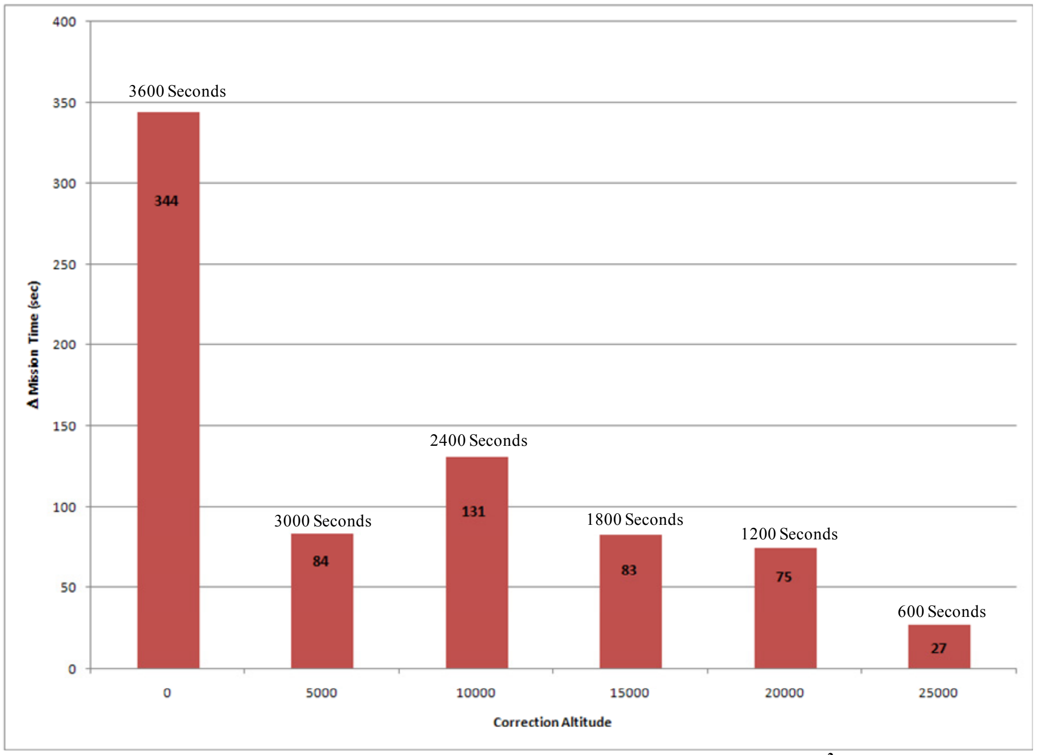


Figure 34 Differnce in Time to 30,000 meters with Correction Factors[2]

[2] Note: Times above bars is the total time required to reach 30,000 meters.

Unfortunately, two major problems remain with predicting the overall performance during the ascent phase. The first problem that we are not currently able to correct for is changes in the winds aloft from the winds information that was obtained from NOAA's Air Resources Laboratory (ARL) Real-time Environmental Applications and Display sYstem (READY). The wind aloft information is available from the Global Forecast System (GFS) for the entire world and allows predictions of the winds from the surface to an altitude of approximately 26km. As with most weather predictions, the information contained can change dramatically depending on local conditions and even more so if the area of predicted flight is near the current location of the Jet stream.

The results on a missions predicted landing zone from the variations in predicted wind starting approximately 168hrs before the launch are shown in Figure 35. The results show that it is possible to have a very large variation in the overall magnitude of the predicted flight path. Having this in mind, care must be taken to obtain the most current prediction before start of the mission. Even with the most current predictions, there will still be errors in the magnitudes of the winds.

One other problem with the predicted winds is that even at best they typically top out at 86k feet. Since it is possible for a mission to reach altitudes of 120k feet, any winds above 86k feet are typically assumed to remain constant to the point of burst. This assumption does increase the error of the predicted flight profile, until such time when this information has been updated during the mission.

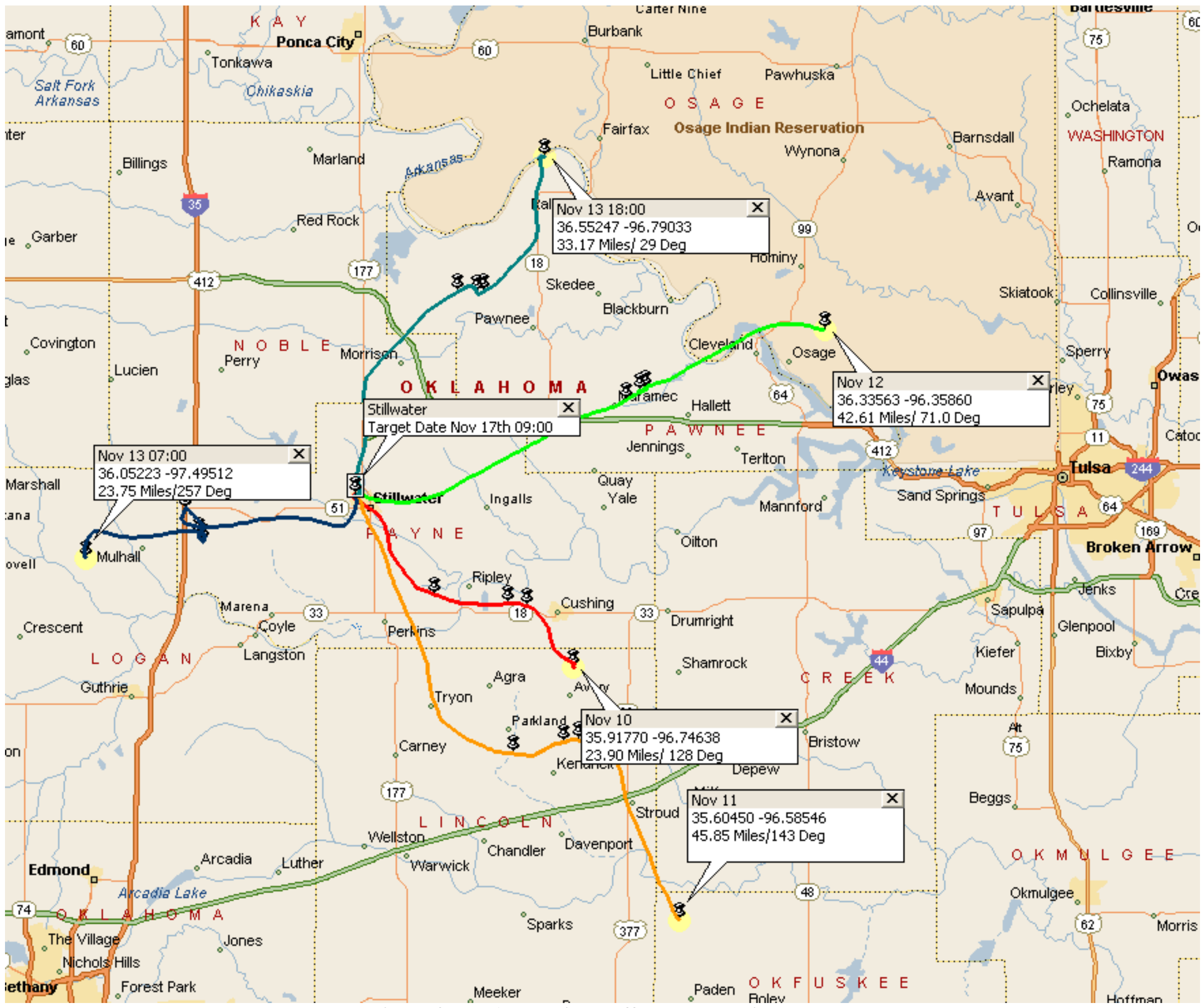


Figure 35 READY Winds Effects on Landing Zone

The second problem is the altitude of the burst point. While the manufacture states burst diameter, many things can affect this. As was seen in Figure 8, as altitude increases the rate at which the balloon’s diameter decreases and as such the burst altitude can vary by 10,000 feet or more. Of the two problems, this one is by far the easiest to counter. The first and most straight forward method is to select an altitude where we wish the mission to translate from ascent to descent and then issue a remote commanded cut-down at altitude. The second method, which does not allow as much control, is to simply adjust the predicted burst altitude to match an altitude that exceeds the current altitude of the system.

Descent Phase Uncertainty

The descent phase begins once the balloon has either burst or a remote cut-down has been issued. In many ways, the descent phase is more critical due to the lack of time one has to determine the performance parameters of the system. Some of the major difficulties that can occur in this phase are the possibility for unknown weight of the system due to any remaining balloon remnants and or a lack of information on how the chute has deployed. If we look at the governing equation for the descent velocity, it breaks down to the following:

$$\boldsymbol{W} - \mathbf{D} = \boldsymbol{m a_z} \qquad [59]$$

which can be expanded to:

$$\boldsymbol{W} - \frac{1}{2}\rho V^2 (C_{D_s} S_s) = \frac{\boldsymbol{W}}{\boldsymbol{g}} \boldsymbol{a_z} \qquad [60]$$

Using the equation above, we can see that descent velocity is governed by four parameters. Weight (*W*) is simply the total weight of the system, which includes the weight of any balloon remnants that might remain attached and payloads, cross sectional area of the chute (S_s) during descent, the density (ρ) of air, and the drag coefficient (C_d). For the purpose of this project, the drag coefficient of a spherical chute is assumed to be a constant of 1.5. If we now assume that at any given altitude a quasi-steady state solution is valid then equation [60] reduces to:

$$\frac{1}{2}\rho V^2 (C_{D_s} S_s) = \boldsymbol{W} \qquad [61]$$

Using this result and assigning an error correction to each of the major parameters, we obtain:

$$\frac{1}{2}\boldsymbol{\gamma}_\rho \rho V^2 (\boldsymbol{\gamma}_D C_{D_s} \boldsymbol{\gamma}_S S_s) = \boldsymbol{\gamma}_W \boldsymbol{W} \qquad [62]$$

Solving for the predicted velocity from the system simulation we obtain:

$$V_{sim}{}^2 = \frac{\gamma_W}{\gamma_S \gamma_D \gamma_\rho} \frac{2W}{(\rho C_{D_s} S_s)} \quad [63]$$

If we assume that the system's real behavior is represented by equation [61] then we can use it to back substitute into equation [63] and obtain:

$$V_{sim}{}^2 = \frac{\gamma_W}{\gamma_S \gamma_{Cd} \gamma_\rho} V_{real}{}^2 = \frac{\gamma_W}{\gamma_D} V_{real}{}^2 \quad [64]$$

$$\frac{\gamma_D}{\gamma_W} = \frac{V_{real}{}^2}{V_{sim}{}^2} = \gamma_{descent} \quad [65]$$

With this result we can now be correct for any changes in the system's performance caused by an error in any of the four governing parameters discussed early. The correction factor in the real-time performance predictor uses equation [60] in the following form:

$$\left\{1 - \gamma_{descent}\left(\frac{D}{W}\right)\right\} g = a_z \quad [66]$$

Upon inspection of equation [66] we note that the acceleration is now a direct function of the drag to weight ratio. To see how well this holds, it was decided to simulate the descent phase over several different payload weights while maintaining this ratio. The runs made are at weights of 6, 8, 10, and 12 lbs while maintaining a drag to weight ratio of one. The results shown in Figure 36 indicate that even if the drag to weight ratio is held constant, there is an effect on the overall descent profile. For example, a six pound payload with six pounds of drag will reach a landing altitude in just over 1660 seconds, where as the twelve pound system, with twelve pounds of drag, reaches the same altitude in just over 2370 seconds.

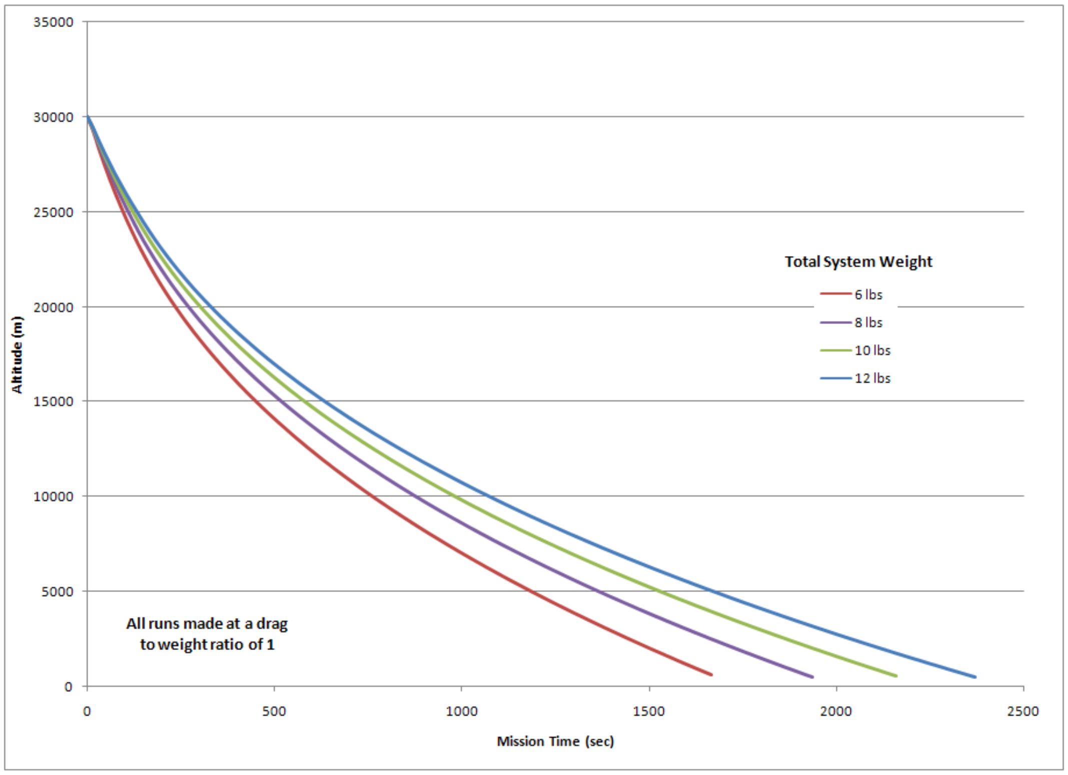


Figure 36 Effect of System Weight with Constant D/W on Descent Profile

With the results above, we can now try to apply the correction factor to the data sets above and see the results. In order to perform this, we are going to select the 12 pound system as the 'baseline' or what the real system is doing. The next step is to determine the velocity of each of the systems at a given altitude. For this purpose we selected 26,000 meters, as this was the altitude that the system reached in 120 seconds. Once this altitude was selected, the other three configurations from Figure 36 where simulated to determine the velocity of each at the altitude selected previous.

System	Velocity at 26,0000 meters	Correction Factor
12 lbs	33.94 m/s	1
10 lbs	36.21 m/s	1.24
8 lbs	41.18 m/s	1.60
6 lbs	45.56 m/s	1.96

Figure 37 Correction Factors for varying System Weights with Constant D/W

The correction factors above are applied to the previous runs, and the results are plotted in the figure below.

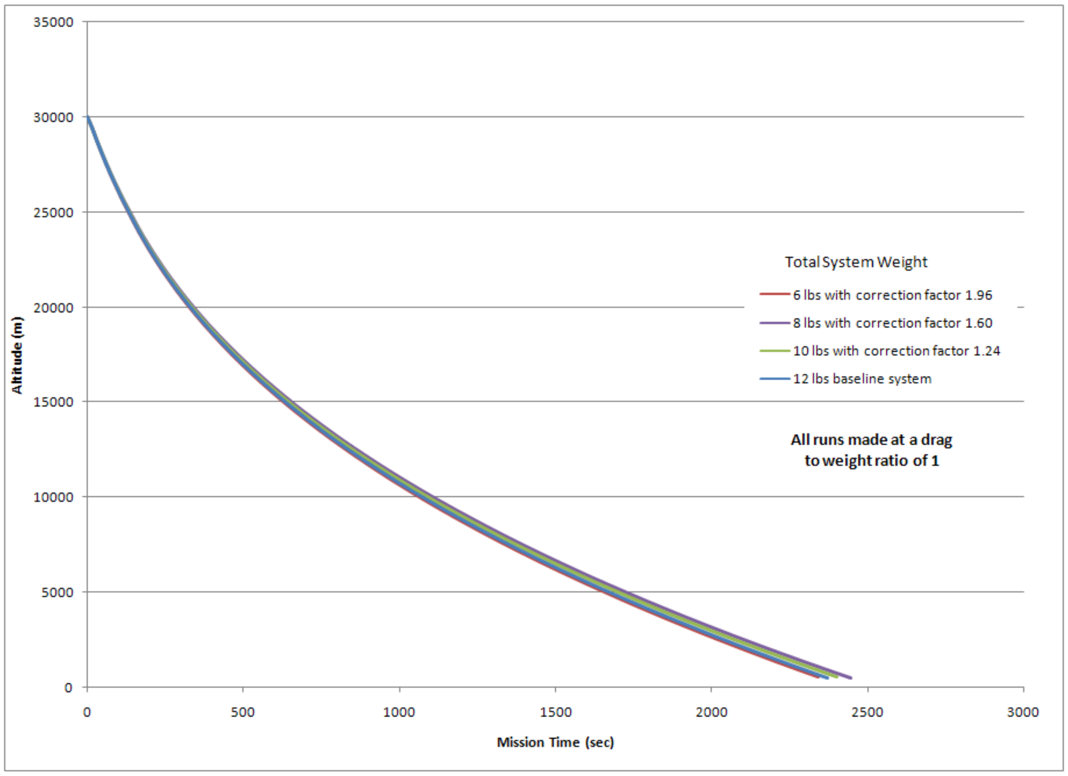


Figure 38 Effect of Correction Factor on Descent Profile with Systems at Constant D/W

As can be seen in Figure 38, the correction factor allows the various simulated systems to converge on the same solution that resulted in the baseline. While this works well, we decided to test this correction factor on several more realistic cases. The first case was a system that had a chute that deployed as expected, but the system weight varied from 18 lbs to 8 lbs with a baseline weight of 12 lbs. As expected, the results in Figure 39 show that as the payload increases, the descent velocity increases.

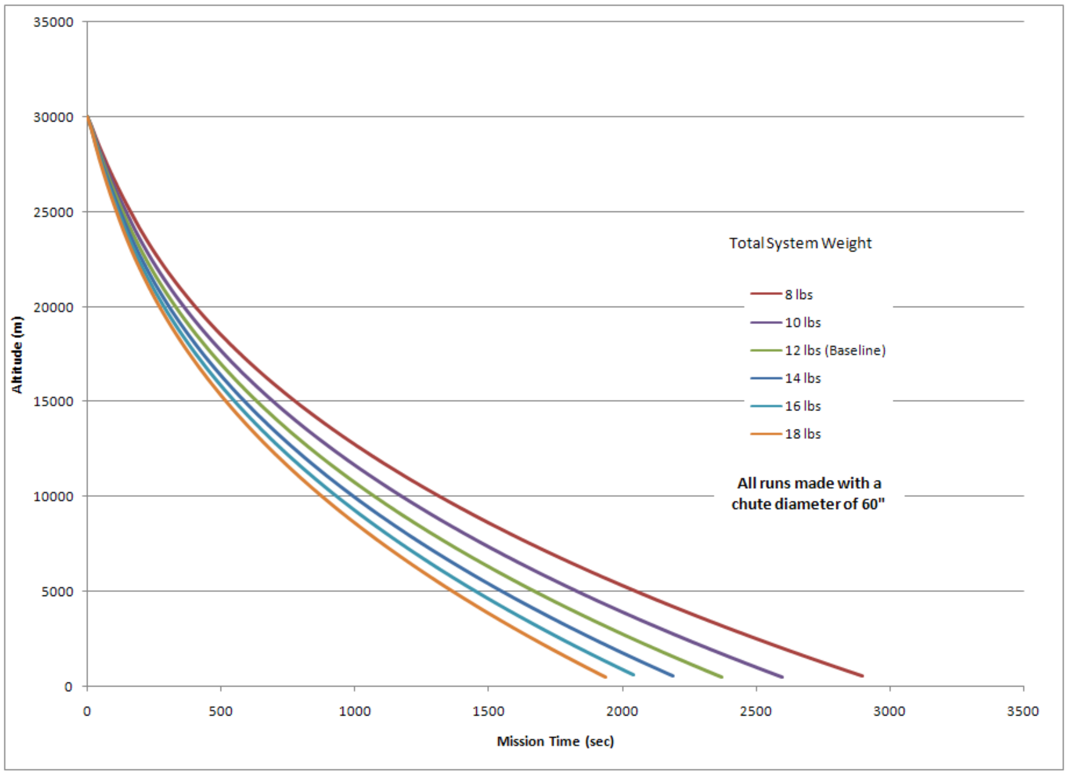


Figure 39 Effect of System Weight with Fixed Chute Diameter on Descent Profile

The results from the varying system weights are then adjusted with the correction factor. The results shown in Figure 40 illustrate that the correction factor does correctly adjust the systems. One way to analyze this data is to look at the total mission time for all six cases.

Weight	Mission Time before	Correction Factor	Mission Time after
8 lbs	2895 seconds	0.65	2340 seconds
10 lbs	2595 seconds	0.82	2340 seconds
12 lbs	2370 seconds	baseline	2370 seconds
14 lbs	2190 seconds	1.21	2415 seconds
16 lbs	2040 seconds	1.33	2355 seconds
18 lbs	1935 seconds	1.59	2430 seconds

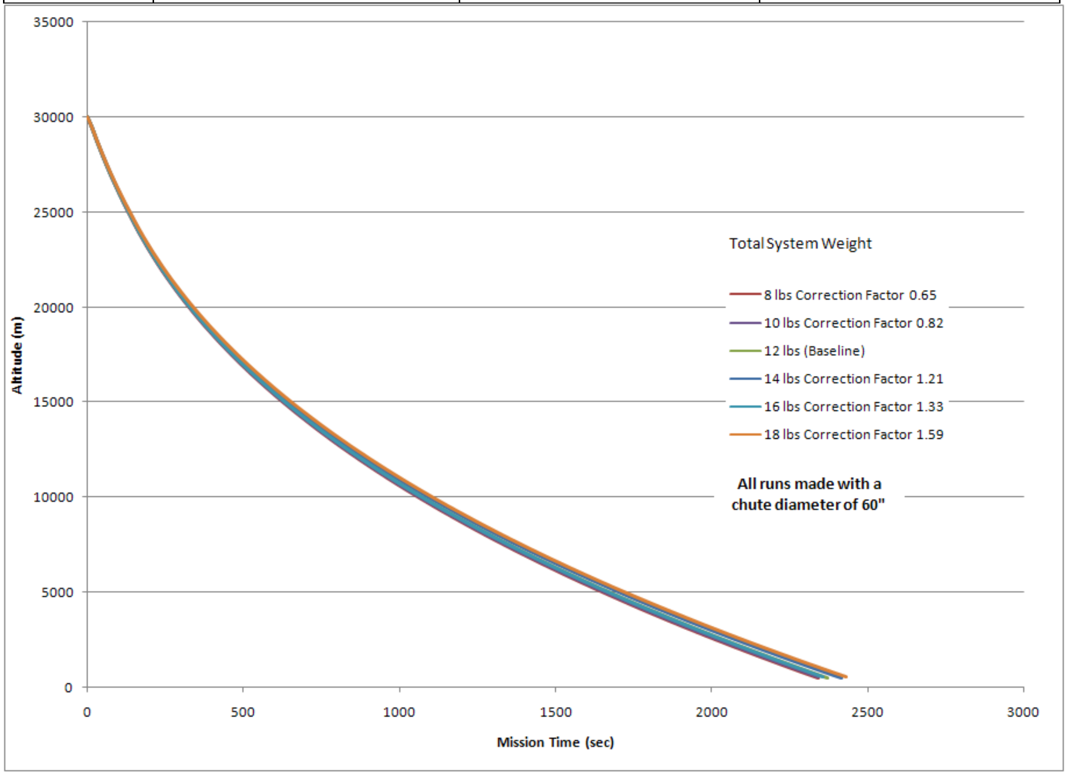


Figure 40 Effect of System Weight with Fixed Chute Diameter on Descent Profile with Correction Factor

REAL-TIME PERFORMANCE PREDICTOR

Now with an understanding of how to create the system parameters during the ascent and descent phases of the mission, we decided to test the concept on several full missions. The data collected for ASTRO missions 11 through 14 was used here because that data was recorded in a format that allowed play-back in real time. Missions flown before ASTRO 11 are not readily able to be used, as the time of arrival packets was not stored, and realistic playback is not possible.

Real-Time Tracker

In order to perform the real-time tracking, a program was written that is able to decode and parse the APRS packets sent by the system. The function of the real-time tracker is to provide the tracking crew a visual way to determine the current location of the system. Since this process is supported by visual aids, the results of the tracker can also be overlaid on Microsoft MapPoint.

Decode APRS

The primary function of the tracker is to decode and then parse out the packets as they arrive from the vehicle. One of the first items that must be detected before we can successfully decode the message is in what format is the message.

Typically there are only two formats that APRS is transmitted in for near-space missions: Mic-E compressed format, or an uncompressed format. Regardless of which packet is sent both use the Amateur Packet-Radio Link-Layer Protocol (AX.25). All APRS transmissions use AX.25 UI-frames, with 9 fields of data. The AX.25 protocol frame is shown below in Figure 41.

	Flag	Destination Address	Source Address	Digipeater Address	Control	Protocol ID	Information	FCS	Flag
Bytes:	1	7	7	0-56	1	1	1-256	2	1

Figure 41 AX.25 UI-Frame Format

The flag field at each end of the frame is the bit sequence 0x7e that separates each frame. This separation is handled by the hardware on the decoder and does not actually appear in the raw packet to be processed. The entry in the frame is the Destination Address and can contain a destination callsign. If it is in Mic-E format, then it contains encoded data. The Source Address contains the callsign of the system and a Secondary Station IDentifier (SSID) of the transmitting station. If the SSID is non-zero, then it specifies a display symbol. The Digipeater Addresses contains information on how the message will be retransmitted within the APRS network. Since most balloon missions are able to reach extreme altitudes, it is standard protocol that the message not be repeated more than 3 times on the non-primary tracking.

A word of caution needs to be given here; since not all packets are transmitted with timestamps, care must be taken when it is possible to pickup digipeated signals. The digipeated signals will be older packets and will provide false status to the mission tracker. One of the best ways to ensure this does not happen is to perform primary tracking on an 'empty' channel.

Control Field and Protocol ID are used to provide a buffer between the digipeater address field and the information field. The Information Field possibly the most important field in the packet, this field contains not only information on the current position and status mission, but also contains an entry on the type of data that is in the packet. The last frame is the Frame Check Sequence and is a 16-bit code sequence that is used by APRS decoding hardware to check the integrity of a received packet. An example of packets that where transmitted during an ASTRO mission are shown below in Figure 42.

KE5CAB-11>APT311,WIDE3-3,qAR,W5LHG:/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu **Uncompressed APRS Format**
W2OSU-11>3U4U7S,WIDE3:`~0cm HO/>$7<} **Mic-E APRS Format**

Figure 42 Raw APRS Packets from ASTRO-14 at ~15:44:35 Zulu

Parsing APRS

One of the first objectives in parsing the APRS packets is locating the placement of the destination, source, and information fields. Once these fields have been located the task of extracting the data from them can begin. To help locate the start of the packets, the AX.25 protocol requires that a source path header be created, prior to sending a packet into the APRS network. The Source Path Header consists of the source, destination and digipeater callsigns and is prepended to the original data packet. This changes the packet to be parsed to the following format.

	Source Path Header	Information Field
Bytes:	1-10	1-256

Figure 43 APRS Data with Source Path Header Prepended

	Source Path Header					
	Source Callsign	**>**	**Digipeaters Callsign**	**Digipeater Callsign**	**:**	**Information**
Bytes:	1-10	1	0-81	1-9	1	1-256

Figure 44 APRS Source Path Header Format

For example using the packets shown in Figure 42, the source path header is:

KE5CAB-11>APT311,WIDE3-3,qAR,W5LHG:/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu
Uncompressed APRS Source Path Header
W2OSU-11>3U4U7S,WIDE3:`~0cm HO/>$7<}
Mic-E APRS Source Path Header

Now that the packet has been modified with a source path header the task of locating the start of the information field has become easier. The other piece of information that is required is the source callsign. The source callsign is an identification code of the target we are tracking and is known before the start of the flight. Since the source path has an established format, there are now only two characters that will have to be located. To determine the source callsign of the message, a search of the raw APRS packet is conducted to determine the location of the character '>'. Once this character has been located, then the source callsign can be parsed by selecting all characters to its left. To determine the source path, the location of the character ':' is found. Once we've located the end of the source path, we can process the remaining message as the information field. The information field has the following generic format:

	Data Type ID	**APRS Data**	**APRS Data Extension**	**Comment(s)**	**Total Bytes: (1-256)**
Bytes:	1	n	7	n	

Figure 45 APRS Information Field Format

The first entry of the information field is the APRS Data Type Identifier (DTI). This entry determines the type data that follows. Since the tracker has been designed specifically for position reports, the number of data types is limited to only three: position without timestamp, position with timestamp, and Mic-E Data. The DTI for each of the data types are '!', '/', and '`' respectively. Once the DTI has been determined the appropriate parsing scheme can be used.

Uncompressed APRS Parsing

Once it has been determined that the DTI is one of the two uncompressed data packets the data can now be assumed to take on a specific format.

Time	(hms)	Latitude		Symbol Table Identifier	Longitude		Symbol Code
hhmmss	h	**ddmm.mm**	N/S	/	**Dddmm.mm**	E/W	O

Figure 46 APRS Uncompressed Data

Course (Deg)		Speed (Knots)
(001-360)	**/**	**(xxx)**

Figure 47 APRS Data Extensions

Altitude (feet)		Additional Comments
/A=xxxxxx	**/**	**n**

Figure 48 APRS Comments

In order to extract the system current status from this format, all one has to do is parse directly by location.

The following is the pseudo code for retrieving information from the packet.

1. Locate start of information field by determining the end of the source path header by locating the first encounter of ':'

KE5CAB-11>APT311,WIDE3-3,qAR,W5LHG:/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu **Raw APRS Packet**
KE5CAB-11>APT311,WIDE3-3,qAR,W5LHG: **Source Path Header**
/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu **Information Field**

2. Determine type of data by looking at the first character in the information field

   - If character is '/' then timestamp is present
   - If character is '!' then timestamp is not present
   - If character is '`' then Mic-E format. This packet format will be treated separately as it has its own unique format and will require a different method to parse correctly.

In the example packet the first character is '/' so this is a data packet that contains a timestamp.

/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu

3. Timestamp is present then decode by assigning characters as follows

   - (23) are assigned to hour , (45) are assigned to minutes, and (56) are assigned to seconds.

For example packet:

Time: 15:44:53 Zulu

/**154453h**3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu

   - Final step is to remove the first eight characters from the message. This step is done so that the following segments of parsing remain the same regardless of timestamp.

3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu

4. If timestamp is not present then assign time of arrival.

5. Parse out latitude

   - Assign (12) to degrees
   - Assign (34567) to minutes and then convert to decimal degrees
   - Determine if (8) is "S", if so then convert (dd.dddd) to (-dd.dddd)

6. Parse out longitude

   - Assign (10-12) to degrees
   - Assign (13-17) to minutes and then convert to decimal degrees
   - Determine if (18) is "W", if so then convert (ddd.dddd) to (-ddd.dddd)

For example packet:

Latitude: 34° 45.84' / 34.764° Longitude: -98° 20.58'/ -98.343°

3545.84N/**09820.58W**O042/043/A=055987/ASTRO.okstate.edu

7. Parse out information contained in Data Extensions, in this case course and speed
   - Assign (20-22) to course in degrees from true North
   - Assign (24-26) to speed in knots

For example packet:

Course: 42° Speed: 43 knots

3545.84N/09820.58WO**042**/**043**/A=055987/ASTRO.okstate.edu

8. Parse out altitude information contained at the start of the comment field.
   - Assign (30-35) to altitude in feet

For example packet:

Altitude: 55,987 feet

3545.84N/09820.58WO042/043/A=**055987**/ASTRO.okstate.edu

The overall results of decoding the example APRS Packet to obtain its mission status are:

KE5CAB-11>APT311,WIDE3-3,qAR,W5LHG:/154453h3545.84N/09820.58WO042/043/A=055987/ASTRO.okstate.edu **Raw APRS Packet**	
Time: 15:44:53 zulu	Altitude: 55,987 feet
Latitude: 34° 45.84' / 34.764°	Longitude: -98° 20.58'/ -98.343°
Course: 42°	Speed: 43 knots

Figure 49 APRS Uncompressed Packet with Timestamp Decoded to Obtain Mission Status

Mic-E APRS

One of the major advantages of the Mic-E format is that the packets are much shorter in length, then in comparison to the uncompressed format. Typically the Mic-E format contains only 37 bytes of data where as the uncompressed has 78 bytes. This size difference decreases the chance of an error in transmission, as the typically Mic-E packet requires ~450msec as compared to ~750msec for an uncompress packet. However, there are two downsides to using the Mic-E format; first is that the information contained in the packet is not time stamped at the time of transmission, and second the information in the packet is compressed and results in a packet that is not readily readable.

Mic-E Packet Format

The Mic-E data format allows a large amount of data to be carried in a very short packet. In the uncompress format the data is carried entirely in the information field, where as in the Mic-E format, the data is split between the destination address field and the information field. The destination address field contains the following information in an encoded 7-byte format: a 6-digit latitude, North/South and West/East indicators, and the longitude offset indicator. The information field then contains the rest of the current status and includes; longitude, course, speed, and altitude.

Mic-E APRS Parsing

To confirm that a packet is in Mic-E format, the first byte in the information field must be ‘`’. Once the packet is confirmed to be Mic-E then it can safely be assumed to take on a specific format. To understand the layout of the packet, we will look at the following example and see where the different fields are:

W2OSU-11>3U4U7S,WIDE3:`~0cm HO/>$7<}	
Source Field	**W2OSU-11>**
Data Destination Address field	**3U4U7S,WIDE3:**
Data Information Field	**`~0cm HO/>$7<}**

To determine the source callsign of the packet a search of the Mic-E packet is performed to determine the location of the character ‘>’, once this character has been located then the source callsign can be parsed by selecting all characters to it left. This method of locating the source callsign actually does not vary at all when compared to the uncompressed format, it is how the rest of the packet is handled that varies dramatically. One of the first items that is of major difference, when compared to the uncompressed format, is that destination field actually contains current mission status information. Before we can actually parse out the information from the destination field we have to understand how the data is compressed and encoded.

The current 6-digit latitude is stored in the first 6-bytes of the destination field. For the example given above the first six characters are **3U4U7S**. This six characters not only contain information on the systems current latitude but also contain information on; N/S latitude indicator, the longitude offset value, and the E/W longitude indicator. The information is encoded by sharing the bytes in the destination address field as shown below in Figure 50.

	Lat Digit 1 + Message Bit A	***Lat Digit 1 + Message Bit A***	***Lat Digit 3 + Message Bit C***	***Lat Digit 4 + N/S Lat Indicator***	***Lat Digit 5 + Longitude Offset***	***Lat Digit 6 + E/W Long Indicator***	***APRS Digi Path Code***
Bytes:	1	1	1	1	1	1	1

Figure 50 Mic-E Data Destination Address field

Since each byte contains actually two pieces of information a table must be used to determine how the Mic-E destination address field is encoded.

Byte:	1-6	4	5	6	Byte:	1-6	4	5	6
ASCII Char	**Latitude Digit**	**N/S**	**Longitude Offset**	**E/W**	**ASCII Char**	**Latitude Digit**	**N/S**	**Longitude Offset**	**E/W**
0	0	South	0	East	**L**		South	0	East
1	1	South	0	East	**P**	0	North	100	West
2	2	South	0	East	**Q**	1	North	100	West
3	3	South	0	East	**R**	2	North	100	West
4	4	South	0	East	**S**	3	North	100	West
5	5	South	0	East	**T**	4	North	100	West
6	6	South	0	East	**U**	5	North	100	West
7	7	South	0	East	**V**	6	North	100	West
8	8	South	0	East	**W**	7	North	100	West
9	9	South	0	East	**X**	8	North	100	West
A	0				**Y**	9	North	100	West
B	1				**Z**		North	100	West
C	2								
D	3								
E	4								
F	5	Note: A-K are not allowed in bytes 4-6.			Message is contained in bytes 1-3 and is assigned a value of zero if the ASCII char is 0-9 or L, otherwise it is assigned as one				
G	6								
H	7								
I	8								
J	9								
K									

Figure 51 Mic-E Destination Address Field Encoding

To implement this table, there are two methods can be used: One method is to utilize a brute force method and create a look-up table for the entire table, as shown above. The second method is to look for a pattern in the table and then generate a simplified look-up table. The second method was used to create a method to decode the data stored in the destination address field. If one looks at the table for how the digits for latitude are stored one will note that only the digits 0-9 are stored. If the character happens to be 0-9 then the decoder is straight forward as it retrains it value, but this leaves behind twenty other possible characters that can be stored in the message. However, if we grouped the ASCII characters by the digits they represent then the following pattern appears:

Digit	0	1	2	3	4	5	6	7	8	9
ASCII Char	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
	P	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**

Figure 52 Mic-E Latitude Digit Encoding

At this point there are now two solutions to the problem. The first method is to assign each of the three groupings to the strings; "**0123456789**", "**ABCDEFGHIJ**", and "**PQRSTUVWXY**". Then using a string search command, determine the correct string to search and then establish the characters position within the string and assign its position to the value of the digit. The second method is to analyze the first six-digits of the data destination address field by their ASCII decimal value. This method changes Figure 52 to the following format.

Digit	0	1	2	3	4	5	6	7	8	9
ASCII Dec	***48***	***49***	***50***	***51***	***52***	***53***	***54***	***55***	***56***	***57***
	65	***66***	***67***	***68***	***69***	***70***	***71***	***72***	***73***	***74***
	80	***81***	***82***	***83***	***84***	***85***	***86***	***87***	***88***	***89***

Figure 53 Mic-E Latitude Digit Encoding in ASCII Decimal Values

This result can then be shifted with respect to the first entry of each row and results in:

Digit	Offset	0	1	2	3	4	5	6	7	8	9
Modified ASCII Dec	***48***	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
	65	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
	80	***0***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***

Figure 54 Mic-E Latitude Digit Encoding with Modified ASCII Decimal Values

Either method will allow decoding of latitude data that is contained within the data destination address field.

The three pieces of information that are important to extract are the N/S latitude indicator, the longitude offset value, and the E/W longitude indicator. Once again if we look at Figure 52, we will see a pattern appear. In the Mic-E destination address field encoding information in regard to the latitude indicator is South, the longitude indicator is East and the longitude offset is zero all transition to North, West, and 100 at the same location. To determine the value of each the decoded simply needs to determine if the entries ASCII value is greater than ‘L’ and if so assign the correct value.

The following is the pseudo code for retrieving information from the destination address field.

1. Locate start of destination field by determining the location the first encounter of ‘>’, and then extract the first 7 bytes.

W2OSU-11>3U4U7S,WIDE3:`~0cm HO/>$7<} **Raw APRS Packet**
3U4U7S, **Data Destination Address Field**

2. Parse out latitude

   - Decode and assign (12) to degrees
   - Decode and assign (34) to minutes
   - Decode and assign (56) to decimal minutes

Degree (dd)		Minutes (mm)		Decimal Minutes(x.mm)	
3	**U**	**4**	**U**	**7**	**S**
3	5	4	5	7	3

35° 45.73’

3. Determine latitude indicator by determining ASCII value of byte-4
    - If greater than or equal to ‘P’, then latitude indicator is North
    - Else latitude indicator is South

			N/S		
3	**U**	**4**	**U**	**7**	**S**
3	5	4	5	7	3

‘**U**’ > ‘**P**’ ∴ North
35° 45.73’ N

4. Determine longitude indicator by determining ASCII value of byte-6
    - If greater than or equal to ‘P’, then longitude indicator is West
    - Else longitude indicator is East

					W/E
3	**U**	**4**	**U**	**7**	**S**
3	5	4	5	7	3

‘**S**’ > ‘**P**’ ∴ West

5. Determine longitude offset determining ASCII value of byte-5
    - If greater than or equal to ‘P’, then longitude offset is +100
    - Else longitude offset is +0

				Offset	
3	**U**	**4**	**U**	**7**	**S**
3	5	4	5	7	3

‘**7**’ < ‘**P**’ ∴ Longitude Offset is +0

At this point all the information that is required has been extracted and parsed from the data destination address field, the remaining data is contained in the information field. The information field contains the encoded longitude, course, speed, and altitude. The longitude is stored in the first 3-bytes of the information field and is complemented with the longitude offset and longitude indicator from the data destination address field to complete the full longitude. The information is encoded in the information address field as shown below:

	Data Type ID	Longitude			Speed and Course			Symbol Code	Symbol Table ID	Mic-E Status Text (Altitude)
		d+28	m+28	h+28	SP+28	DC+28	SE+28			
Bytes:	1	1	1	1	1	1	1	1	1	n

Figure 55 Mic-E Data Information Field

The longitude encoding is performed by parsing out each of the bytes independently and then combining them along with the longitude offset and longitude indicator to determine the current longitude. Mic-E longitude degrees are the second byte in the Information field and contain the encoded value of the longitude degrees in the range 0–179 degrees. To obtain the value in degrees, the ASCII character that is in this location is checked to see if it is in the ASCII decimal range of 38 to 127, the range corresponds to the ASCII characters '&' to 'DEL'.

Once this has been established, the actually value of degree longitude is obtained by the following steps:

1. Subtract 28 from the second byte value to obtain dd.
2. If the longitude offset is +100 degrees, add 100 to dd.
3. If $180 \leq dd \leq 189$, then subtract 80 from ddd. This places the longitude in a range between 100 and 109 degrees.
4. If $190 \leq dd \leq 199$, then subtract 190 from ddd. This places the longitude in a range between 0 and 9 degrees.
5. Finally since the valid range of characters, 38 to 127 ASCII decimal, contains the printable ASCII command DEL, ASCII decimal 127, a check must be made to

ensure that this byte is in this range. If the byte is outside this range, then the digit that was encoded was either 9, if offset is 100, or 99 if offset is zero.

The third byte is the longitude minutes and ranges from 0 to 59 minutes. This information is also encoded using the same +28 offset as the longitude degree, with one major difference. In the case of the longitude minutes the ASCII decimal range is 38-97, which corresponds to the ASCII characters '&' to 'a', all of which are printable characters. The longitude minutes (mm) are obtained by the following steps:

1. Subtract 28 from the third byte value to obtain mm.
2. If mm $\geq$ 60, then subtract 60 from mm. This places the longitude minutes in the range of 0 to 9 minutes.

The fourth byte is the longitude hundredths of minutes (hm), in the range 0.00 to 0.99 minutes. This information also is encoded using the +28 offset as the longitude degree, with this byte taking on a value in the ASCII decimal range from 28 through 127, again most all the characters in this range are printable, with the except of ASCII decimal 28 to 31 and 127. The longitude minutes hundredths are obtained by the following steps:

1. Determine if byte is in the ASCII decimal range of 32-126.
2. If in range, then subtract 28 from hm
3. Else assign hm as x.00

The speed and course of a station are encoded in bytes five to seven in a combined encoding that covers the a range in speed from 0–799 knots, and course from 0–360 degrees.

The speed and course is encoded in over three bytes in the information address field as shown below:

	Speed		**Course**
	Encoded Speed (hundreds/tens of knots)	Encoded Speed (units) and Encoded Course (hundreds of degrees)	Encoded Course (tens/units)
Bytes:	**1**	**1**	**1**

Figure 56 Mic-E Speed and Course Field

To decode the speed and course a three step process is required and can be found by utilizing the following steps:

1. To obtain the encoded speed hundreds and tens units (hhx), subtract 28 from the ASCII decimal value of the first byte in the speed and course field
2. Multiply result by 10.
3. To obtain the encoded speeds ones unit (xxh), subtract 28 from the ASCII decimal value of the second byte and divide the result by 10. The quotient is the units of speed.
4. Using results from steps 1 to 3 add values together to obtain current speed (hhh) in knots.
5. To obtain the encoded course the second and third bytes are used. The remainder from step 3 is the current course in hundreds of degrees (cxx).
6. The tens and ones units for course are obtained by subtracting 28 from the third bytes ASCII decimal value (xcc).
7. Using results from steps five and six assembly the current course as (ccc).
8. If computed speed hhh $\geq$ 800 knots, then subtract 800 from hhh. The result is a speed that can range from 000 to 799 knots.

9. If computed course ccc $\geq$ 400 degrees, then subtract 400 from ccc. This resulting course will have a range from 000 to 360 degrees. It should be noted that the course heading of zero is actually reserved as a special case where the course heading is unknown.

The last piece of information to be extracted is the system's current altitude. The altitude information is stored in the Mic-E status text. The Kenwood TH-D7 radio automatically insert a '>' as the 10th character of the information field before the encoding of the altitude in the next 4 bytes. The altitude is expressed in the form '>xxx}', where xxx is in meters. The altitude is set relative to 10km below mean sea level and is encoded using a base 91 system, and then offsetting the value by 33 to obtain the corresponding ASCII character.

For example, the systems encoded altitude at 56,000 feet is:

56,000 feet = 17,069 meters altitude

17,069 meters + 10km mean sea level offset = 27,069 meters

27,069 / 91^2 = 3, remainder 2,226. First digit is now ASCII decimal 36 (+33 offset).

2,226 / 91 = 24, remainder 42. Second digit and third digits now become ASCII decimal 57 and 75, respectively (+33 offset).

Converting the three digits, (36, 57, 75) to their ASCII codes, we obtain **$9K}**. To decode the altitude we reverse the process as is shown in the following pseudo code:

1. Determine location of either '}' or '>' in the Mic-E status text field. The three bytes between them are the encoded altitude in meters relative to 10km below mean sea level.

2. Subtract 33 from the ASCII decimal value of the three bytes in the altitude string. (Alt1, Alt2, and Alt3)

3. Multiply Alt1 by 91^2

4. Multiply Alt2 by 91
5. Determine altitude in meters by adding Alt1 + Alt2 + Alt3
6. Finally offset altitude by 10km to obtain current height of system in meters

The following is a sample packet sent during the ASTRO-14 mission, if we follow the decoding schemes laid out above we can decode the message as follows:

Raw Mic-E Packet	W2OSU-11>3U4U7S,WIDE3:`~0cm HO/>$7<}
Source Field	**W2OSU-11>**
Data Destination Address field	**3U4U7S,WIDE3:**
Data Information Field	**`~0cm HO/>$7<}**

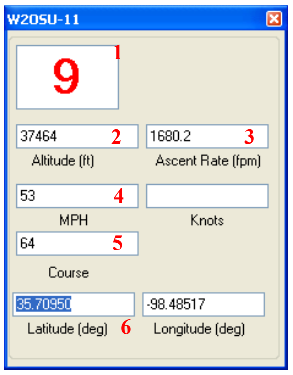
	Lat Digit 1 + Message Bit A	***Lat Digit 1 + Message Bit A***	***Lat Digit 3 + Message Bit C***	***Lat Digit 4 + N/S Lat Indicator***	***Lat Digit 5 + Longitude Offset***	***Lat Digit 6 + E/W Long Indicator***	***APRS Digi Path Code***
Bytes:	**3**	**U**	**4**	**U**	**7**	**S**	**,**
Decoded	35° 45.73' N						
	35.7622°			N	+0	W	

	Data Type ID	**Longitude**			**Speed and Course**			**Symbol Code**	**Symbol Table ID**	**Mic-E Status Text (Altitude)**
		d+28	m+28	h+28	SP+28	DC+28	SE+28			
Byte:	**`**	**~**	**0**	**c**	**m**	**<SP>**	**H**	**O**	**/**	**/>$7<}**
Decoded		98° 20.71' W			10 knots					16,872 m
		-98.3452°			44°					55,354 ft

Figure 57 Example Mic-E Decoding from ASTRO-14

ASTRO Tracker

Once the information has been parsed, it is used by the tracking program to display the system's current latitude, longitude, altitude, ascent/descent velocity, ground speed, and ground heading. The system's current status is displayed in several locations: a small tracking window, the real-time tracking module, and overlaid on a map. Examples of all three windows are shown below in Figure 58 to Figure 60.


W2OSU-11

1 9

Altitude (ft): 37464 2

Ascent Rate (fpm): 1680.2 3

MPH: 53 4

Knots:

Course: 64 5

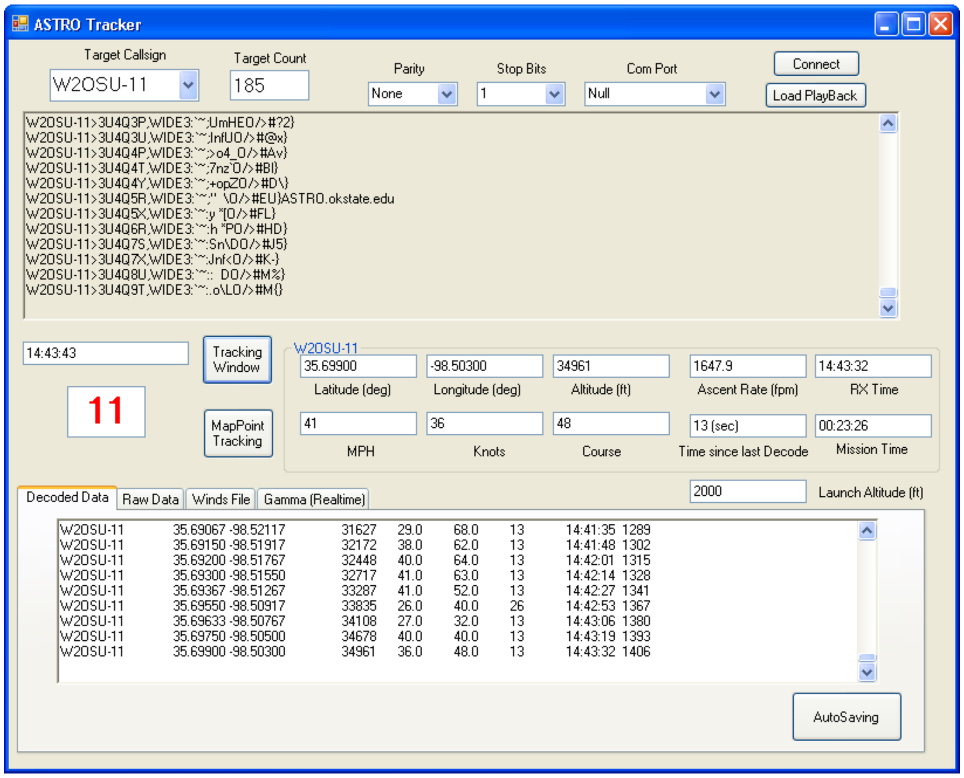
Latitude (deg): 35.70950

Longitude (deg): -98.48517

6

1. Time since last decode
2. Altitude at last decode
3. Ascent Rate at last decode
4. Ground Speed at last decode
5. Ground Heading at last decode
6. Latitude and Longitude at last decode

Figure 58 Quick Status Tracking Window



Target Callsign: W2OSU-11

Target Count: 185

Parity: None

Stop Bits: 1

Com Port: Null

```
W2OSU-11>3U4Q3P,WIDE3:`~;UmHEO/>#?2}
W2OSU-11>3U4Q3U,WIDE3:`~;InfUO/>#@x}
W2OSU-11>3U4Q4P,WIDE3:`~;>o4_O/>#Av}
W2OSU-11>3U4Q4T,WIDE3:`~;7nz`O/>#BI}
W2OSU-11>3U4Q4Y,WIDE3:`~;+opZO/>#D\}
W2OSU-11>3U4Q5R,WIDE3:`~;"  \O/>#EU}ASTRO.okstate.edu
W2OSU-11>3U4Q5X,WIDE3:`~:y *[O/>#FL}
W2OSU-11>3U4Q6R,WIDE3:`~:h *PO/>#HD}
W2OSU-11>3U4Q7S,WIDE3:`~:Sn\DO/>#J5}
W2OSU-11>3U4Q7X,WIDE3:`~:Jnf<O/>#K-}
W2OSU-11>3U4Q8U,WIDE3:`~::  DO/>#M%}
W2OSU-11>3U4Q9T,WIDE3:`~:.o\LO/>#M{}
```

14:43:43

11

W2OSU-11

Latitude (deg): 35.69900

Longitude (deg): -98.50300

Altitude (ft): 34961

Ascent Rate (fpm): 1647.9

RX Time: 14:43:32

MPH: 41

Knots: 36

Course: 48

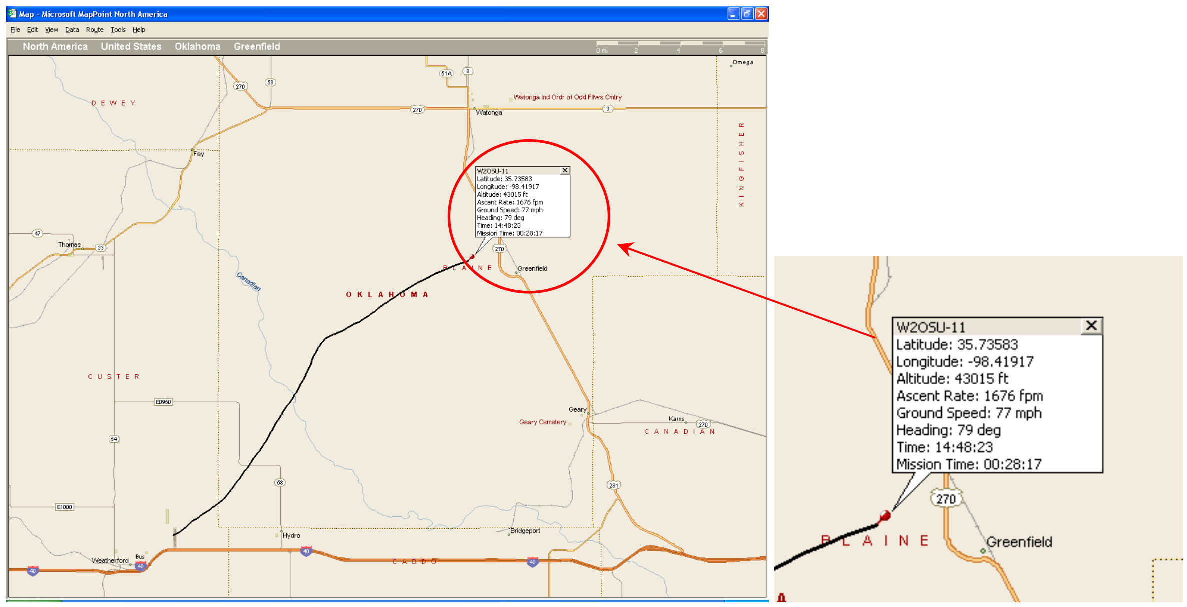
Time since last Decode: 13 (sec)

Mission Time: 00:23:26

Launch Altitude (ft): 2000

```
W2OSU-11        35.69067 -98.52117      31627   29.0    68.0    13      14:41:35  1289
W2OSU-11        35.69150 -98.51917      32172   38.0    62.0    13      14:41:48  1302
W2OSU-11        35.69200 -98.51767      32448   40.0    64.0    13      14:42:01  1315
W2OSU-11        35.69300 -98.51550      32717   41.0    63.0    13      14:42:14  1328
W2OSU-11        35.69367 -98.51267      33287   41.0    52.0    13      14:42:27  1341
W2OSU-11        35.69550 -98.50917      33835   26.0    40.0    26      14:42:53  1367
W2OSU-11        35.69633 -98.50767      34108   27.0    32.0    13      14:43:06  1380
W2OSU-11        35.69750 -98.50500      34678   40.0    40.0    13      14:43:19  1393
W2OSU-11        35.69900 -98.50300      34961   36.0    48.0    13      14:43:32  1406
```

Figure 59 Real-Time Tracking Display Screens


W2OSU-11
Latitude: 35.73583
Longitude: -98.41917
Altitude: 43015 ft
Ascent Rate: 1676 fpm
Ground Speed: 77 mph
Heading: 79 deg
Time: 14:48:23
Mission Time: 00:28:17

Map - Microsoft MapPoint North America
File Edit View Data Route Tools Help
North America United States Oklahoma Greenfield
DEWEY
CUSTER
OKLAHOMA
BLAINE
KINGFISHER
CANADIAN
CADDO
Watonga
Greenfield
Geary
Weatherford
Hydro
Bridgeport
Thomas
Fay
270

Figure 60 Real-Time Overlay on MapPoint

As discussed earlier, one of the major problems with the Mic-E format of the APRS packet is the lack of a timestamp at the time of transmission. In order to overcome this problem the tracking program adds a timestamp to the packages once they are decoded. One other problem that is faced is that there is no guaranty that the packets will arrive and be decoded successfully at even time intervals. In order to resolve the problem of uneven time intervals, we decided to use the method of Least Squares to produce a linear curve fit over a small moving flight window. The results of the Least Squares are the current ascent or descent velocities. The program determines which phase of the mission the system is currently in by monitoring the time history of the past altitudes of the system. In order to allow the real-time predictor to work, the tracking program creates a file that contains four pieces of information: the current ascent/descent rate, the current altitude, the mission time, and a status flag as to which phase of the mission has been detected. A sample history of this data file is shown below in Figure 61.

Ascent Rate, fps	Altitude, ft	Mission Time, sec	Phase Flag
33.2102	98999	3233	FALSE
32.218	99288	3246	FALSE
31.1327	99764	3272	FALSE
19.0194	95830	3298	FALSE
-191.8866	70653	3416	TRUE
-191.8866	67343	3429	TRUE
-176.9738	64154	3456	TRUE
-170.614	61424	3469	TRUE
-163.0812	60112	3482	TRUE
-159.5972	58852	3495	TRUE
-154.2805	57608	3508	TRUE

Figure 61 Sample of Mission History Status Provided by Tracking

This data shown above is written in two files, one file is only the most current mission status, and the other file is an appended mission history. There is approximately a two minutes gap between the data when the phase flag switches from FALSE to TRUE. This is caused by the tracking program detecting the transition from the ascent to the descent phase. This action resets the time history to zero and then tracker must wait for at least ten new packets to arrive before calculations of the first descent phase velocity can be made. After the initial ten calculations have been made, the system will update with a new velocity upon the arrival of a new packet.

Real-Time Winds Aloft

One of the other major parameters that must be updated to help improve real-time predictions is the winds aloft. Since the winds used during the planning phase are predictions for the flight area, a need to update the winds information with the latest winds are required to help improve the real-time predictions of the system. Since the system is all ready transmitting its current location and status, this information can be used to improve the predicted winds with current wind conditions. In order to perform this task, the real-time tracker uses the current information on ground speed and heading to determine the current wind conditions at a given altitude. During the ascent phase, the only lateral force acting on the balloon and system is drag from the wind. We assume that the balloon and system react quickly to this force, and therefore does not require a lateral acceleration calculation (F=ma). Since this is the case, the tracking program utilizes the system's current ground speed and ground heading to assigns the wind conditions at the altitude during decoding. The ground heading is actually 180° out of phase with wind bearing so this must be corrected before the wind bearing is stored. This information is then stored in an appended file that the real-time prediction program uses to track the most current wind data.

Altitude, ft	Wind Bearing, deg	Wind Speed, mph
89970	89	33
90650	94	46
91289	124	42
91634	129	34
91985	115	37
92356	120	37
93091	132	24

Figure 62 Sample of Winds Aloft Provided by Tracking

Mission Predictor

The main goal of this project has been to understand the dynamics behind a high altitude balloon mission. So far we have discussed the parameters that effect both the ascent and descent phases of the mission, including how to correct the mission parameters to improve a prediction in the landing zone. A program has been written that utilizes all the skills developed for this project. The mission development program is composed of two phases: mission planning and real-time mission predictions. The best way to explain this program is to step through how this program can be used during each phase and then to show the results of the real-time mission predictions on several real missions.

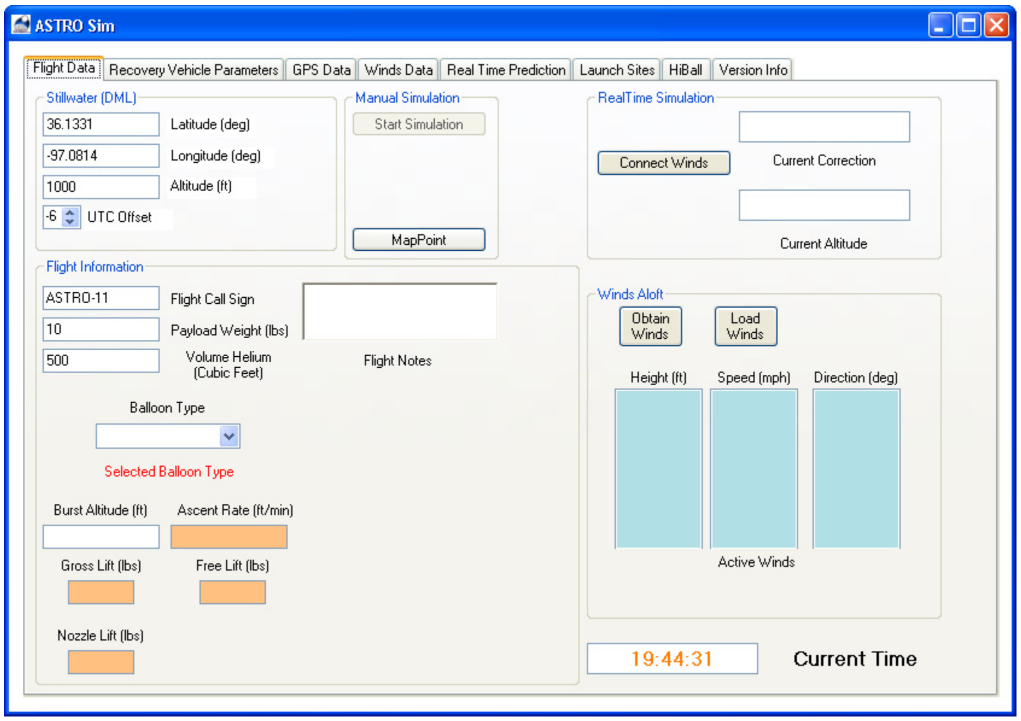
Preflight Predictions

The goal of the mission planner is to aid in the long term planning for a near space mission. In order to prepare for a mission several things must be known or assumed upfront.

- Type of Balloon to be flown
- Size of parachute to be flown
- Approximate payload weight to be flown
- Launch Site and Time

Once this information is know you can start the mission planner to obtain a prediction for a landing zone with the given conditions. To help visualize the landing zone as well as the flight path, the program has been designed to interface with Microsoft MapPoint 2006. With use of the mission planner and MapPoint, a user can quickly change flight parameters to see the effects on the predicted landing zone. The following are the steps to getting the program setup and making predictions.

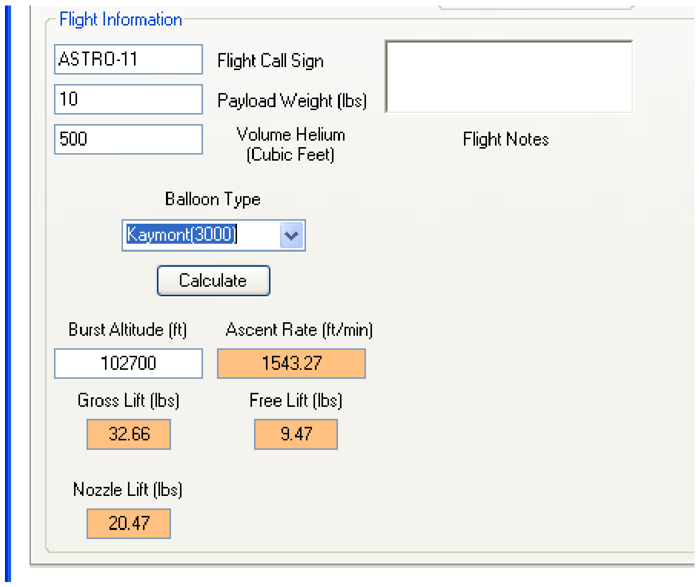
The first step is to start the ASTRO Flight Prediction and, if map overviews are desired, also start MapPoint. While MapPoint is very useful in helping to visualize the overall mission, it is by no means required as all the major flight locations will be readily available within the flight prediction program.



Latitude (deg): 36.1331
Longitude (deg): -97.0814
Altitude (ft): 1000
UTC Offset: -6
Flight Call Sign: ASTRO-11
Payload Weight (lbs): 10
Volume Helium (Cubic Feet): 500
Current Time: 19:44:31

Figure 63 Flight Predictions Flight Data Screen

Once you've opened the program there are several parameters that you will have to set before you can run a flight prediction. The first parameter that must be set is the type of balloon. If you would like to see the effects of a different balloon on the flight, you can change the balloon type to a new selection and then repeat the predictions. For this mission we are going to select a Kaymont 3000g balloon. Once the type of balloon has been selected, you can enter in estimation for your payload weight and the amount of helium that you plan to use. After entering in these parameters you can then click on *Calculate* and the program will return several important parameters.



Flight Information

Flight Call Sign	ASTRO-11
Payload Weight (lbs)	10
Volume Helium (Cubic Feet)	500
Balloon Type	Kaymont(3000)
Burst Altitude (ft)	102700
Ascent Rate (ft/min)	1543.27
Gross Lift (lbs)	32.66
Free Lift (lbs)	9.47
Nozzle Lift (lbs)	20.47
Flight Notes	

Figure 64 Flight Predictions Flight Information

The parameters that are returned in the Flight Information area are: burst altitude, ascent rate, gross lift, nozzle lift, and free lift. Burst altitude is determined from the amount of Helium placed in the balloon and the maximum diameter of the balloon provided by the manufacture. In order to determine the altitude where the balloon should burst the ideal gas expansion of the balloon, shown below, is iteratively solved until the burst altitude is found to within 300 feet.

$$\boldsymbol{\forall_{alt} = \forall_{gnd}\left(\frac{P_{gnd}}{P_{alt}}\right)\left(\frac{T_{alt}}{T_{gnd}}\right)} \quad \textbf{[67]}$$

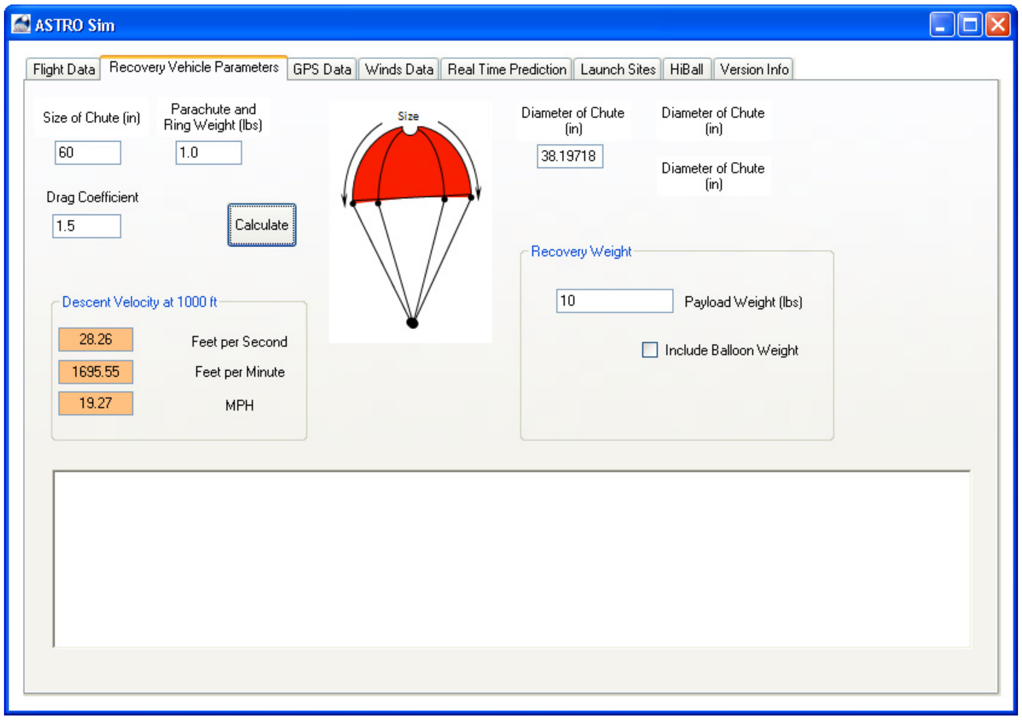
The ascent rate is found by solving the ascent rate model for the first 120 seconds of flight and as such will only give you an indication to the expected ascent rate but it should not be assumed that this will be the ascent rate during the entire flight.

Gross lift is determined by the amount of helium that is placed in the balloon and is governed directly by:

$$\boldsymbol{B = \forall(\rho_{air} - \rho_{He})g} \quad \textbf{[68]}$$

Nozzle lift is found by removing the weight of the balloon from gross lift. The nozzle lift can be directly measured before launch as the force being exerted at the nozzle of the balloon before the addition of payload. Free lift is then found by removing the weight of the payload from nozzle lift and it is this free lift that provides the driving force for the balloon and system to ascend, before you are allowed to run a prediction the free lift must be greater than zero. This is one area that you can improve your preflight predictions in by knowing the amount of payload you are flying with and monitoring either the amount of helium you are placing in the balloon or the amount of nozzle lift.

The next set of parameters that can be changed is under the *Recovery Vehicle Parameters* tab. Once selected this window will allow you to change to the type of spherical chute to be used in the mission, for this mission, we are going to select a 60 inch chute that has a chute plus chute ring weight of one pound. You should note that the typical size of spherical chutes is actually not the diameter of the chute but is instead is the span across the top of a deployed chute. This program has been designed to enter the size of the chute with respect to its half circumference and not diameter. However, since some chutes are rated by their diameter one can correct the size of the chute for the diameter given. Once the size and weight of the chute have been entered, you can click on the *Calculate* to obtain the velocity of the system if it was dropped from 2000 feet above the deck.



Size of Chute (in): 60

Parachute and Ring Weight (lbs): 1.0

Drag Coefficient: 1.5

Diameter of Chute (in): 38.19718

Diameter of Chute (in)

Diameter of Chute (in)

Descent Velocity at 1000 ft

28.26	Feet per Second
1695.55	Feet per Minute
19.27	MPH

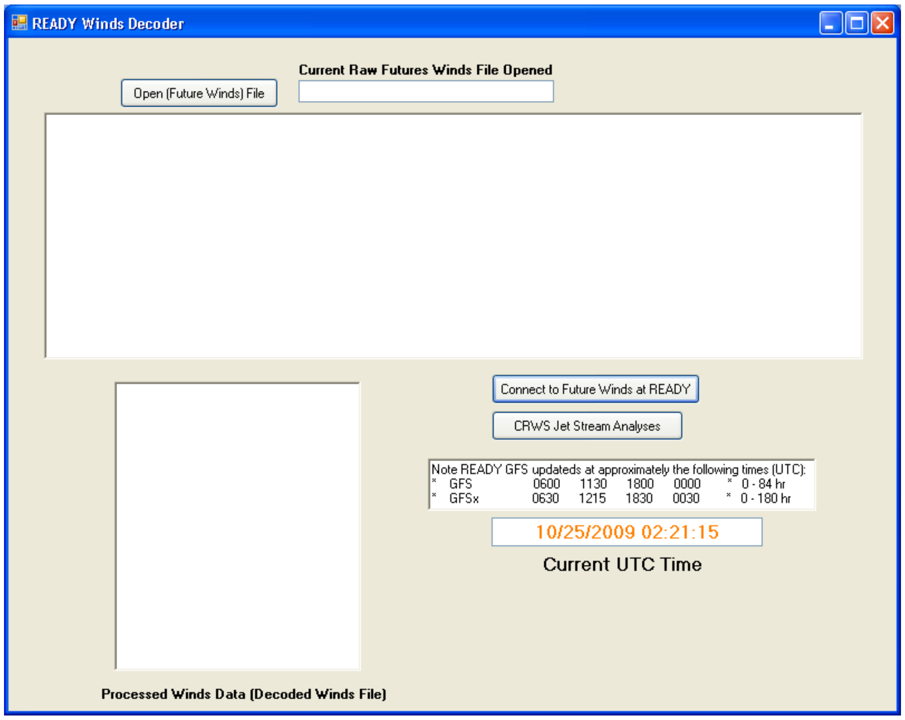
Recovery Weight

10 Payload Weight (lbs)

☐ Include Balloon Weight

Figure 65 Flight Predictions Recovery Vehicle Parameters Screen

After you are satisfied with the flight parameters you will need to obtain, or load, winds aloft predictions for the time and date of the flight. To obtain predictions of the winds aloft you will need to click the *Obtain Winds* button, as seen in Figure 63. Once this has been clicked a new screen, shown in Figure 66, will appear. From here there are several options. The most common option will be to select *Connect to Future Winds at READY*. This will take you to NOAA's Air Resources Laboratory (ARL) Real-time Environmental Applications and Display sYstem (READY).


READY Winds Decoder
Current Raw Futures Winds File Opened
Open (Future Winds) File
Connect to Future Winds at READY
CRWS Jet Stream Analyses
Current UTC Time
Processed Winds Data (Decoded Winds File)

```
Note READY GFS updateds at approximately the following times (UTC):
*   GFS        0600   1130   1800   0000    *  0 - 84 hr
*   GFSx       0630   1215   1830   0030    *  0 - 180 hr
```

10/25/2009 02:21:15

Figure 66 Flight Predictions READY Winds Decoder

The READY wind aloft information is available from the Global Forecast System (GFS) for the entire world in 1 degree resolution and allows predictions of the winds from the surface to an altitude of approximately 26km. To obtain the winds for the flight area, you will have two choices: Enter the International Civil Aviation Organization code for the newest airport or enter the latitude and longitude of your launch site. A list of ICAO sites (6) in Oklahoma is shown in Figure 67. After entering this information, the site will take you to a new screen where you will need to select one of three GFS models. The Global Forecasting System or GFS contains forecast data every 3 hours from 0 to 84 hours from the time the forecast is made, GFSx contains forecast data every 6 hours from forecast hours 0 to 180, and the GFSlr contains forecast data every 12 hours from forecast hours from 192 to 384 hours from the time the forecast is made.

ID	*Description*	*Lat*	*Lon*	*Elevation (meters)*
1F0	ARDMORE DOWNTOWN EXEC ARPT	34.17	-97.12	257
ADH	ADA MUNICIPAL AIRPORT	34.81	-96.67	310
ADM	ARDMORE MUNICIPAL	34.30	-97.02	232
AQR	ATOKA MUNI AIRPORT	34.40	-96.15	180
AVK	ALVA RGNL AIRPORT	36.77	-98.67	449
BVO	BARTLESVILLE/PHILLI (ASOS)	36.77	-96.03	217
CHK	CHICKASHA MUNI AIRPORT	35.10	-97.97	351
CQB	CHANDLER MUNI AIRPORT	35.72	-96.82	300
CSM	CLINTON-SHERMAN (ASOS)	35.33	-99.20	586
DUA	DURANT	33.99	-96.39	213
DUC	HALLIBURTON FIELD AIRPORT/DUNCAN	34.47	-97.96	339
END	VANCE AFB/ENID	36.33	-97.92	398
FDR	FREDERICK MUNICIPAL (ASOS)	34.35	-98.98	382
FSI	FORT SILL	34.65	-98.40	362
GAG	GAGE/SHATTUCK (ASOS)	36.30	-99.77	678
GCM	CLAREMORE RGNL AIRPORT	36.29	-95.48	221
GMJ	GROVE MUNI AIRPORT	36.61	-94.74	254
GOK	GUTHRIE MUNICIPAL (ASOS)	35.85	-97.42	327
GUY	GUYMON (ASOS)	36.68	-101.50	952
HBR	HOBART MUNICIPAL (ASOS)	34.98	-99.05	477
JSV	SALLISAW MUNI AIRPORT	35.44	-94.80	161
JWG	WATONGA	35.87	-98.42	472
LAW	LAWTON MUNICIPAL (ASOS)	34.57	-98.42	338
LTS	ALTUS AFB	34.67	-99.27	420
MKO	MUSKOGEE/DAVIS FLD (ASOS)	35.65	-95.37	186
MLC	MC ALESTER REGIONAL (ASOS)	34.88	-95.78	235
OKC	OKLAHOMA CITY (ASOS)	35.40	-97.60	397
OKM	OKMULGEE MUNI AIRPORT	35.67	-95.95	220
OUN	NORMAN	35.22	-97.47	362
PNC	PONCA CITY MUNI	36.73	-97.10	307
PVJ	PAULS VALLEY MUNI AIRPORT	34.71	-97.22	295
PWA	OKLAHOMA CITY/WILEY (ASOS)	35.53	-97.65	396
RKR	POTEAU/ROBERT S KERR AIRPORT	35.02	-94.62	138
RVS	TULSA/LLOYD JONES (ASOS)	36.03	-95.98	193
SNL	SHAWNEE MUNI AIRPORT	35.36	-96.94	327
SWO	STILLWATER MUNI (ASOS)	36.17	-97.08	300
TIK	TINKER AFB	35.42	-97.38	394
TUL	TULSA INTL ARPT (ASOS)	36.20	-95.90	206
WDG	ENID/WOODRING MUNI	36.38	-97.78	356
WWR	WEST WOODWARD	36.43	-99.53	667

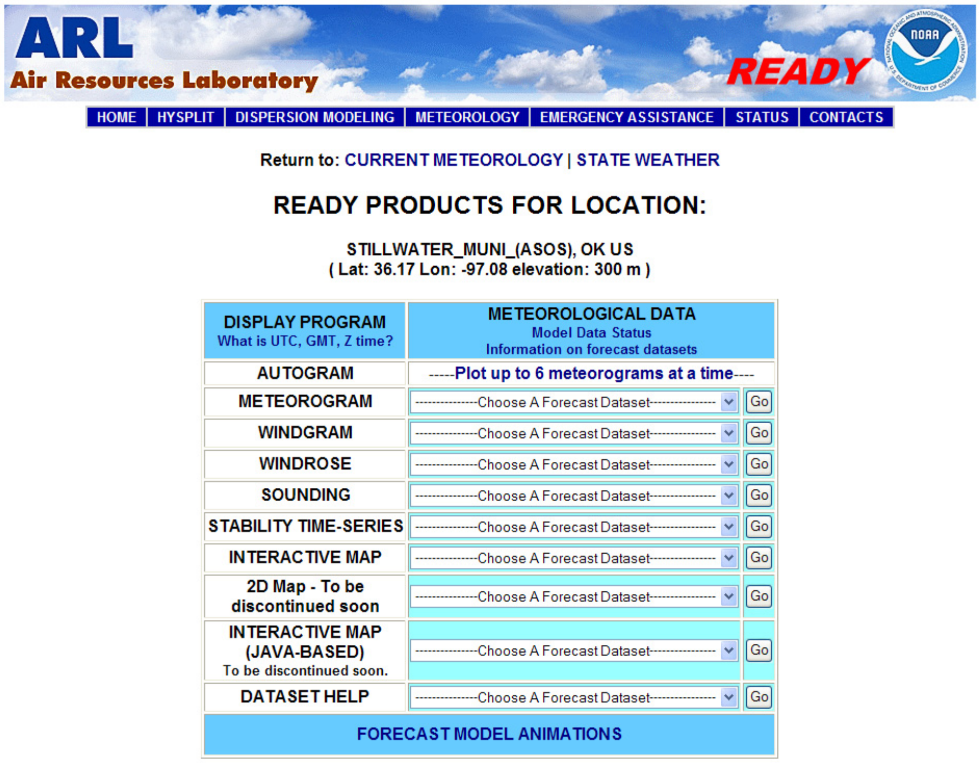
Figure 67 READY Sites in Oklahoma

For the most reliable data forecast, use the GFS model. The READY site does not update the models on a continuous basis; instead the models are updated following the rough schedule below.

Model	Update Time UTC				Time Frame, Windows
GFS	0600	1130	1800	0000	(0-84h, 3hr)
GFSx	0630	1215	1830	0030	(0-180h, 6hr)
GFSlr	0600	1130	1800	0000	(192-384h, 12hr)

Figure 68 GFS Model Update Time Schedule

Once the model has been selected, you will then be asked for the type of output in which you would like to receive the data. For the use of this program, you will need to select *Text Only* and then follow any on-screen directions for typing in the access code to obtain the data. Now you should see a screen that contains the winds prediction for the site and correct date and time. If the data set is correct, you will now need to save the page in text format.


ARL
Air Resources Laboratory
READY
NOAA
HOME | HYSPLIT | DISPERSION MODELING | METEOROLOGY | EMERGENCY ASSISTANCE | STATUS | CONTACTS

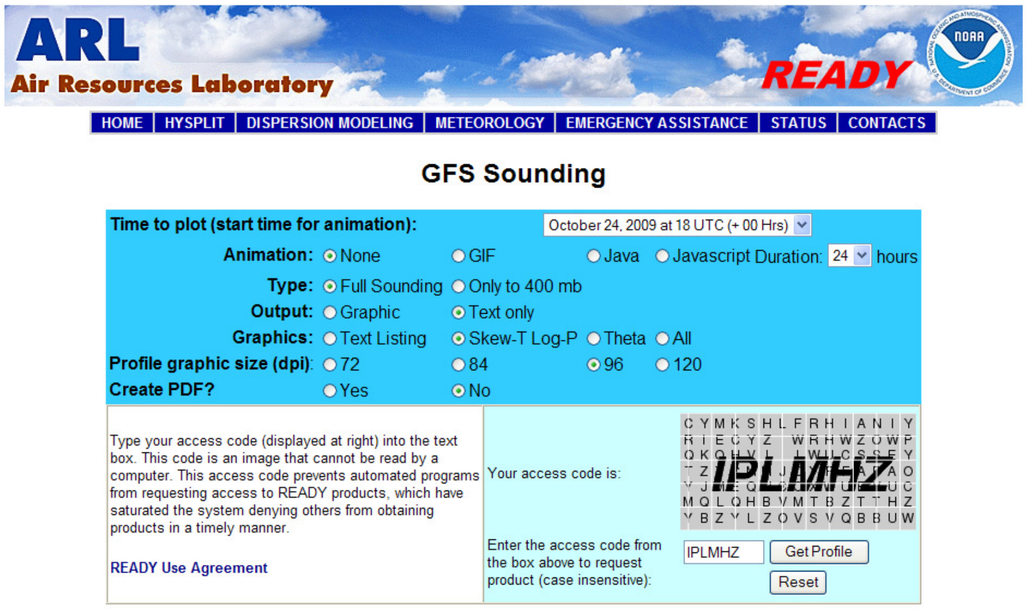
Return to: CURRENT METEOROLOGY | STATE WEATHER

READY PRODUCTS FOR LOCATION:

STILLWATER_MUNI_(ASOS), OK US
(Lat: 36.17 Lon: -97.08 elevation: 300 m)

DISPLAY PROGRAM What is UTC, GMT, Z time?	METEOROLOGICAL DATA Model Data Status Information on forecast datasets	
AUTOGRAM	-----Plot up to 6 meteorograms at a time----	
METEOROGRAM	---------------Choose A Forecast Dataset---------------	Go
WINDGRAM	---------------Choose A Forecast Dataset---------------	Go
WINDROSE	---------------Choose A Forecast Dataset---------------	Go
SOUNDING	---------------Choose A Forecast Dataset---------------	Go
STABILITY TIME-SERIES	---------------Choose A Forecast Dataset---------------	Go
INTERACTIVE MAP	---------------Choose A Forecast Dataset---------------	Go
2D Map - To be discontinued soon	---------------Choose A Forecast Dataset---------------	Go
INTERACTIVE MAP (JAVA-BASED) To be discontinued soon.	---------------Choose A Forecast Dataset---------------	Go
DATASET HELP	---------------Choose A Forecast Dataset---------------	Go
FORECAST MODEL ANIMATIONS		

Figure 69 READY Forecast Models


ARL
Air Resources Laboratory
READY
HOME | HYSPLIT | DISPERSION MODELING | METEOROLOGY | EMERGENCY ASSISTANCE | STATUS | CONTACTS
Get Profile
Reset

GFS Sounding

Time to plot (start time for animation):	October 24, 2009 at 18 UTC (+ 00 Hrs)			
Animation:	[✓ None]	GIF	Java	Javascript Duration: 24 hours
Type:	[✓ Full Sounding]	Only to 400 mb		
Output:	Graphic	[✓ Text only]		
Graphics:	Text Listing	[✓ Skew-T Log-P]	Theta	All
Profile graphic size (dpi):	72	84	[✓ 96]	120
Create PDF?	Yes	[✓ No]		

Type your access code (displayed at right) into the text box. This code is an image that cannot be read by a computer. This access code prevents automated programs from requesting access to READY products, which have saturated the system denying others from obtaining products in a timely manner.

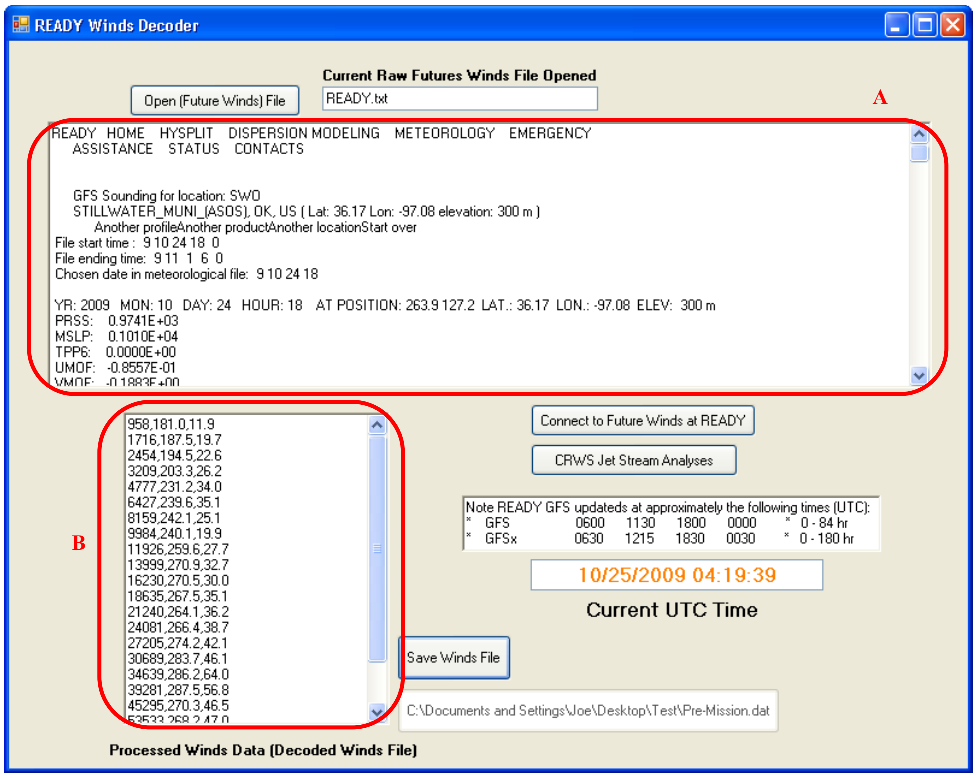
READY Use Agreement

Your access code is: IPLMHZ

Enter the access code from the box above to request product (case insensitive): IPLMHZ

Figure 70 READY GFS Sounding Data

Now you are ready to process the winds data so that the flight prediction program can access it. To do this return to the READY Winds Decoder screen, Figure 66, and then load the READY winds by *Open (Future Winds) File*. After opening the file, you will see several screens fill in as shown Figure 71. The top screen (A) contains a copy of the file you just opened. This is the 'Raw' READY file. The smaller screen (B), in the lower left, is the data after being processed. You can now save the winds aloft file for use. Click *Save Winds File* and give it a name that you can recall later. If you are planning missions around a given date you can repeat the READY winds decoding as many times as needed. Since the presence of the Jet Stream can heavily influence the range of your mission a direct link to Jet Stream Analyses can be accessed by clicking *CRWS Jet Stream Analyses*.


READY Winds Decoder
Current Raw Futures Winds File Opened
Open (Future Winds) File
A
B
Connect to Future Winds at READY
CRWS Jet Stream Analyses
Current UTC Time
Save Winds File
Processed Winds Data (Decoded Winds File)

```
READY.txt
```

```
READY   HOME   HYSPLIT   DISPERSION MODELING   METEOROLOGY   EMERGENCY
    ASSISTANCE   STATUS   CONTACTS


    GFS Sounding for location: SWO
    STILLWATER_MUNI_(ASOS), OK, US ( Lat: 36.17 Lon: -97.08 elevation: 300 m )
        Another profileAnother productAnother locationStart over
File start time :  9 10 24 18  0
File ending time:  9 11  1  6  0
Chosen date in meteorological file:  9 10 24 18

YR: 2009   MON: 10   DAY: 24   HOUR: 18   AT POSITION: 263.9 127.2  LAT.: 36.17  LON.: -97.08  ELEV:  300 m
PRSS:    0.9741E+03
MSLP:    0.1010E+04
TPP6:    0.0000E+00
UMOF:   -0.8557E-01
VMOF:   -0.1883E+00
```

```
958,181.0,11.9
1716,187.5,19.7
2454,194.5,22.6
3209,203.3,26.2
4777,231.2,34.0
6427,239.6,35.1
8159,242.1,25.1
9984,240.1,19.9
11926,259.6,27.7
13999,270.9,32.7
16230,270.5,30.0
18635,267.5,35.1
21240,264.1,36.2
24081,266.4,38.7
27205,274.2,42.1
30689,283.7,46.1
34639,286.2,64.0
39281,287.5,56.8
45295,270.3,46.5
53533,268.2,47.0
```

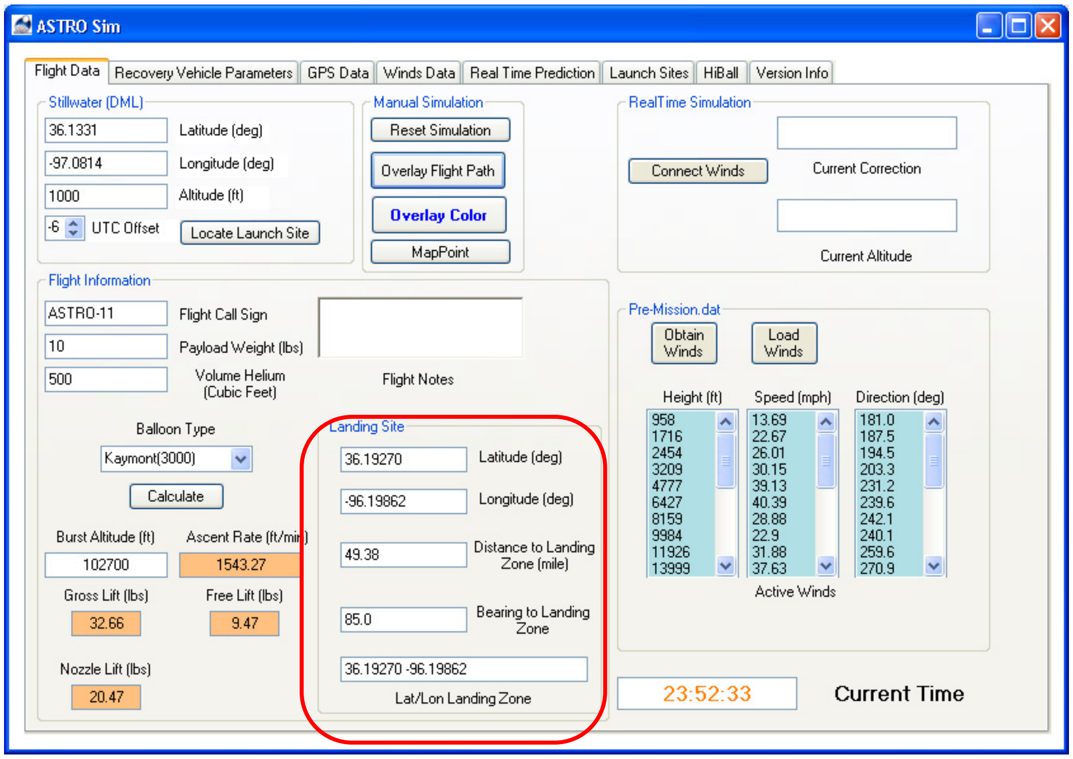
```
Note READY GFS updateds at approximately the following times (UTC):
*   GFS          0600    1130    1800    0000     *  0 - 84 hr
*   GFSx         0630    1215    1830    0030     *  0 - 180 hr
```

```
10/25/2009 04:19:39
```

```
C:\Documents and Settings\Joe\Desktop\Test\Pre-Mission.dat
```

Figure 71 Flight Predictions READY Winds Decoder with Processed Winds

Now you will load the winds data into the Prediction program by clicking on *Load Winds* on the Flight Data tab in flight predictions, see Figure 63. After the winds have been loaded and you are satisfied with the flight parameters, a flight prediction can be made. Starting the predication is done by clicking *Start Simulation*. The flight prediction is calculated using the ascent model discussed above until the system reaches the maximum altitude defined by either burst or cut down. After burst, the model switches to the descent mode and continues to solve until the system has reached 1000 ft AGL. Once the simulation is complete, the predicted landing zone is displayed in the Landing Site information area, as seen in Figure 72. If MapPoint has been linked, the predicated flight path can be displayed as is shown in Figure 73.



Stillwater (DML)

36.1331	Latitude (deg)
-97.0814	Longitude (deg)
1000	Altitude (ft)
-6	UTC Offset

Manual Simulation

RealTime Simulation

Current Correction

Current Altitude

Flight Information

ASTRO-11	Flight Call Sign
10	Payload Weight (lbs)
500	Volume Helium (Cubic Feet)

Flight Notes

Balloon Type: Kaymont(3000)

Burst Altitude (ft)	Ascent Rate (ft/min)
102700	1543.27

Gross Lift (lbs)	Free Lift (lbs)
32.66	9.47

Nozzle Lift (lbs): 20.47

Landing Site

36.19270	Latitude (deg)
-96.19862	Longitude (deg)
49.38	Distance to Landing Zone (mile)
85.0	Bearing to Landing Zone
36.19270 -96.19862	Lat/Lon Landing Zone

Pre-Mission.dat

Height (ft)	Speed (mph)	Direction (deg)
958	13.69	181.0
1716	22.67	187.5
2454	26.01	194.5
3209	30.15	203.3
4777	39.13	231.2
6427	40.39	239.6
8159	28.88	242.1
9984	22.9	240.1
11926	31.88	259.6
13999	37.63	270.9

Active Winds

23:52:33 Current Time

Figure 72 Flight Predictions Flight Data with Landing Site

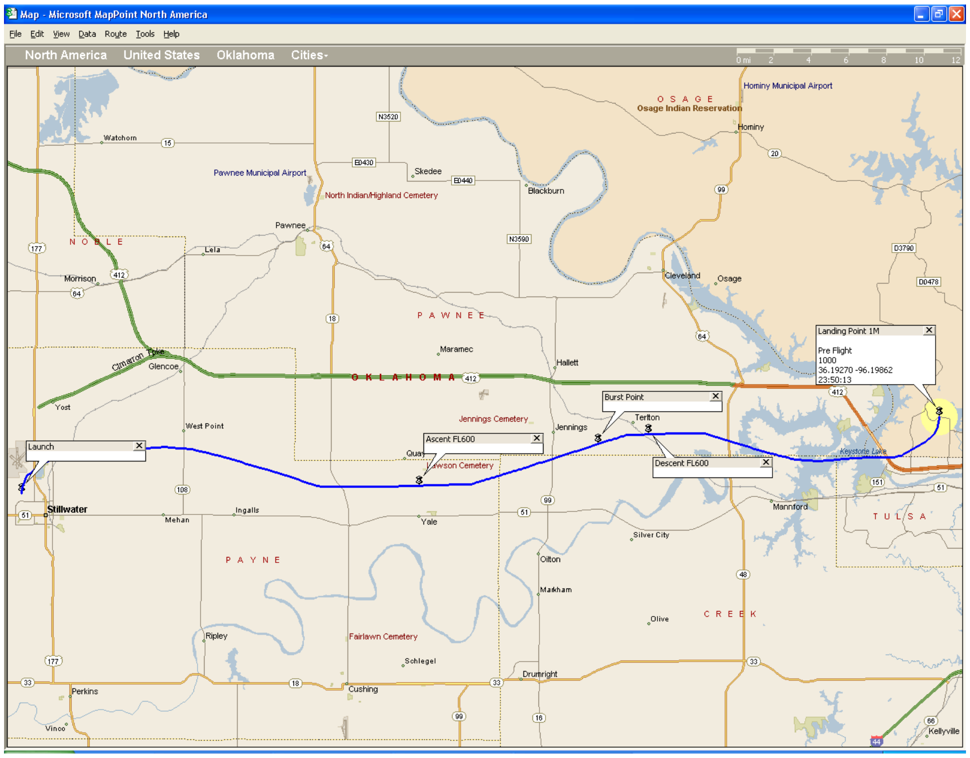


Figure 73 Predicted Flight Path overlaid on MapPoint

Real-Time Mission Predictor

One of the goals of this project was to improve not only the ability to predict the landing zone before the start of the flight, but more importantly to improve the prediction of the landing zone in real-time during a mission. The following is a list of items that would occur in a perfect mission:

Ascent Phase

- Weight of the payload is known.
- Fill volume, size, and weight of the balloon are known.
- Drag coefficients of the system follow the drag model that has been established.
- Encounter wind profile matches the READY winds predictions
- Atmosphere follows the assumed model
- Balloon and system transition to descent phase as planned

Descent Phase

- Weight of the payload is known.
- Size and effectiveness of the chute is known and remains constant.
- Drag coefficients of the system follow the drag model that has been established
- Encounted wind profile matches the READY winds predictions
- Atmosphere follows the assumed model

If all the above assumptions worked out perfectly ever time, then the recovery phase would literally be: Drive to the predicting landing zone and recover payloads. In fact if things performed this perfect, then there would be no need for tracking gear at all. However in reality most of these assumptions are not going to be perfect or in some cases even consistent throughout the mission. In order to improve the predictions on the landing zone, we must update the performance parameters during the mission using the behavior of the real system. In order to accomplish this goal the tracking program and the flight predictions program were designed to interact with each to form a real-time mission predictor.

The real-time mission predictor (RTMP) uses the same basic algorithm as the mission planner with several additions. The additions to the interface allow the tracker to pass several parameters about the ongoing mission into the program to then be utilized to increase the predictive accuracy of the landing zone. The main parameters that are passed to the RTMP are the system's current ascent/decent speed, latitude, longitude, altitude, mission time, mission phase, and updated winds aloft.

In order to test the program, missions ASTRO-11 to ASTRO-14, which were originally tracked using the real-time tracker (RTT) and were played back in real-time using the RTT built in playback feature. This playback feature allows the RTT to behave as if it was receiving data from the system and then process it in real-time. This feature allows the RTT to be used to train tracking crews, as well as testing out new prediction schemes without the need to physically launch a mission. The only requirement to produce a playback file is a record of the original raw packets and their timestamps. If the ASTRO RTT was used to track the original flight, then a playback file was automatically produced under the settings found in the *Raw Data* tab as seen in Figure 74.

W2OSU-11>3U4R7Y,WIDE3:`~8c!zHO/>#Y2}ASTRO.okstate.edu	10:35:40 AM
W2OSU-11>3U4S0S,WIDE3:`~84!pUO/>#Zu}	10:36:06 AM
W2OSU-11>3U4S1V,WIDE3:`~7z">XO/>#\w}	10:36:19 AM
W2OSU-11>3U4S2Y,WIDE3:`~7`"HRO/>#]o}	10:36:32 AM
W2OSU-11>3U4S2Y,WIDE3:`~7`"HRO/>#]o}	10:36:32 AM
W2OSU-11>3U4S3V,WIDE3:`~7J"pXO/>#^g}	10:36:45 AM
W2OSU-11>3U4S5R,WIDE3:`~7'#pQO/>#`W}	10:36:58 AM
W2OSU-11>3U4S6Q,WIDE3:`~6u#HeO/>#aO}	10:37:11 AM
W2OSU-11>3U4S7R,WIDE3:`~6Y"*]O/>#bI}ASTRO.okstate.edu	10:37:24 AM

Figure 74 ASTRO Real-Time Tracking Playback File

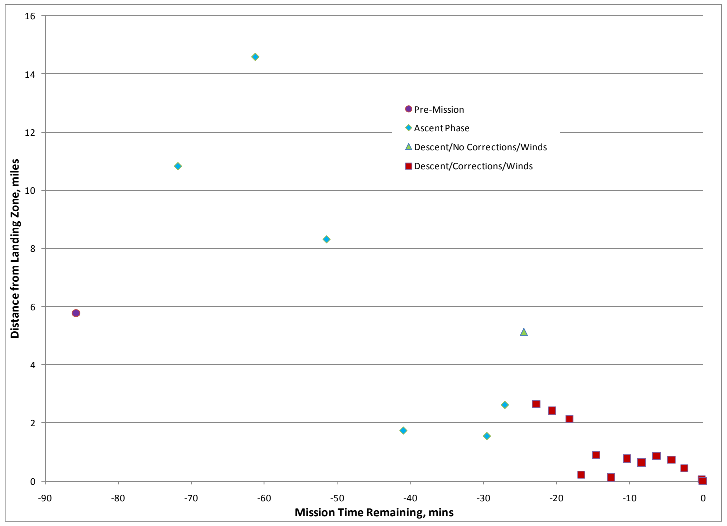
Since the process of using the RTMP does not differ between a live mission and one that is being played back, the results seen in a playback would be the same results seen during a real mission.

Real-Time Mission Predictor Results

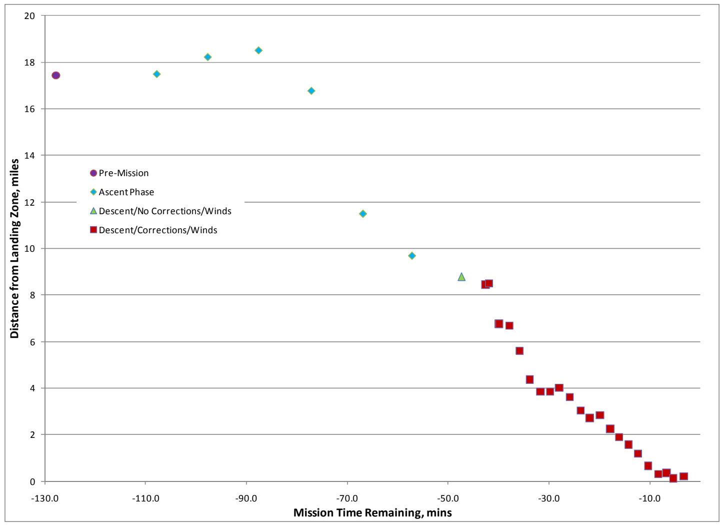
Since the objective of the RTMP is to increase the accuracy of the predicted landing zone, it is best to show the results of how the landing zone prediction evolves during a mission. In order to obtain the results shown, the missions were played back in real-time using the original raw packets as recorded during the missions.

The overall results of all four missions are shown below in Figure 75 and indicate that as the mission progresses, the RTMP is able to update the system parameters. Using the updated parameters the RTMP's accuracy for the predicated landing zone increases. The increase in accuracy has the largest change just after the system is able to update the system parameters during descent. Overall the results indicate that the RTMP is able to predicate the correct landing zone to within three miles only four minutes after burst, which is normally about 28 minutes before touchdown.

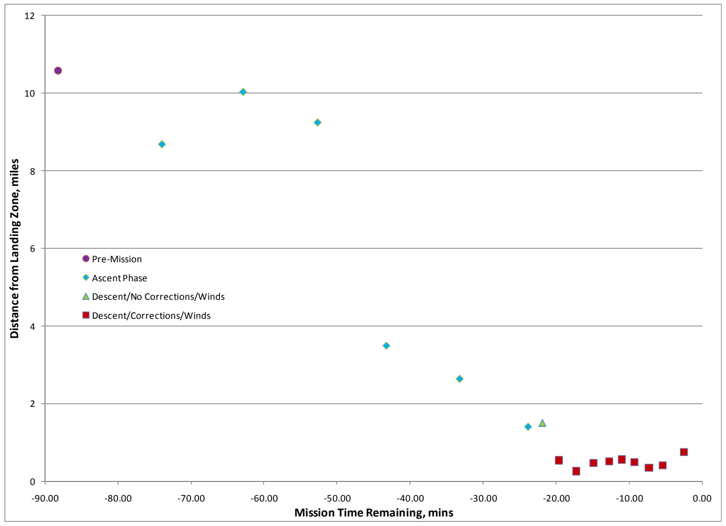
Since the average descent time is on the order of 30 minutes, this means that with 93% of the descent time remaining the tracking team can be within a three mile radius of the landing zone. Once the tracking crew is within the area of the landing zone, the predictions only gets better. Approximately 20 minutes before landing, the prediction has now increased to within one mile of the landing zone, once again giving the tracking crew time to plan their final location and reposition as needed.



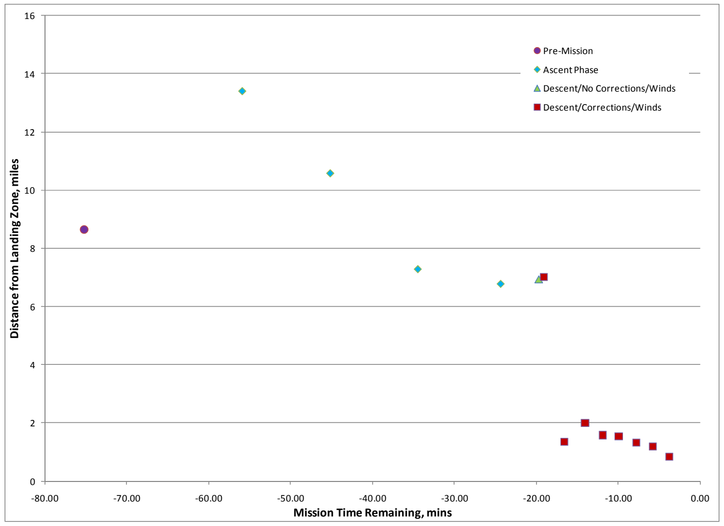
ASTRO-11



ASTRO-12



ASTRO-13



ASTRO-14

Figure 75 Landing Zone Predictions

ASTRO-11 Results

ASTRO-11 was originally flown on Saturday, Nov. 22, 2008 with the following configuration:

- Payload Weight: 8.5 lbs
- Tracking Weight: 4.0 lbs
- Chute Size: 60”
- Chute and Ring weight: 1.0 lbs
- Balloon: Kaymont 3000gram
- Fill Volume: ~520cf Helium
- Launch location: OSU Design and Manufacturing Lab (36.1331, -97.0814)
- Launch time: 15:00:00zulu
- Predicted Peak Altitude: 100,000 ft.

The pre-mission flight profile is shown in the following figure.

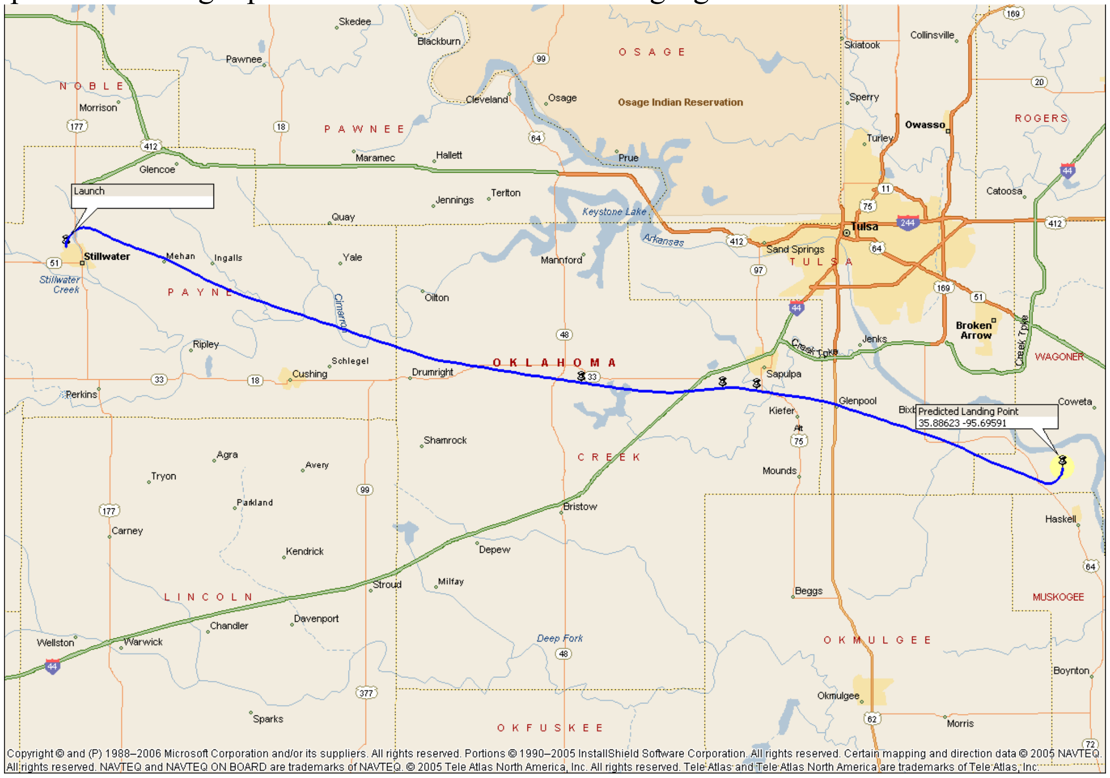

Launch
Stillwater
Predicted Landing Point
35.88623 -95.69591
O K L A H O M A
Tulsa
Copyright © and (P) 1988–2006 Microsoft Corporation and/or its suppliers. All rights reserved. Portions © 1990–2005 InstallShield Software Corporation. All rights reserved. Certain mapping and direction data © 2005 NAVTEQ. All rights reserved. NAVTEQ and NAVTEQ ON BOARD are trademarks of NAVTEQ. © 2005 Tele Atlas North America, Inc. All rights reserved. Tele Atlas and Tele Atlas North America are trademarks of Tele Atlas, Inc.

Figure 76 ASTRO-11 Flight Path Prediction

To test out how effective the RTMP is at predicting the landing zone data from the original flight ASTRO-11 was used in real-time. The first objective was to establish the location of the pre-mission landing zone using the flight configuration of the flight, along with the GFS winds from the morning of the flight. Once this baseline had been established the RTMP was given access to data in real-time by the playback feature in the tracker. To obtain the results of updated prediction on the landing zone, we decided that the prediction would be updated approximately every 10 minutes during the ascent phase and every 2 minutes during the descent phase. Once the flight was completed the predicted landing zone where processed to determine their distance from the real landing zone. The results of this data, shown in Figure 77, indicate the overall trend of an increase in accuracy in the final solution as the mission progressed.

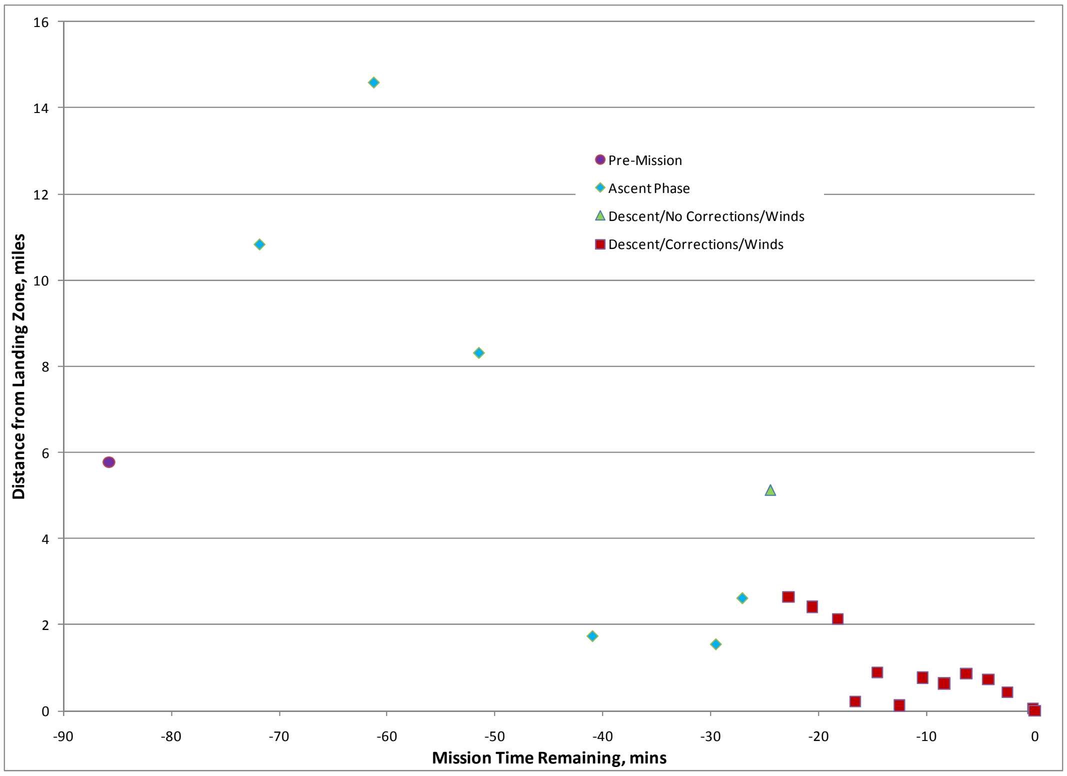


Figure 77 ASTRO-11 Landing Zone Range Error as a Function of time to Landing

To since the distance to the target landing zone is only a part of the picture, the direction and change of location as play an important role, the ground crew really needs to have an understanding of how the target landing zone is moving with respect to the local roads. The need for being able to quickly reposition for the landing zone depends heavily on the grid system that is available for the tracking vehicles. To aid in this task, the results of the updated landing zones are overlaid on a map of the area. This overview not only provides a means of 'planning' the route but also givens an indication of how a typical solution marches toward the landing site. To give an idea of the overall scale of the mission, from launch to landing, the flight profiles along with all the predictions made are shown in Figure 78.

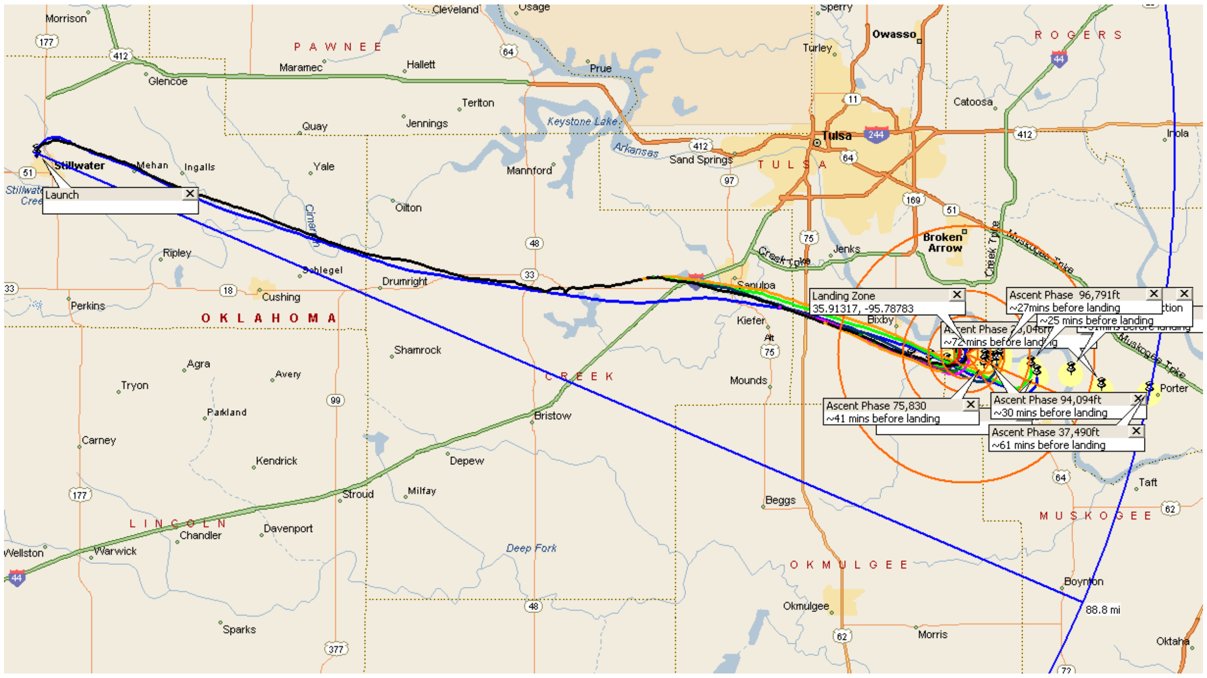


Figure 78 Overall Flight Profile and Predicted Landing Zones for ASTRO-11

The results above show the general tead of the predicted landing zone marching closer to the actual landing zone as the flight progresses. However, what is not clear, on this scale, is the separation in the landing zones between predictions. To clarify the separation of predicted landing zones, it was decided to analyze the ascent and descent phase separately. The first phase is shown in Figure 79 and focuses on the predictions that occurred during pre-mission and ascent phase of the mission.

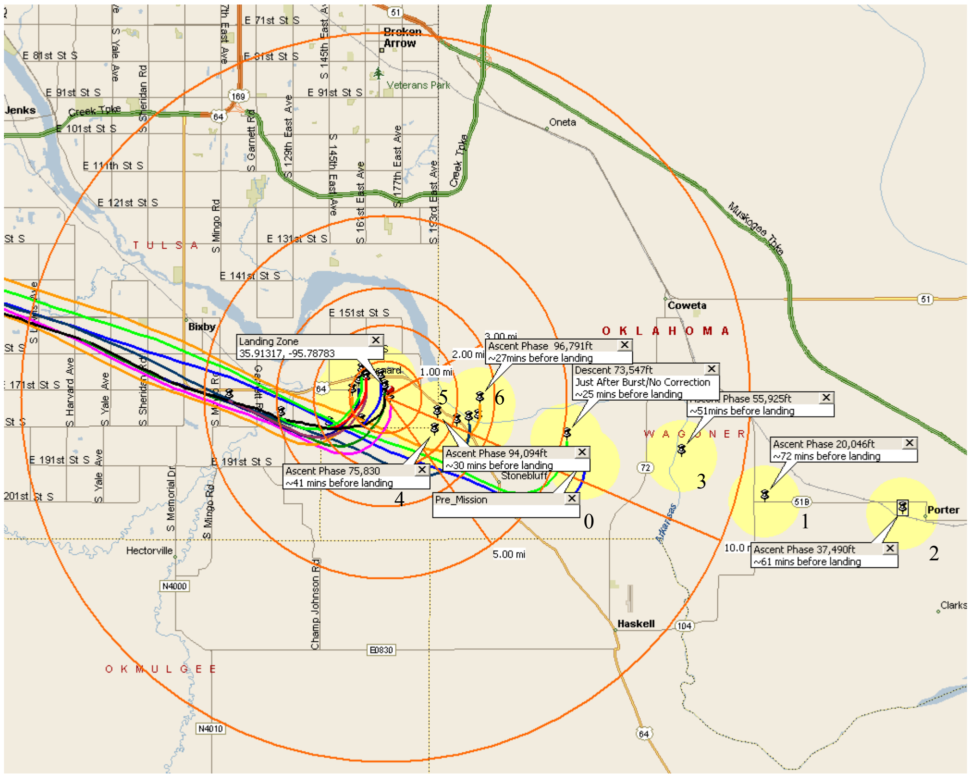


Figure 79 Predicted Landing Zones for ASTRO-11 During Ascent Phase

During the ascent phase the error in distance to the landing zone actually increases at first and then as the mission progresses this error decreases. The main source of this behavior is the differences between the predicted ascent rate profile and the real ascent rate profile. As was discussed earlier in the paper, the ascent rate is by no means linear and in fact does change as a function of altitude. The results of the ascent rate profile and the predicted rate profile are shown in Figure 80 and indicate that during the ascent the real system had an ascent rate that was slower than originally predicted. Since there was a difference in the ascent rates, the RTMP was able to detect this and then apply this ‘correction’ to the updated predictions. As a result, early on the distance to the predicted landing zone actually increases until the difference between the two approaches a constant offset near 75k feet, at which time the solution improves with time.

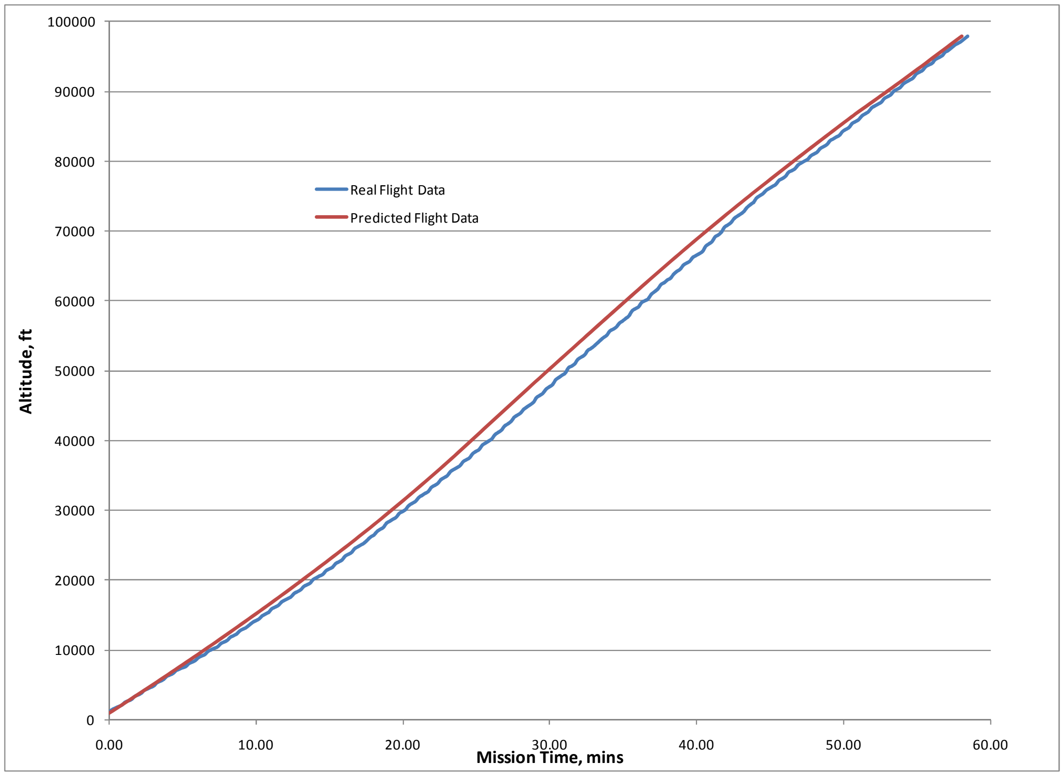
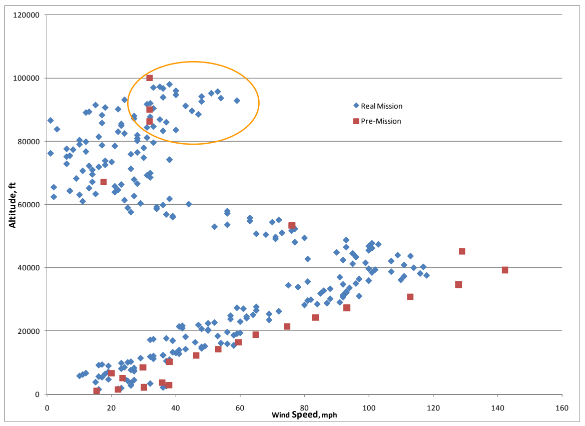
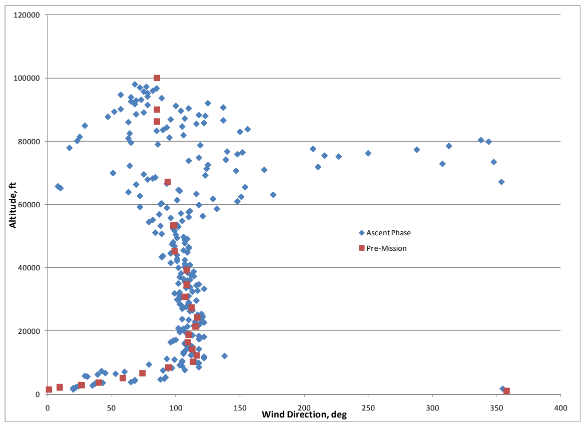


Figure 80 Predicted versus Actually Ascent Rate Profile for ASTRO-11

While the solution does start to improve above 75k feet the solution does not continue to increase in accuracy. Once the system crosses 86k feet the pre-mission winds aloft predictions stop and result in a constant wind speed and direction for the remained of the prediction. However in the real environment this rarely if ever happens. As a result once the system crosses this 86k feet threshold, there will be an increase in error due to differences in predicted and mission winds. As can be noted in Figure 81, the real system encounters higher winds aloft then was originally predicted during pre-mission, as a result the predicted landing zone actually moved farther away.



Wind Speed



Wind Direction

Figure 81 ASTRO-11 Winds Aloft Changes between Flight and Pre-Mission

Once the balloon has burst, it transitions from the ascent to the descent phase. During this time the RTMP is not able to update the system parameters, with the exception of winds aloft. During the ascent phase the RTMP was updating the winds profile using the system's current speed and heading. Once the system has updated the winds profile, it will them use this 'new' profile in place of the pre-mission winds file to create a new appended mission profile. The new winds profile will contain a mixture of older winds data, at higher altitudes, and new winds data for altitudes the system has already seen. This update in winds helps increase the accuracy of the descent phase, but as discussed earlier this is not the only source of error during descent. One of the major unknowns during descent is the overall drag that is acting on the system. This drag can come from several sources: Effective size of the chute, weight of the system, and effects of any remaining balloon remnants. The results of the RTMP during the descent phase are shown in Figure 82 and indicate how the accuracy of the solution continues to increase, after it is able to update the system parameters.

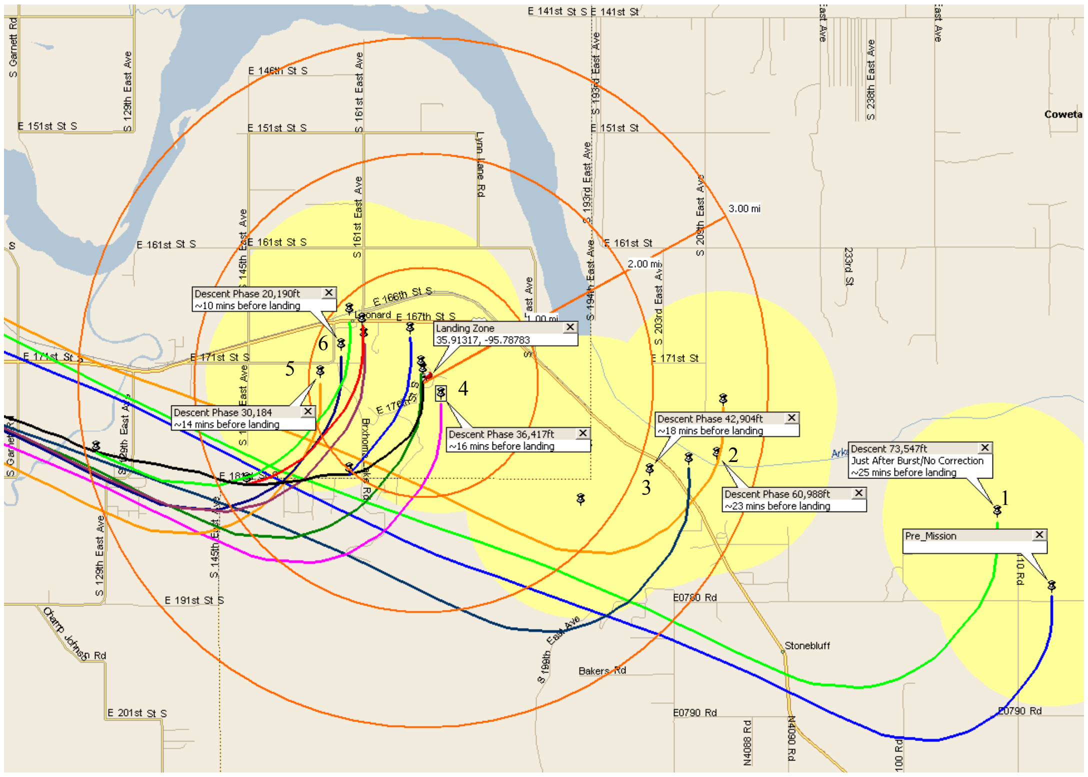


Figure 82 Predicted Landing Zones for ASTRO-11 During Descent Phase

Once the RTMP was able to start correcting for the real system parameters, the accuracy of the solution increased to under three miles and then continued to increase as the flight progressed. One of the major items of interest was the large increase in the solution between the time of the balloon just after burst and the solution once the correction factor was found. To see the resulting descent profiles for the mission and that of the predicted mission, the two profiles where plotted for compassion and are shown in Figure 83. The results indicate that during the mission the chute's effective drag was less then what was predicted during the planning stage. Approximately two minutes after burst, the RTMP detected that a correction factor of 0.80 needed to be applied to the original model to allow it to behave as the real system. This updated correction factor is also plotted in Figure 83. Once applied the system was now able to more accurately model the descent phase, and the only variability left was the possibility of winds changes between the ascent and descent phases. To see if there was any major shift in the winds between the flights, it was decided to process the system's speed and direction during the mission and then compare the results during each of the two phases of the mission. This result of this comparison, shown in Figure 84, did not indicate any major shifts in the winds in either speed or direction during the mission until the system reached an altitude below 10k feet. At this time the surface winds had general increased in speed from less than 15 mph to between 20 and 30 mph, while at the same time there was little to no change in the direction, these results can be seen in Figure 85. The effect of this increase in winds speed can be seen in the final prediction in the landing zone, shown in Figure 86, as a shift to the East in each prediction and then final landing location.

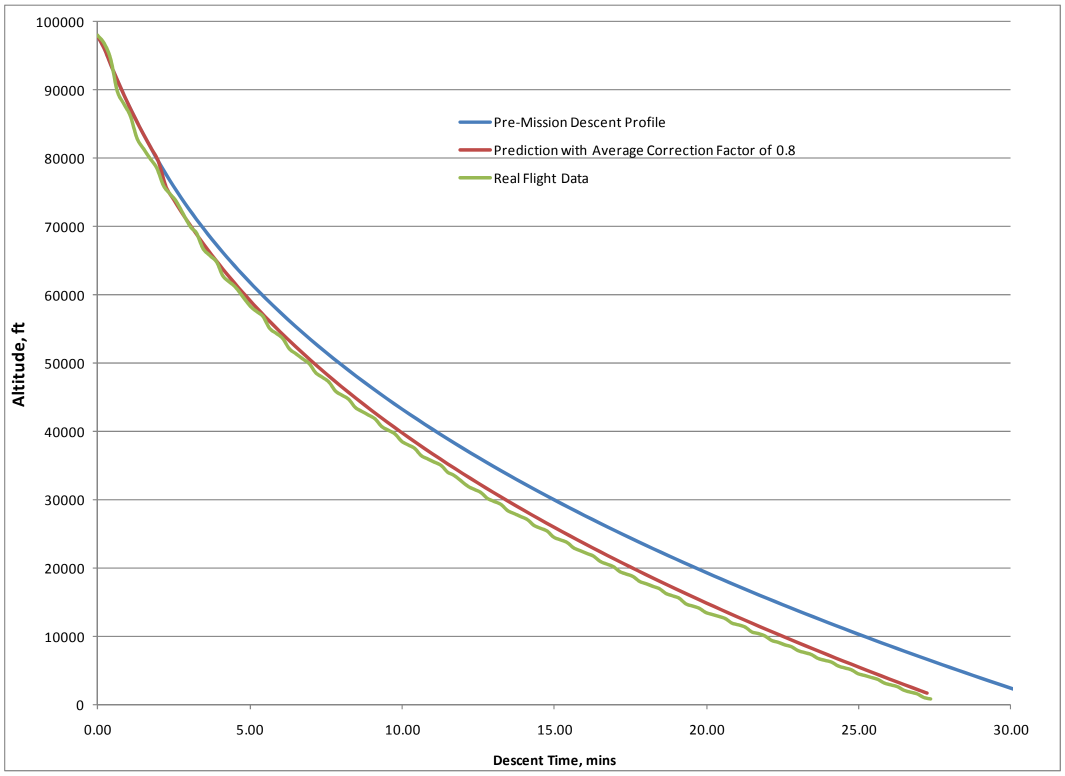
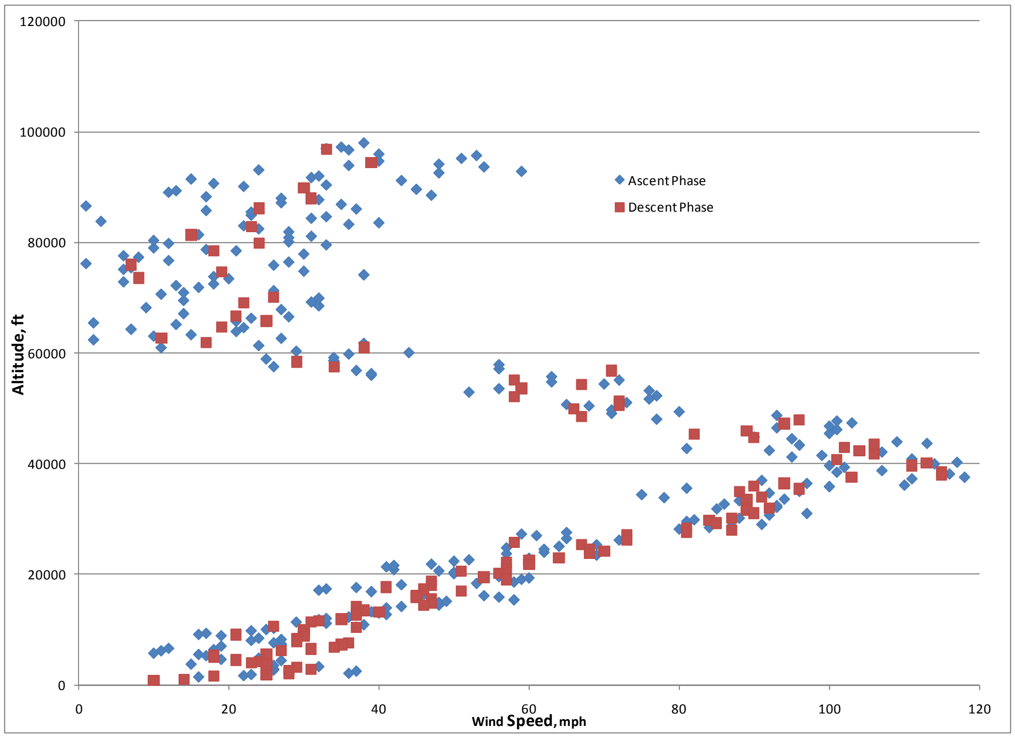
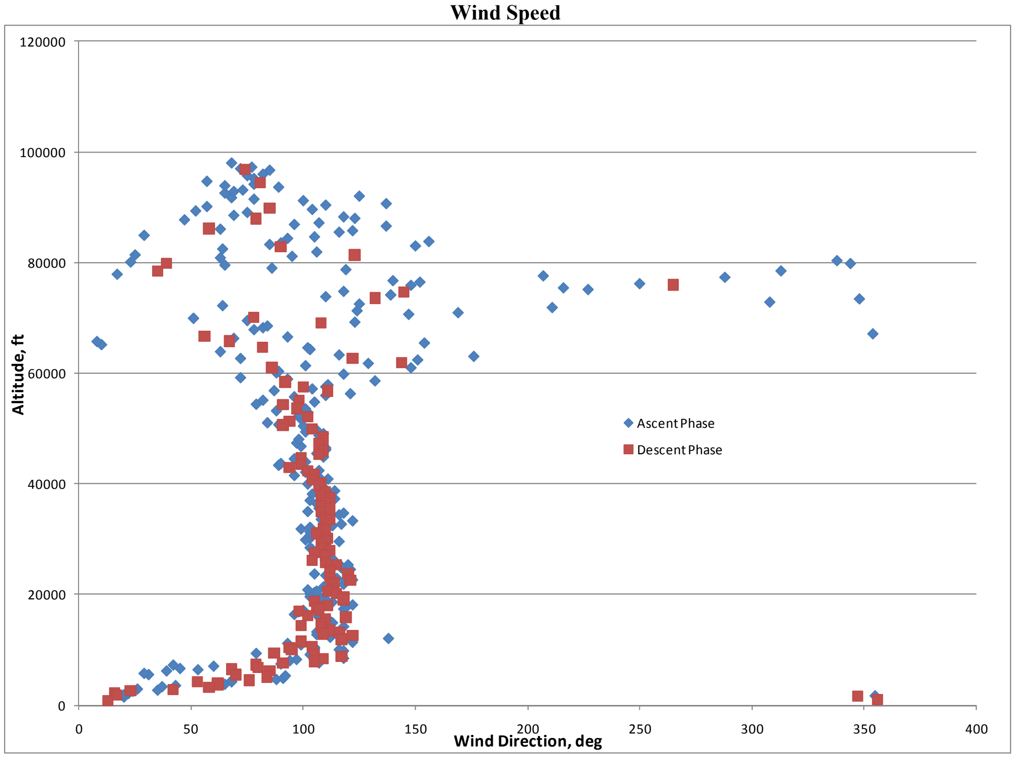


Figure 83 Descent Phase Profiles for ASTRO-11

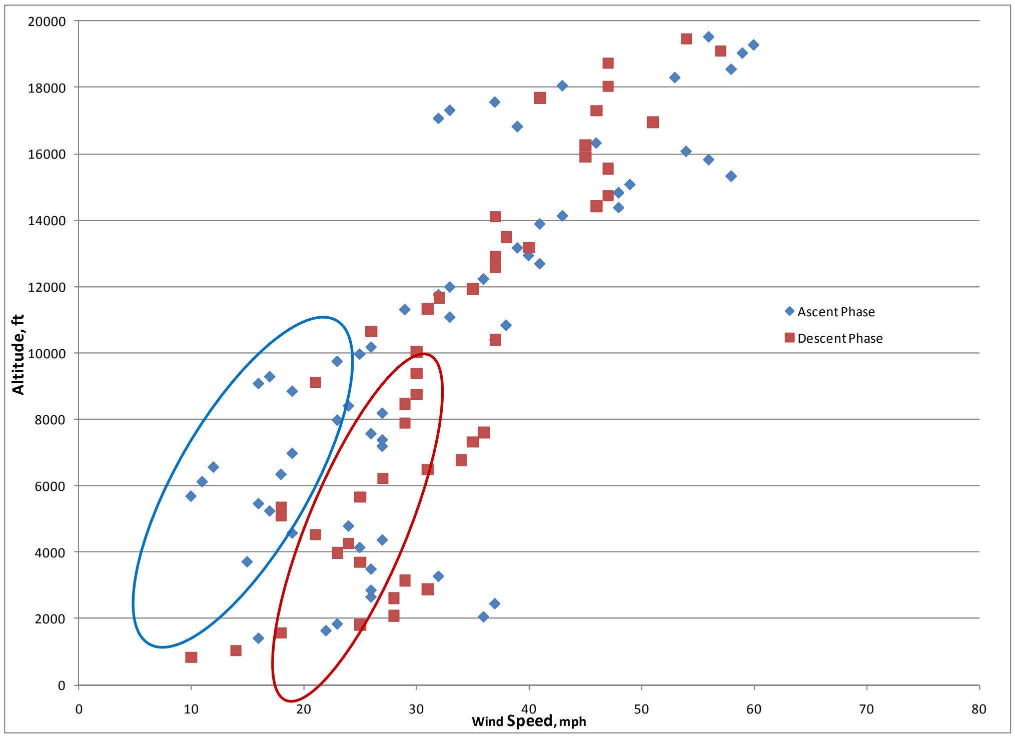


Wind Speed

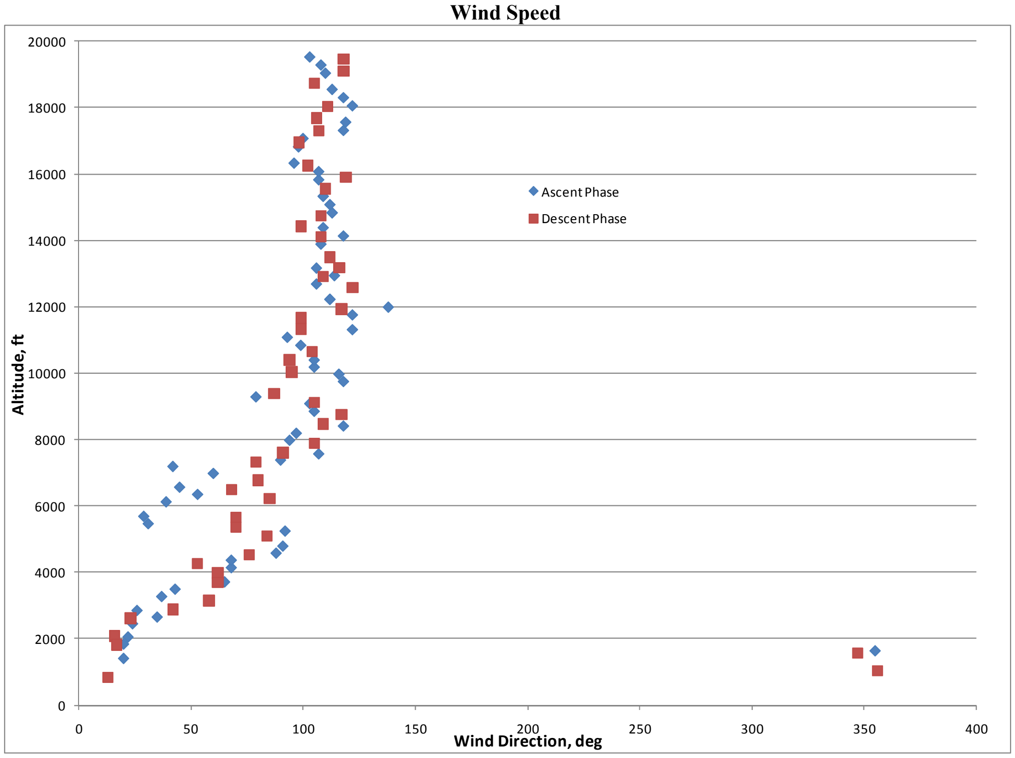


Wind Direction

Figure 84 ASTRO-11 Winds Aloft Changes during Flight



Wind Speed



Wind Direction

Figure 85 ASTRO-11 Winds Aloft Changes at Altitudes below 20,000 feet

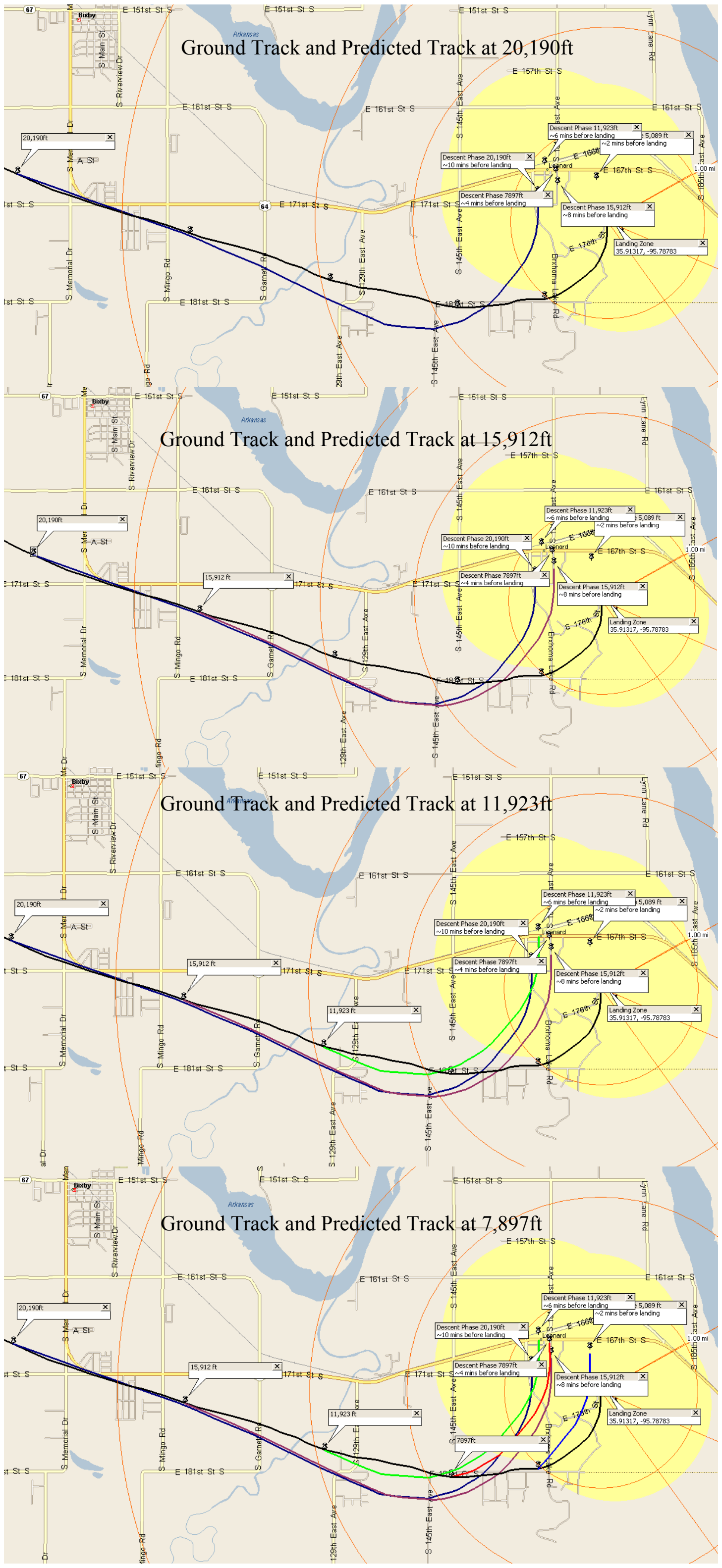

Ground Track and Predicted Track at 20,190ft
Ground Track and Predicted Track at 15,912ft
Ground Track and Predicted Track at 11,923ft
Ground Track and Predicted Track at 7,897ft
20,190ft
15,912 ft
11,923 ft
7897ft
Descent Phase 20,190ft ~10 mins before landing
Descent Phase 15,912ft ~8 mins before landing
Descent Phase 11,923ft ~6 mins before landing
Descent Phase 7897ft ~4 mins before landing
5,089 ft ~2 mins before landing
Landing Zone 35.91317, -95.78783
Bixby
E 151st St S
E 161st St S
E 171st St S
E 181st St S
S Mingo Rd
S 145th East Ave
S 129th East Ave
1.00 mi

Figure 86 Predicted Landing Zone below 20,000 feet for ASTRO-11

ASTRO-12 Results

ASTRO-12 was originally flown on December 11, 2008 and is one of the heaviest payload trains flown to date. The configuration for the mission was as follows:

- Payload Weight: 11.5lbs
- Tracking Weight: 4.0 lbs
- Chute Size: 120”
- Chute and Ring weight: 1.5 lbs
- Balloon: Kaymont 3000gram
- Fill Volume: ~520cf Helium
- Launch location: OSU Design and Manufacturing Lab (36.1331, -97.0814)
- Launch time: 15:00:00zulu
- Predicted Peak Altitude: 100,000 ft.

The pre-mission flight profile is shown in the following figure.

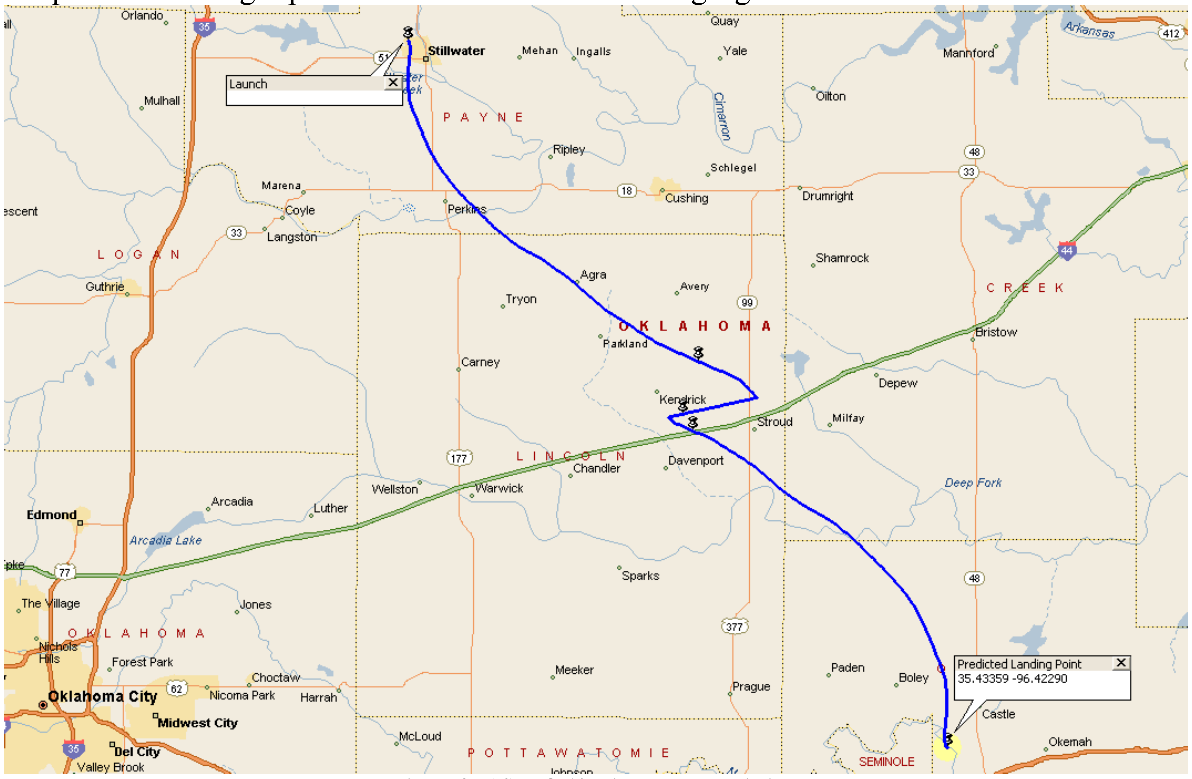


Figure 87 ASTRO-12 Flight Path Prediction

One the pre-mission prediction was made, the RTMP was then utilized to determine the updated predicted landing zone during a real time play back of the mission. Once this data was collected the results were compiled and then plotted as a function of mission time remaining compared to the error in distance from the projected landing zone to the actually landing zone. The results, shown in Figure 94, indicate the while the overall trend is an increase in accuracy to the final solution, there still remains a large difference between the ascent and descent phase. The second item of note is that during the descent phase the error does not appear to decrease as rapidly as ASTRO missions, 11, 13, and 14.

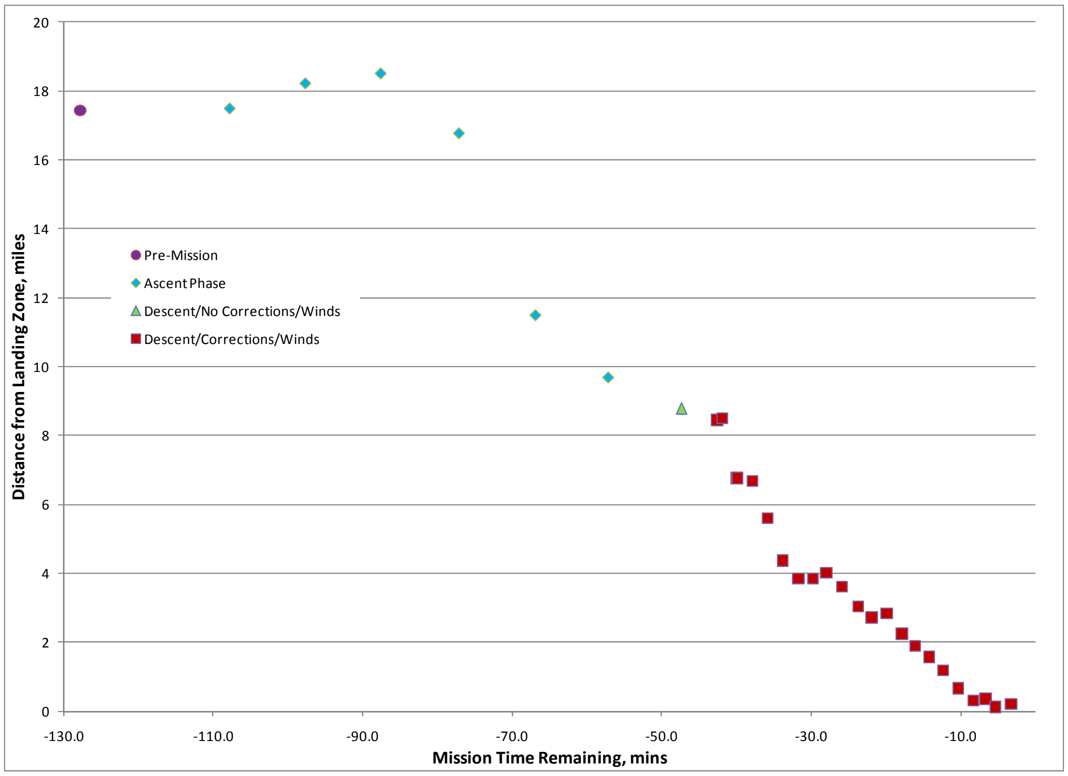


Figure 88 ASTRO-12 Landing Zone Range Error as a Function of time to Landing

ASTRO- 12 has a lower descent rate than the other mission, as can be seen in Figure 89, and this is thought to contribute to the change in error over this flight. Typical an ASTRO mission takes less than 30 minutes to descend from approximately 100k feet, where as this mission took nearly 45 minutes. This slower descent rate means that the system is more susceptible to changes in the winds profile between the ascent and descent phases, where as the faster systems are not as affected due to their higher speeds.

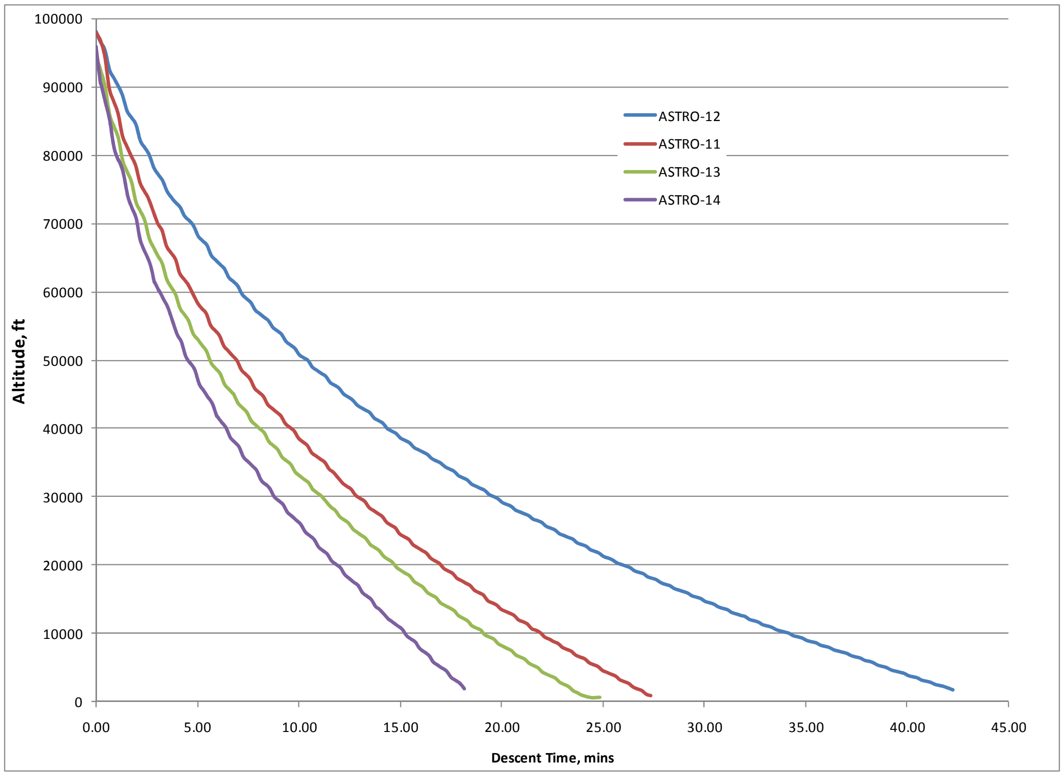


Figure 89 Descent Rates for ASTRO missions 11 to 14

One of the other items that can affect this error is the correction factor that needs to be applied during descent. Interesting enough ASTRO-12 had one of the lowest average correction factors that were applied during descent, which implies that the predicted model should have behaved correctly, assuming the winds encounter during the descent where close that those seen during ascent. To determine if changes in winds were the primary cause, a comparison of the winds profile between the ascent and descent phases was produced. The results, shown in Figure 91, indicate that at higher altitudes the winds during the two phases matched well, however the winds below 20k feet do not align in wind speed. Below 20k feet there is a major difference between the winds experience during ascent and descent. As can be seen in Figure 92, the greatest difference occurs between 20k and 14k feet where a difference upwards of 20 mph was encountered. As the system encountered lower altitude the winds started to converge until approximately 8,000 feet, where the winds had converged.

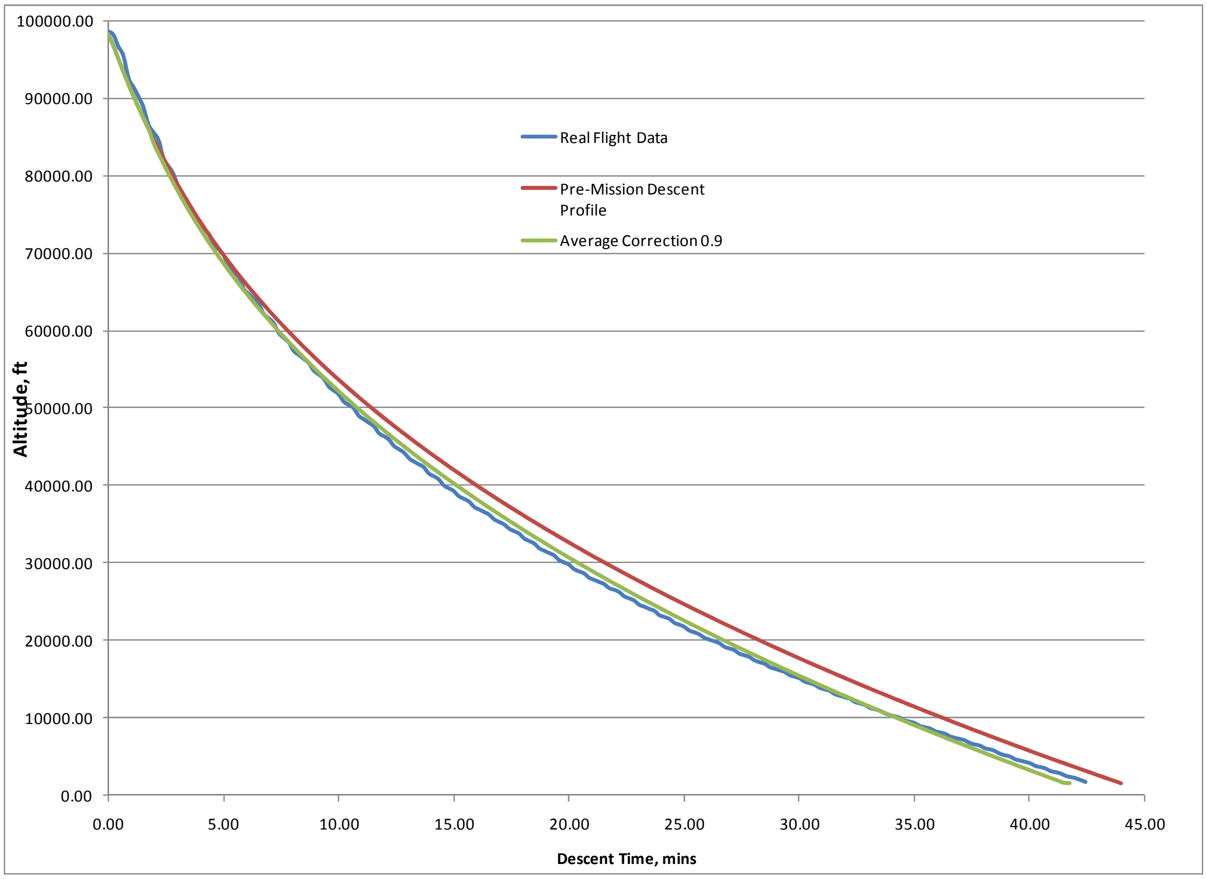
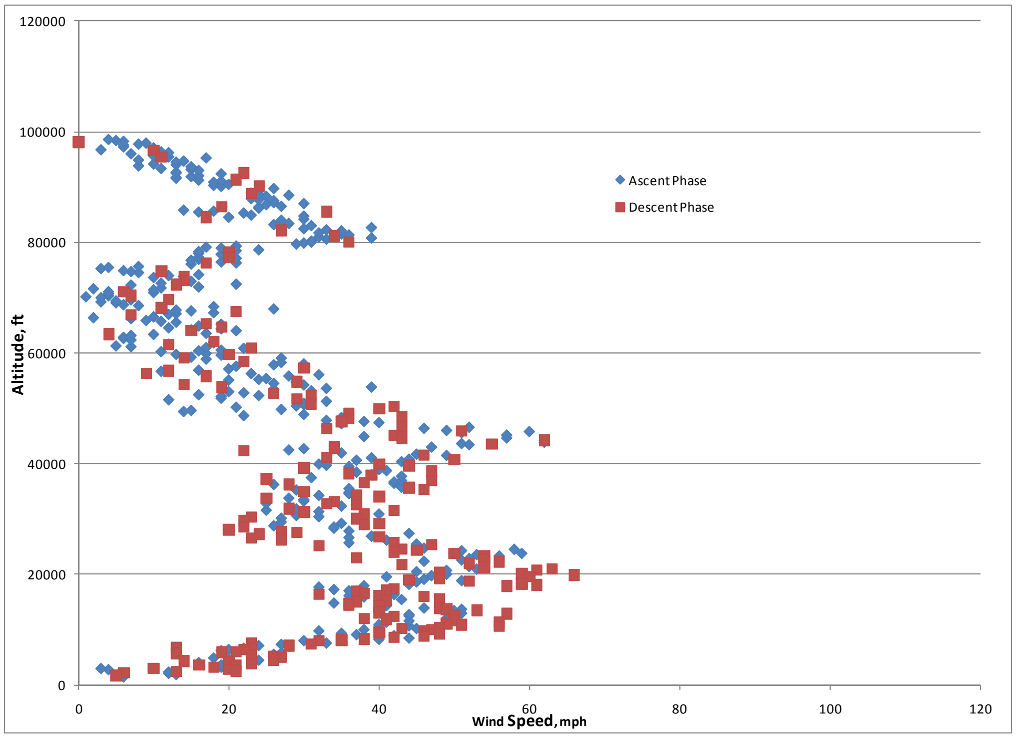


Figure 90 Descent Phase Profiles for ASTRO-12



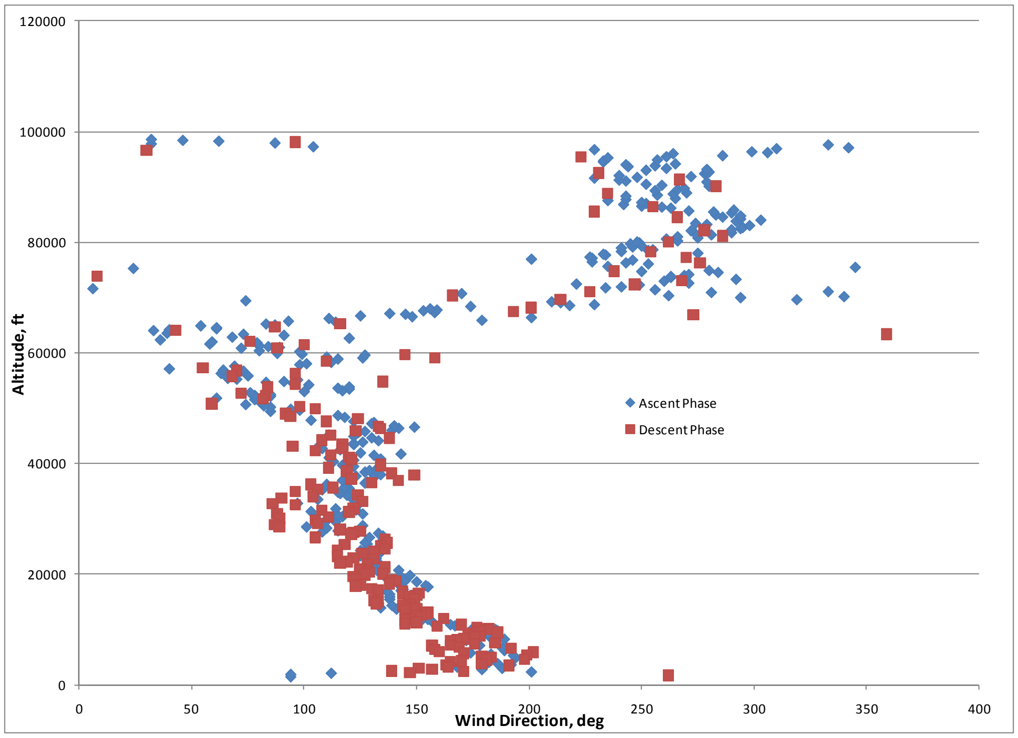
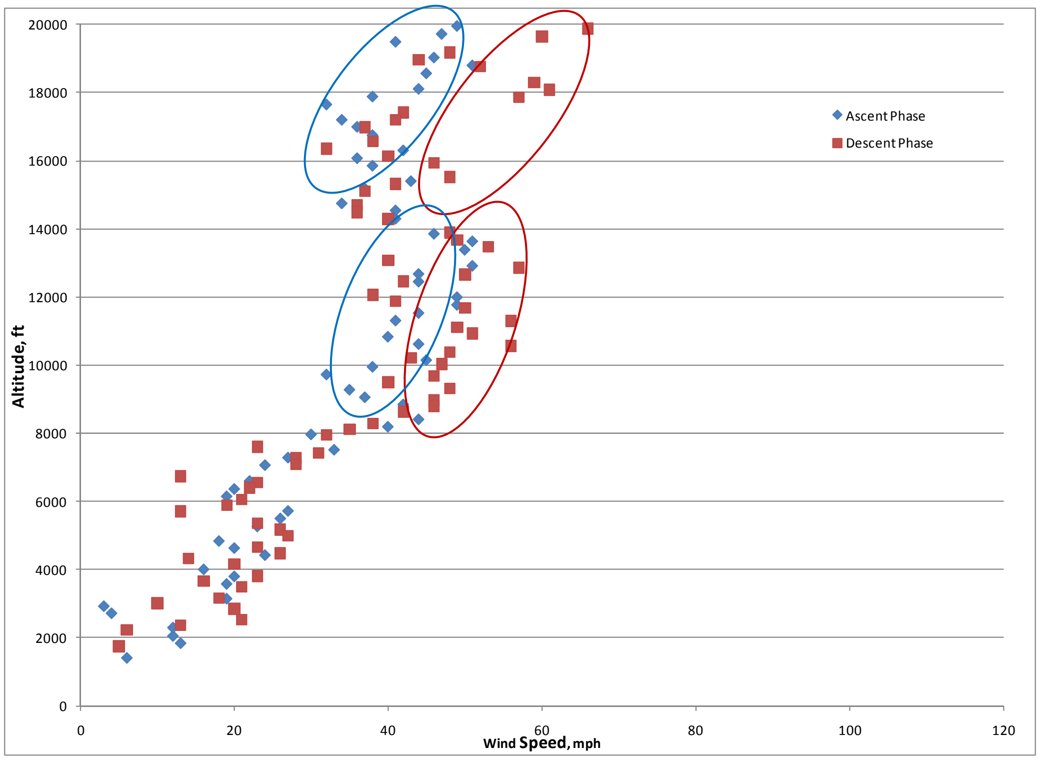


Figure 91 ASTRO-12 Winds Aloft Changes during Flight



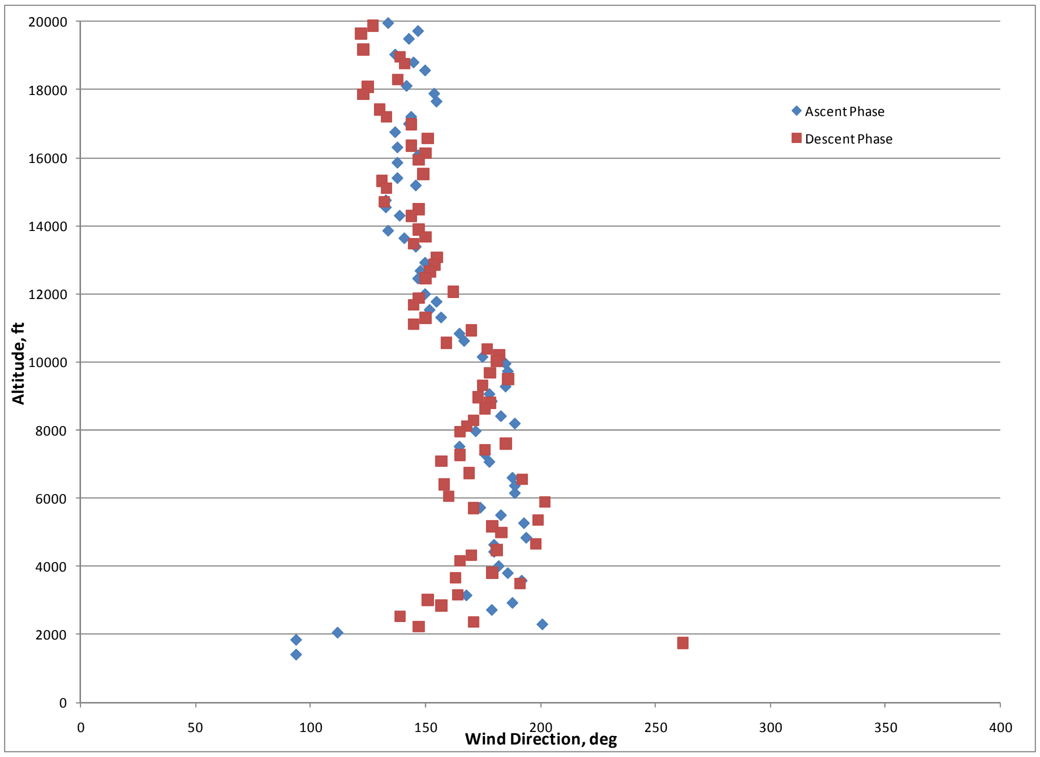


Figure 92 ASTRO-12 Winds Aloft Changes below 20,000ft

<u>ASTRO-13 Results</u>

ASTRO-13 was originally flown on March 19th, 2009, with a mission configuration as follows:

- Payload Weight: 10.5lbs
- Tracking Weight: 4.0 lbs
- Chute Size: 60”
- Chute and Ring weight: 1.0 lbs
- Balloon: Kaymont 3000gram
- Fill Volume: ~500cf Helium
- Launch location: OSU Design and Manufacturing Lab (36.1331, -97.0814)
- Launch time: 15:00:00zulu
- Predicted Peak Altitude: 100,000 ft.

The pre-mission flight profile is shown in the following figure.

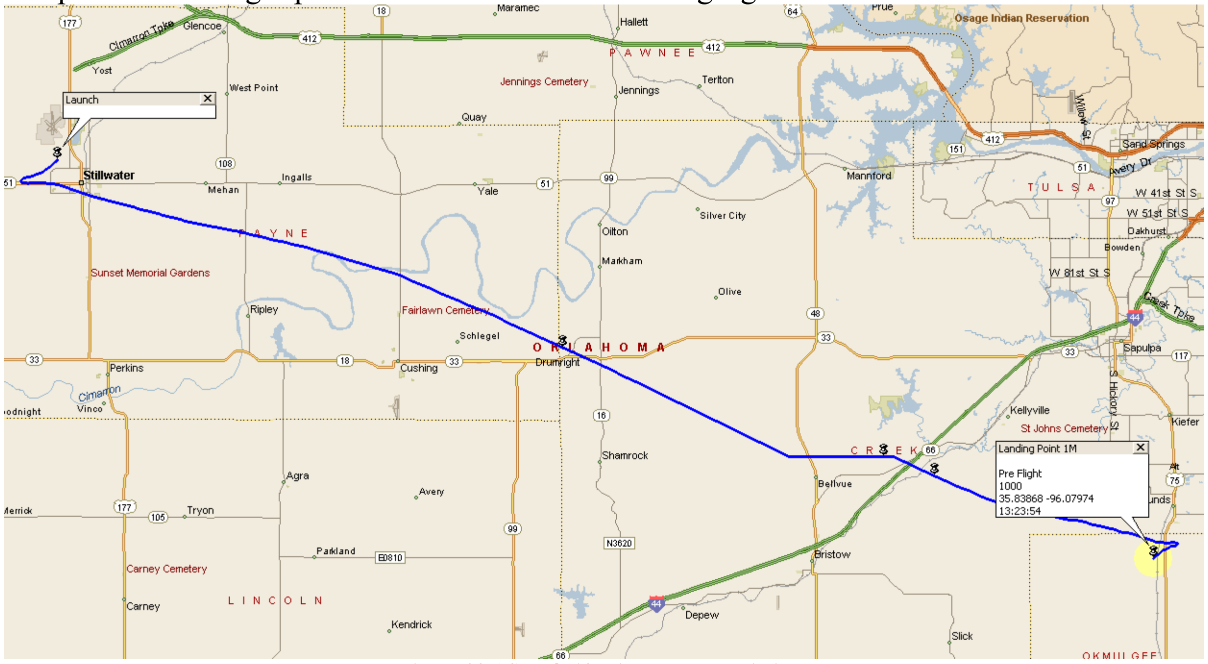


Figure 93 ASTRO-13 Flight Path Prediction

The primary objective was to determine the locations of the predicted landing points during the mission with respect to the actually landing zone. Once the predicted landing zones had been establish, the results were compiled to determine the distance each prediction was from the real landing zone. The results of this data, shown in Figure 94, imply an improvement in the solution as the mission progresses but what is of real interest is the major improvement in accuracy of the solution once the mission has started its descent and had time to correct for the descent parameters.

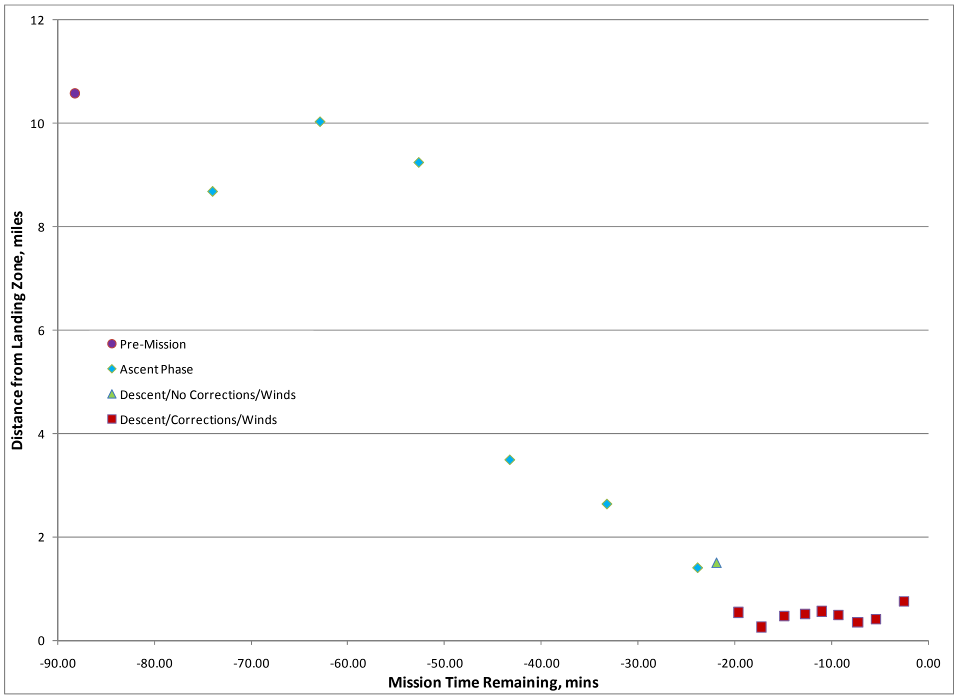


Figure 94 ASTRO-13 Landing Zone Range Error as a Function of time to Landing

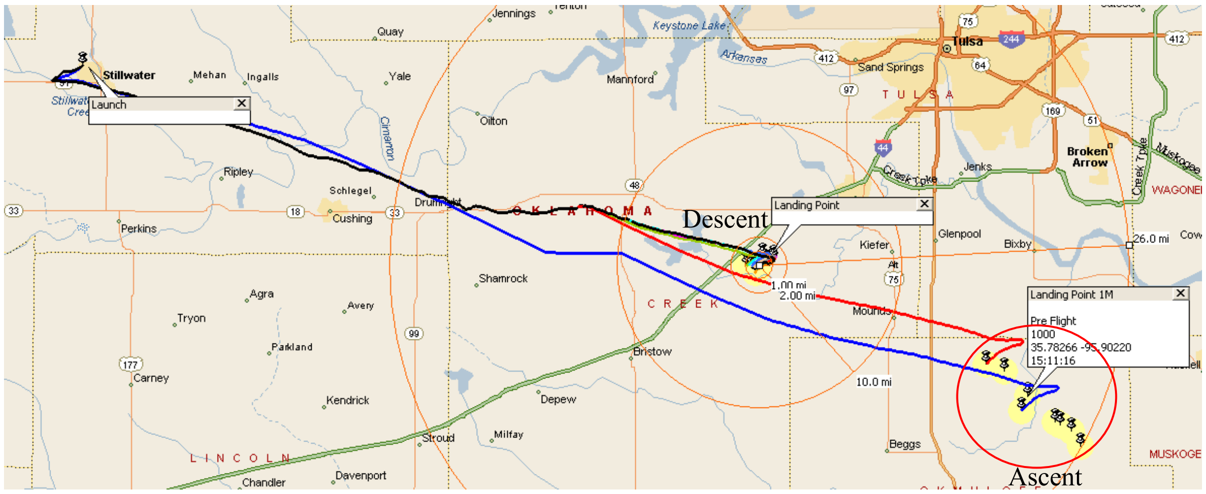


Figure 95 Overall Flight Profile and Predicted Landing Zones for ASTRO-13

The results, shown above, clearly show an increase in accuracy as the mission progresses. However, once again during the ascent phase there is a slight decrease in the predicted landing accuracy until the system crosses approximately 71k feet, or 45 min remaining in the mission. After this time the solution moves to within 3.5 miles or better, while before this time frame the solution was over 9 miles from the final landing zone. There are two possible reasons behind this: Changes in the balloons ascent rate during the mission, or variations in the winds from the pre-mission prediction and the real winds. The ascent rate that occurred during the mission, shown in Figure 96, indicates very little variations between the real and predicted ascent rates. This matching of the data was actually seen while running the RTMP, as the average correction gamma remained at or near one.

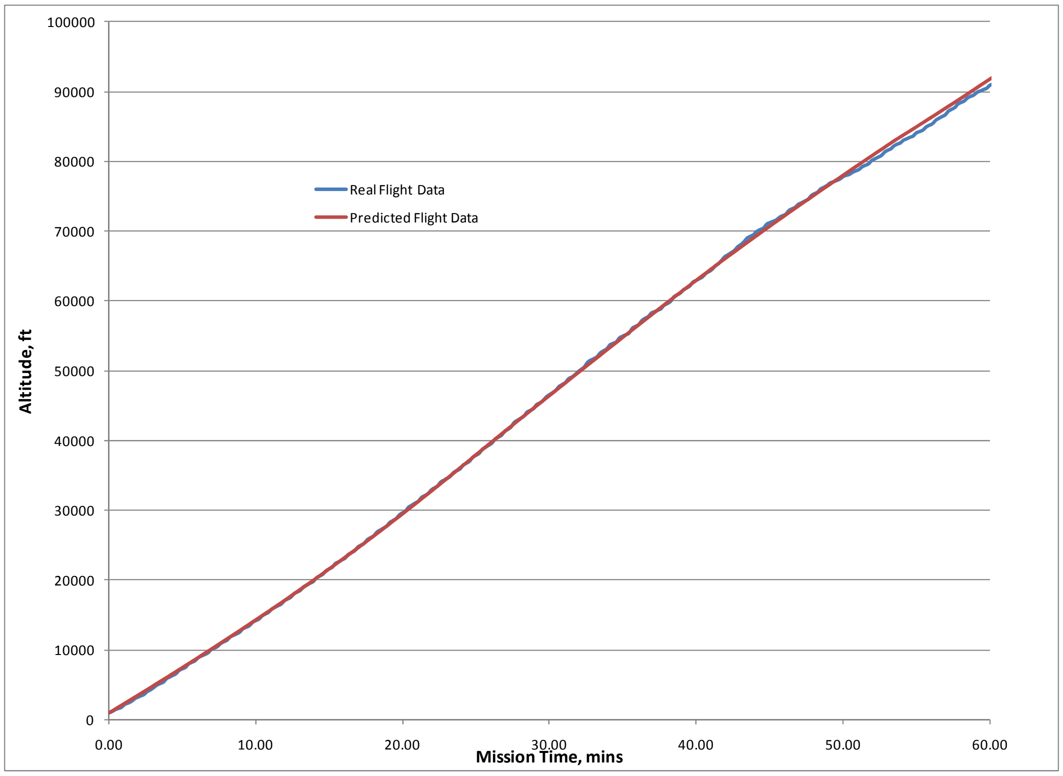
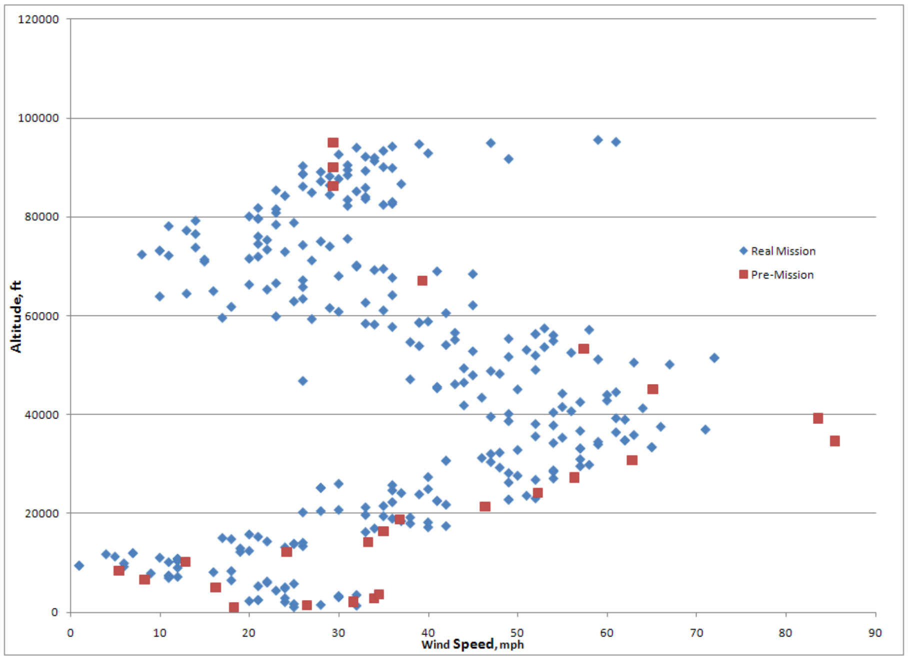
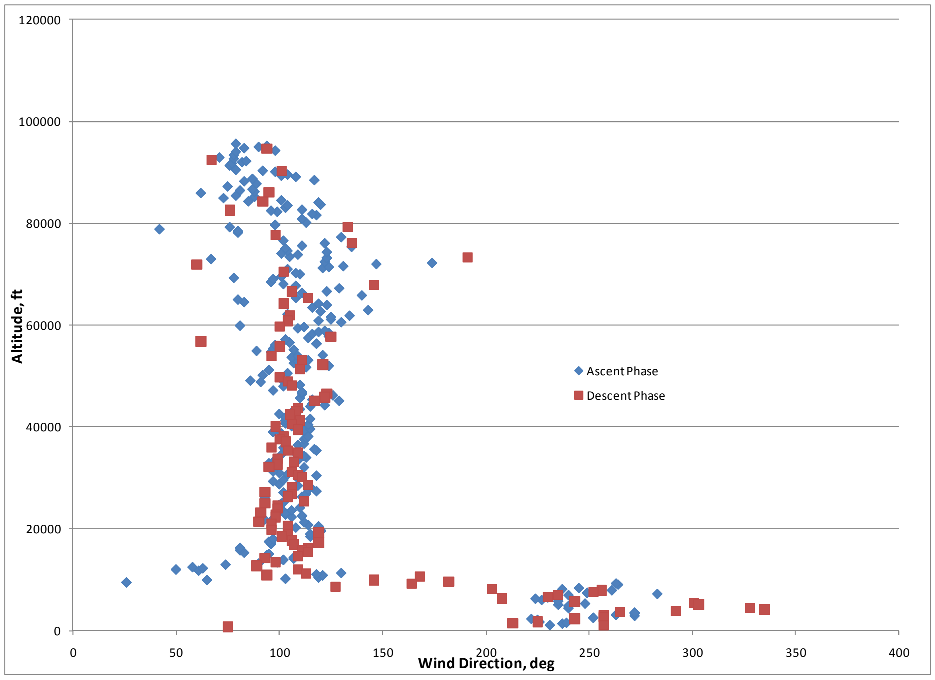


Figure 96 Predicted versus Actually Ascent Rate Profile for ASTRO-13

However since there was a difference between the pre-mission prediction and the real flight path before burst the source of this error must derive from changes in winds-aloft. From the comparison, shown in Figure 97, between pre-mission and in-flight winds, for winds below roughly 70k feet, the winds aloft where predicted to be higher then what was actually encounter. Once the system crossed this threshold, the real winds became stronger then what was predicted. As for the wind direction, there appears to be little to no change during the ascent phase, as compared to the pre-mission winds.



Wind Speed



Wind Direction

Figure 97 ASTRO-13 Winds Aloft Changes between Flight and Pre-Mission

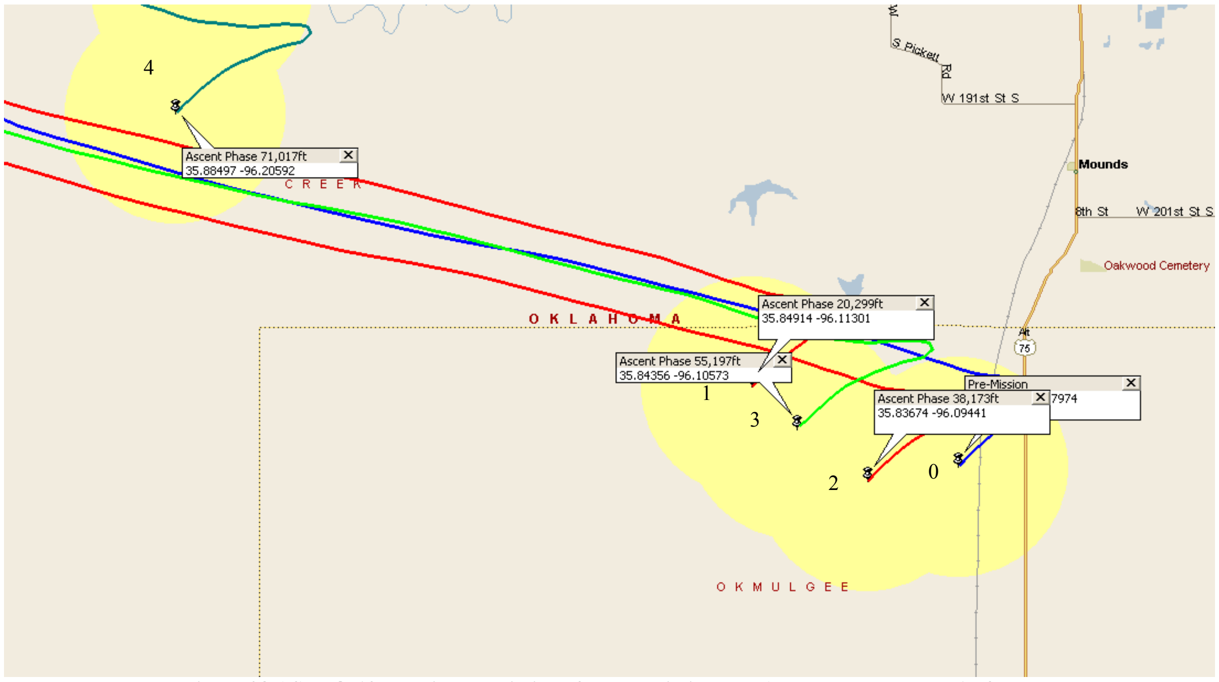
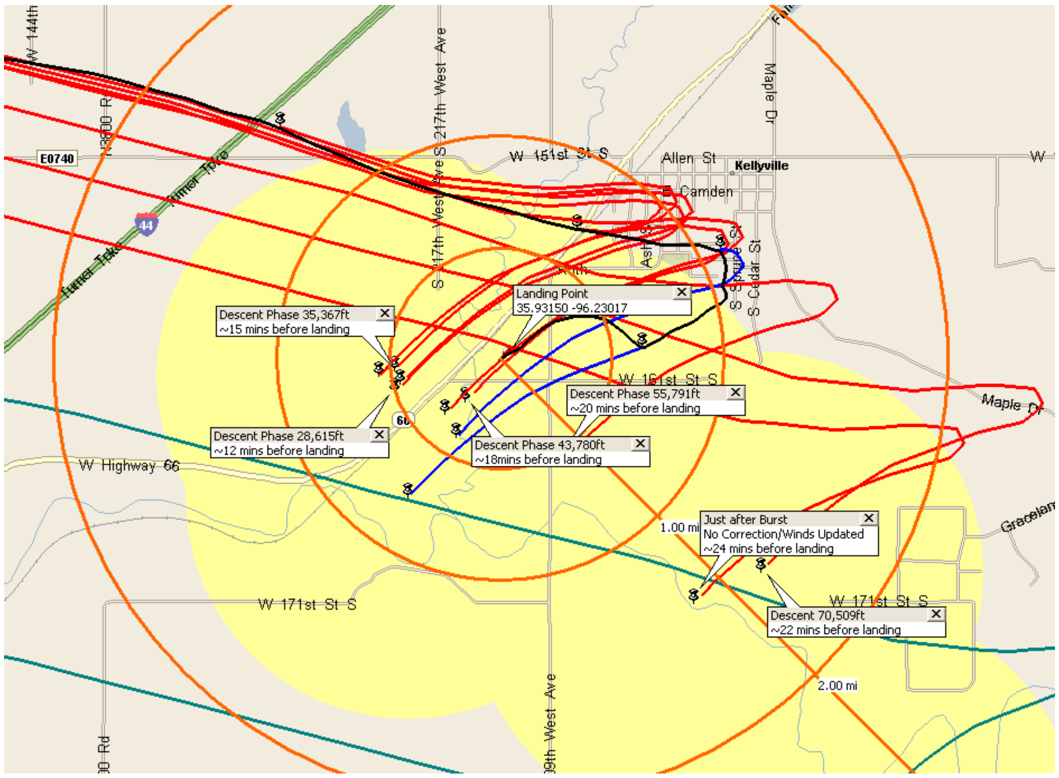


Figure 98 ASTRO-13 Landing Predictions for Pre-Mission and Ascent Phase below 71k feet

By the time the balloon has reached burst, the system has now replaced the pre-mission winds with the real winds aloft encountered during ascent. As such the descent phase now only has two major sources of error that can affect prediction of the final solution: Changes in the descent rate profile and any changes in the winds between the ascent and descent phases. During the descent phase the prediction increased in accuracy from just over 1.5 miles before burst to under a half mile after burst, with corrections. From the results of the flight path, it would appear that once the system reached below 20k feet the surface winds experienced minor changes with respect to the winds encountered during the ascent phase. As can been seen in Figure 99, the ground tracks appear to shift slightly to the East, since the descent rate used in the prediction method appears to match that of the real descent rate the only major source for this would be changes in the surface winds.



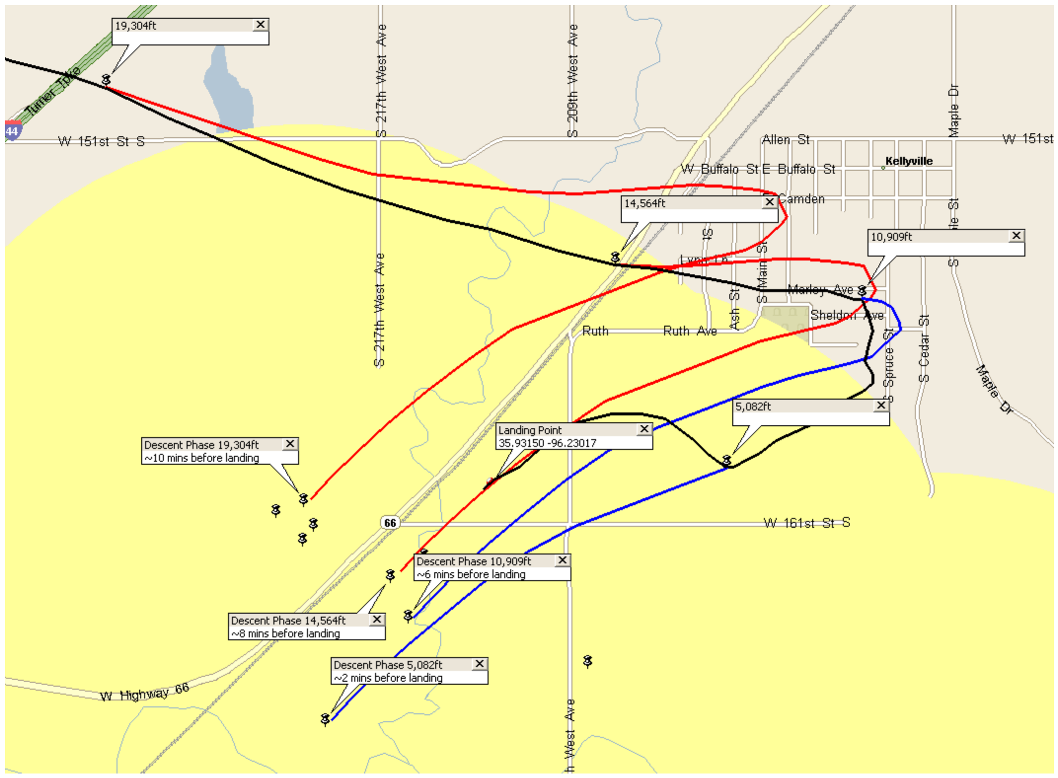


Figure 99 Surface Winds Effects on ASTRO-13

ASTRO-14 Results

The last mission to be analyzed is ASTRO-14 which originally flew on June 17th, 2009. This mission was one of the lightest combined payloads flown to date and during the descent phase this mission encountered a major foaling of the chute and payload train, which resulted in the fasted descent time to date. The mission was flown with a configuration as follows:

- Payload Weight: 8.0 lbs
- Tracking Weight: 4.0 lbs
- Chute Size: 60”
- Chute and Ring weight: 1.0 lbs
- Balloon: Kaymont 3000gram
- Fill Volume: ~460cf Helium
- Launch location: OSU Design and Manufacturing Lab (36.1331, -97.0814)
- Launch time: 15:00:00zulu
- Predicted Peak Altitude: 100,000 ft.

The pre-mission flight profile is shown in the following figure.

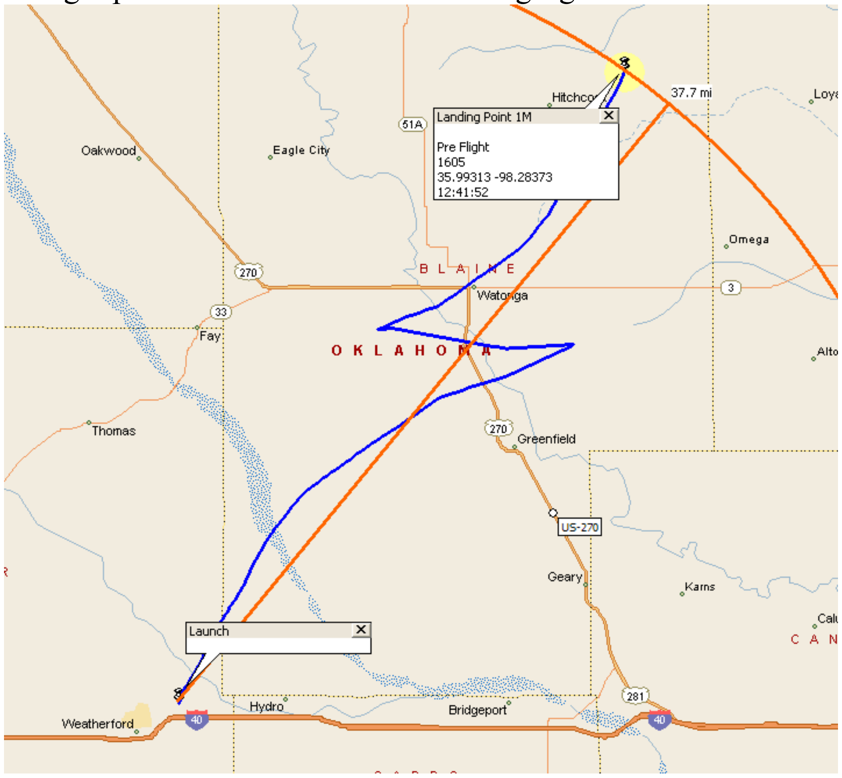


Figure 100 ASTRO-14 Flight Path Prediction

To determine the effectiveness of the RTMP under the flight conditions of ASTRO-14, the original flight data from the mission was loaded and then played back in real-time. During this replayed mission, predictions were made on ten minutes intervals during the ascent phase and then increased to every two minutes during descent. An overview of all the real-time predictions are shown on a map overlay in Figure 101 and give an indication of the major shift in predicted landing zones that occurred between the ascent and descent phases. To clarify this dramatic change in the accuracy of the landing zone error, the results are plotted in terms of the absolute distance in error as can be seen in Figure 102

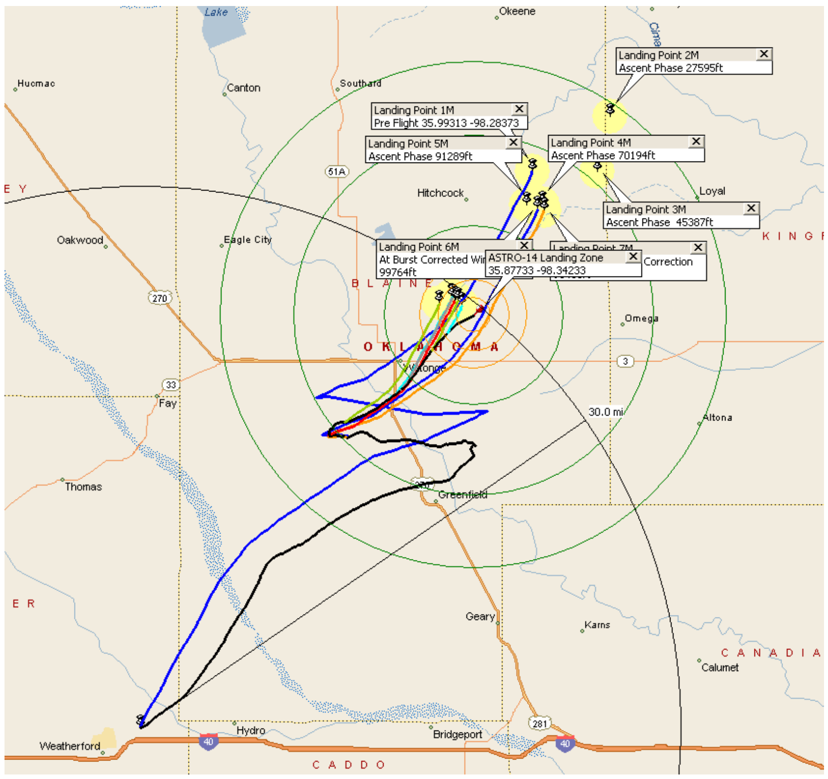


Figure 101 ASTRO-14 Overall Flight Profile and Predicted Landing Zones

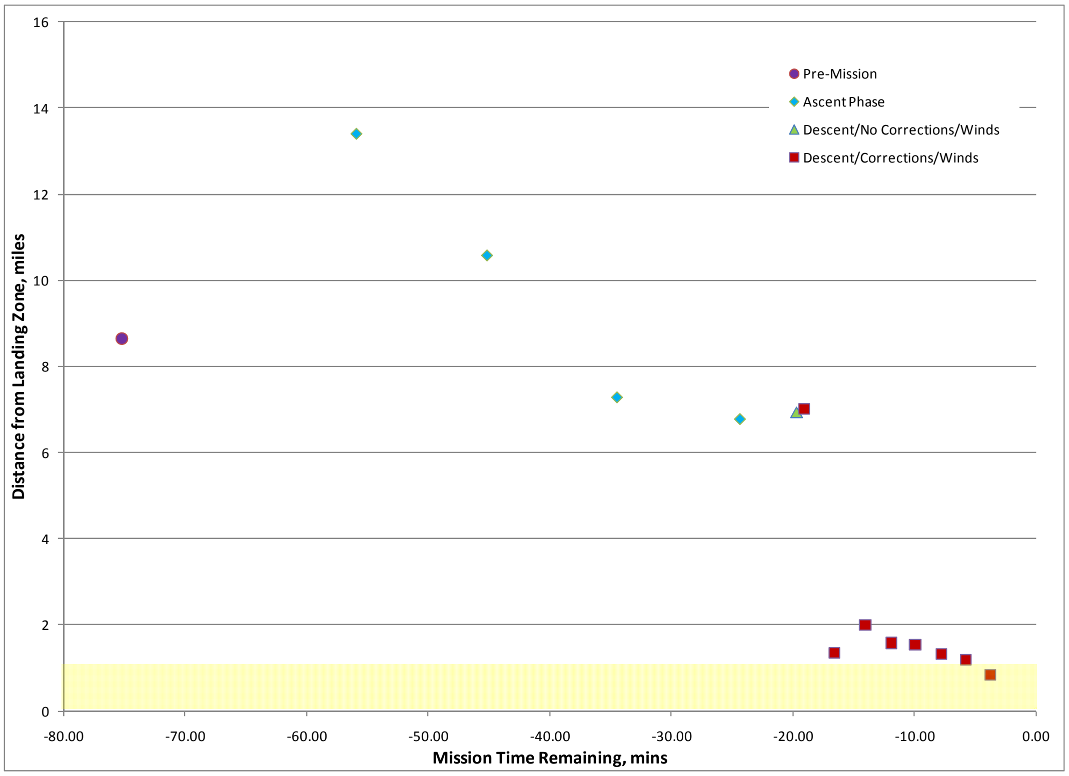


Figure 102 ASTRO-14 Landing Zone Range Error as a Function of time to Landing

Since there was a very large difference between the ascent and descent phase between the predicted phases, we will look at the overall mission to determine the cause. As we've seen in the past missions, the accuracy of the solution does increase as the mission progresses, as it does in this case, the major difference in ASTRO-14 when compared to the other missions is a very large discrete jump in accuracy between the prediction just before burst and the prediction made shortly after the system dynamics is corrected during descent. In order to understand why this happened, we decided to analyze the descent profile in comparison to the predicted profile.

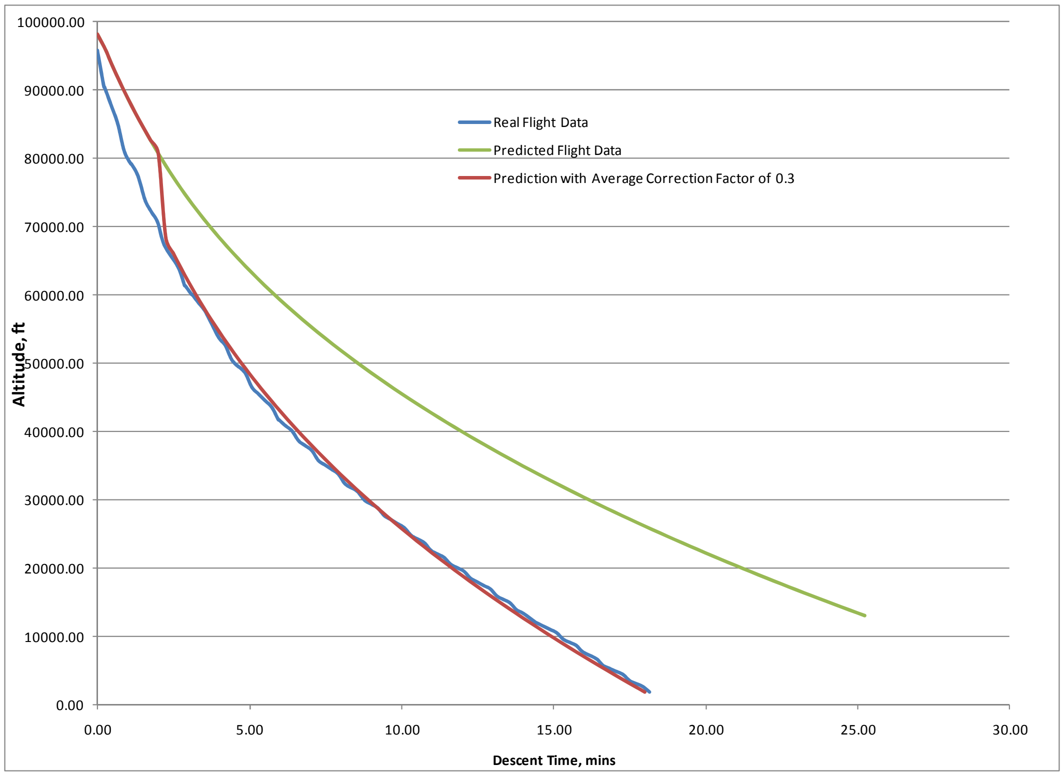


Figure 103 ASTRO-14 Descent Phase Profiles

The results of this descent comparison seem to support the source of the difference. As can be seen in Figure 103, the difference between the predicted behavior and the real system was quite large. Originally it was assumed that the system would be descending under a 60" chute, however due to major fouling of the chute it effective size was drastically reduced.

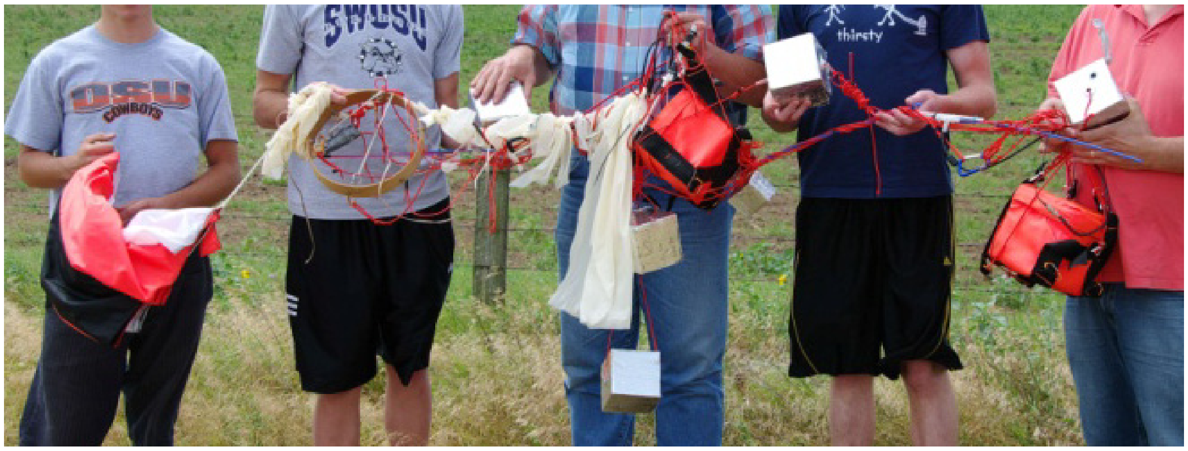


Figure 104 Results of the Chute Fouling during ASTRO-14

During the first two minute phase of the descent the RTMP had not collected enough data to correctly adjust the parameters and thus during this phase the results produced a much longer decent phase and thus a larger error. However once the system was able to update the dynamics of the model, the RTMP was able to quickly correct for this fouling and recalculate a landing zone that was now better than 2 miles. In order to help clarify this major increase in accuracy after the RTMP had updated the dynamics, we decided to overlay the solutions that occurred during the ascent and descent phases on two separate maps so that the resulting clustering of solutions that occurred in each phase can more clearly be seen in Figure 105 and Figure 106.

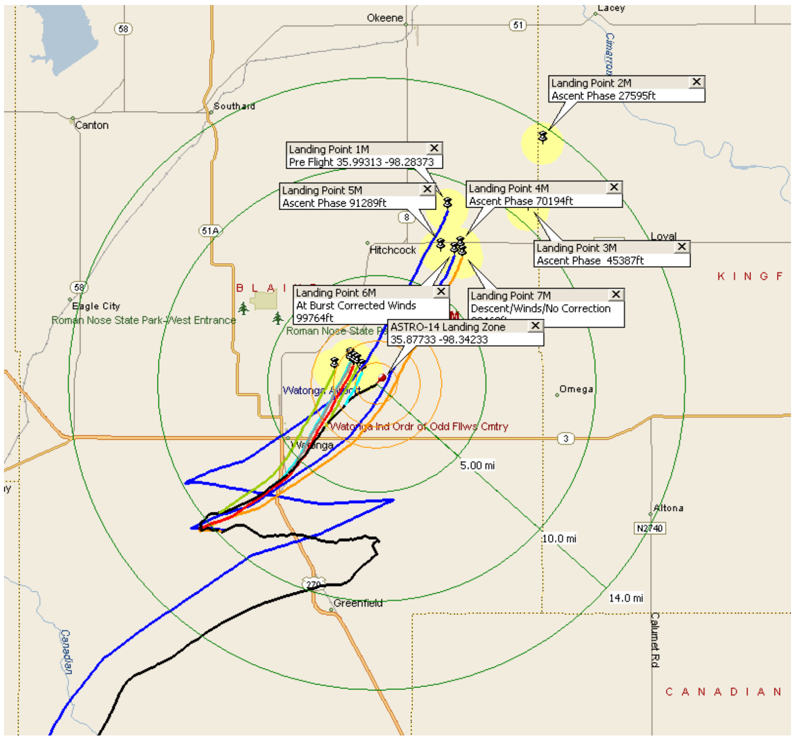


Figure 105 ASTRO-14 Ascent Phase Landing Zone Predictions

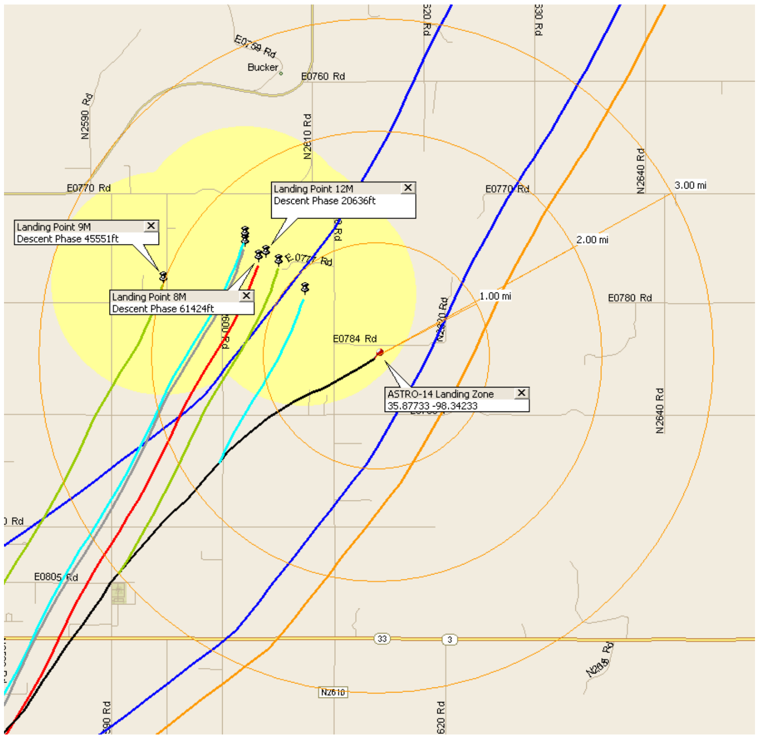


Figure 106 ASTRO-14 Descent Phase Landing Zone Predictions

GLOBAL POSITION SYSTEM

Since it is proposed that the majority of navigational information comes from use of the Global Positioning System (GPS), a brief introduction to how GPS works will be presented. GPS is based on 24 satellites circulating in 6 different orbits around the Earth. The system is operated by the US Department of Defense. The orbits and the number of satellites in each orbit have been chosen to assure that at least *5* satellites may be seen at any time from any location on Earth. Each satellite broadcasts its orbital parameter information at very low data rates. These parameters are used by the GPS receiver to predict the future position of the satellites. The measured range to the satellites in view is determined through use of standard triangulation between the receiver and at least three satellites that are within view. The accuracy of the solution is affected by several main factors.

The first factor that affects the precision of the solution is the geometry of the relative positions of the satellites that are used in the solution and the position of the receiver. The error that is produced by the relative positions of the satellites and receiver is called geometric Dilution of Precision (DOP). DOP's main effect is to cause a smearing of the solution over a range surrounding the correct location, and as this effect grows larger, the accuracy decreases from under 1 meter to over 100 meters. If a receiver sees 4 satellites and all are arranged in the same area of the sky, then this leads to a "bad" geometry. In the worst case, no position determination is possible at all, when all distance determinations point to the same direction. Even if a position is determined, the errors in position may be up to ±100 – 150 m. If, on the other hand, the four satellites are well distributed over the whole sky the determined position will be much more accurate. To help explain why this happens, let's take a look at the following examples. First, let's assume the four satellites are positioned in the sky at 90° steps. The distances between the satellites and the receiver can then be measured in four different directions, reflecting "good" satellite geometry. Good geometrical alignment of satellites is defined when from the view of the receiver the satellites can be seen in an angle of approximately 90° to each other. The point of intersection of the four circles is a rather small and therefore the calculated position will be rather accurate. However, if the satellites are more or less positioned inline from the view of the receiver, the plane of intersection of possible positions is considerably larger and elongated and leads to the determination of a position that is far less accurate. Examples of both high and low DOP are shown.

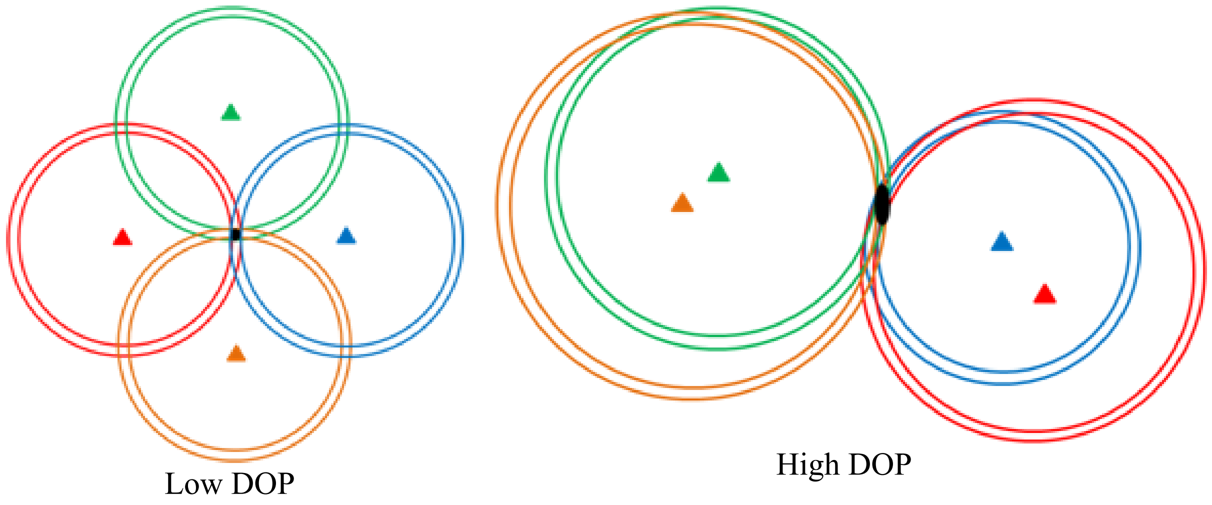


Figure 107 Geometric Dilution of Precision

DOP values are reported in three types of measurements: horizontal, vertical, and mean.

Horizontal DOP (HDOP) measures DOP as it relates to latitude and longitude; vertical DOP (VDOP) measures precision as it relates to altitude; and, Mean DOP, also known as Position DOP (PDOP), gives an overall rating of precision for latitude, longitude and altitude. Each DOP value is reported as a number between 0.5 and 99.5, where 99.5 represents very poor precision and 0.5 represents ideal accuracy. In order to determine the effect of DOP on the solution the following equation is used. Figure 108 shows the effects of PDOP on a location at the 50 yard line on Lewis Field.

$$DOP_{Error} = Accuracy\ of\ GPS\ Receiver * DOP \qquad [69]$$

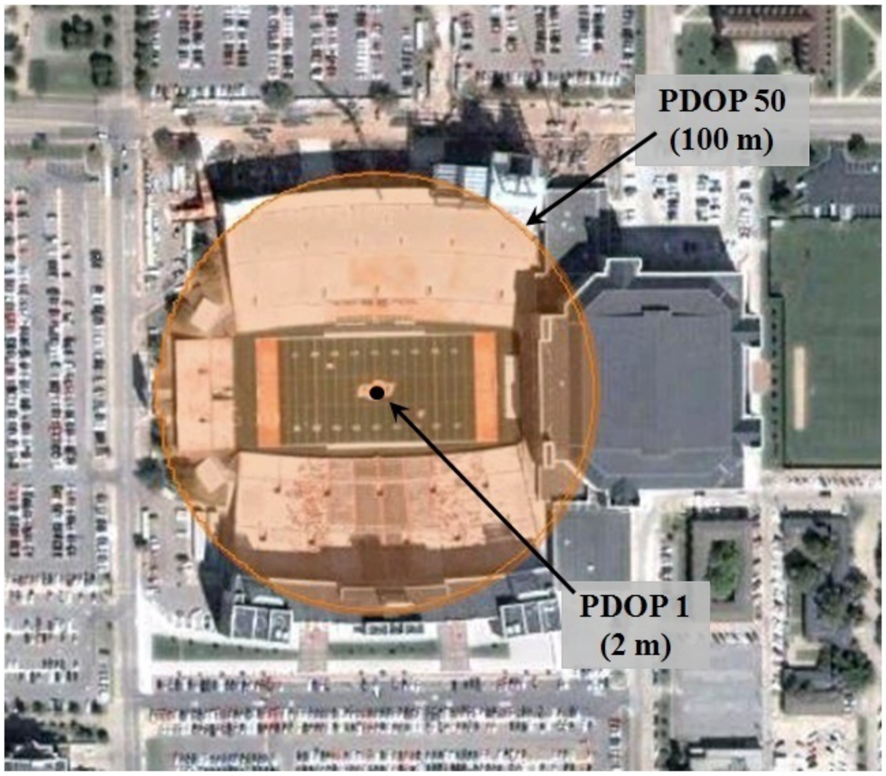


Figure 108 Effects of PDOP on Solution

One other major source of inaccuracy is the reduced speed of propagation in the troposphere and ionosphere. While radio signals travel with the velocity of light in the outer space, their propagation in the ionosphere and troposphere is slower. In the ionosphere, between 80 and

400km, a large number of electrons and positive charged ions are formed by the ionizing force of the sun. The electrons and ions are concentrated in four conductive layers in the ionosphere and result in refraction of the electromagnetic waves from the satellites, resulting in an elongated runtime of the signals. Typically, most receivers are able to correct the signal for normal variations of the velocity while passing the ionosphere. Theses variations are taken into account for all calculations of positions. However, most receivers are not capable of correcting unforeseen runtime changes, for example the effects of major solar activity. The troposphere adds even more elongating to the runtime of electromagnetic waves by refraction. The reasons for the refraction are different concentrations of water vapor in the troposphere, caused by various weather conditions. The error caused by the troposphere is smaller than that of the ionosphere error, but cannot be eliminated by calculation.

In order for the receiver to correctly triangulate a signal, the GPS signals contain information about ephemeris errors, and about the rate of clock drift for the broadcasting satellite. Unfortunately, the data concerning ephemeris errors are not able to model exactly the true satellite motion or the exact rate of clock drift. The disparity in ephemeris data can introduce ±1-5 meters of positional error while clock drift disparity can introduce ±0-1.5 meters of error.

The last major issue that can affect the solution is known as multi-path error. Multi-path error is caused by the signals from the satellites being reflected off obstacles before reaching the receiver, which causes an increase overall path length. While multi-path error would be a problem that most ground based system would have to take into account, this project will not need to focus on this source of error as the vehicle will be well above most common sources of multi-path error within just a few minutes of the mission.

Solution of Typical GPS Position

Measurement of time, to under a micro-second, is critical in the equations that need to be solved to determine the receiver's position in 3D space. Each satellite in the GPS network that orbits the Earth is equipped with an atomic clock so that they know exactly when they send a signal. Unfortunately these clocks are very large and expensive. Thus it is not feasible to embed these into every receiver on the market. So a compromise was made to include more inexpensive clocks into the receivers, but instead of treating time as a known in the GPS equations, it becomes a variable. With time now among one of the many unknowns that has to be solved for, a fourth satellite is now required to reach a valid solution for the receiver's 3D position.

Once the data is collected from the satellites, a set of four simultaneous equations are solved for four unknowns. The unknowns are positions (X, Y, and Z) and time of the receiver. It should be noted at this time, if the receiver is only able to obtain a solution using three satellites, then it will produce a 2D solution in which altitude (Z) is held constant and therefore not solved for.

If the receiver is able to receive a valid signal from more than four satellites, proprietary methods are employed by the receiver to determine which combination of four satellites will result in the best solution. The best solution that is typically used is the solution that results in the lowest DOP. Normally the solution obtained, without outside augmentation, is ±10 meters at best in both the horizontal and vertical directions.

Wide Area Augmentation System

Beginning in 1994, Wide Area Augmentation System (WAAS) was put forth by the FAA in attempt to increase the accuracy of GPS for aviation purposes. WAAS utilizes 25 ground stations, which detect and send GPS error information to a master control site. The master control site uses this information to compute a correction factor that can then be used by a WAAS enabled GPS receiver to improve it accuracy from ~10m to less than 5m. Actual performance measurements of system at locations throughout most of the contiguous United States and large parts of Canada and Alaska, have shown that accuracy is typically better than 1.0 meters laterally and 1.5 meters vertically. For example, in 2008 the station located in Oklahoma City, Oklahoma reported accuracies to within 0.634 and 0.993 meters in horizontal and vertical respectively (7).

The correction factors that are taken into account by WAAS are:

1. Ionosphere and Troposphere delays
2. Integrity information
3. Short-term and long term satellite clock errors
4. Short-term satellite position error (Ephemeris)
5. Long term satellite position error (Almanac)

Once the correction factors have been determined, information is relayed to WAAS geosynchronous Inmarsat satellites from the master control stations and is re-broadcast to the receivers as a grid of corrections. A GPS receiver uses this information grid to interpolate the proper Ionosphere correction based on its position within the grid. The "extrapolation" of this information outside the WAAS coverage is less and less precise to the point of inducing errors in the now 'corrected' signal. Interesting enough, the remaining correction factors are not location dependant and as such are not dependant on the receiver's location within the grid. The WAAS correction information is different than Radio Technical Commission for Maritime Services (RTCM) corrections transmitted by the Coast Guard for uses in DGPS because WAAS decomposes the errors into their primary elements (atmospheric, clock, and ephemeris). RTCM, on the other hand, broadcasts pseudo range corrections which are the sum of all error sources as observed by the RTCM reference station. This information is only valid relatively close to the reference station, where as WAAS is useful over a wide area.

NMEA Messages

The signals or messages that are sent from the satellites are defined by the National Marine Electronics Association (NMEA). This group has defined standards for just about every possible device used for navigation and instrumentation. In fact, the standards defined for use with GPS actually define not one, but many different messages designed to provide every possible piece of useful information. Examples of the NMEA sentences that are typically outputted are the following:

```
$GPRMC,153719,A,3559.053,N,09701.304,W,013.5,358.8,280408,004.9,E*66
$GPRMB,A,,,,,,,,,,,,V*71
$GPGGA,153719,3559.053,N,09701.304,W,1,06,1.8,270.1,M,-26.3,M,,*7D
$GPGSA,A,3,,,10,15,,18,21,,,26,29,,2.1,1.8,1.0*39
$GPGSV,3,1,11,06,27,310,39,09,02,149,00,10,12,056,35,15,59,086,38*7D
$GPGSV,3,2,11,16,08,295,00,18,49,263,50,21,55,320,33,22,11,248,00*73
$PGRMZ,888,f,3*13
$PGRMM,WGS 84*06
$GPBOD,,T,,M,,*47
$GPRTE,0,1,c,*36
```

Figure 109 Raw GPS Sentences

Decoding of NMEA sentences

The most important NMEA sentences include the GPRMC, which provides the minimum GPS sentence information required for navigation, and GPGGA, which provides the current 3-D fix data.

The GPRMC sentence, also known as the "Recommended Minimum" sentence, is the most common sentence transmitted by GPS devices. This one sentence contains nearly everything a GPS application needs: latitude, longitude, speed, bearing, satellite-derived time, fix status and magnetic variation. The only major piece of information that is missing from this message format is altitude. The major advantage of this sentence over others is that the fields within the sentence remain the same size for a given receiver. Figure 110 shows how the GPRMC sentence can be parsed as well as the size of each field.

$GPRMC,153719,A,3559.053,N,09701.304,W,013.5,358.8,280408,004.9,E*66		
Name	Example Data	Description
Sentence Identifier	$ GPRMC	Global Positioning Recommended Minimum
Time	153719	15:37:19 Zulu (hhmmss)
Solution: - V = Invalid - A = Valid	A	Solution is Valid
Latitude	3559.053, N	35º59.053' N (ddmm.mmm)
Longitude	09701.304,W	97 º 01.304' W (dddmm.mmm)
Speed over Ground	013.5	13.5 knots (xxx.x)
Course over Ground	358.8	358.8 º (xxx.x)
UTC Date	280408	04/28/2008 (ddmmyy)
Magnetic Variation	004.9,E	4.9 º E (ddd.d)
Checksum	*66	Used by program to check for transmission errors

Figure 110 Global Positioning Recommended Minimum

There are two major problems with the Recommended Minimum sentence: First, this sentence does not contain any information on the current altitude, and second, there is no information on the accuracy of the solution. Both problems can be overcome by reading in additional sentences. On most Garmin GPS receivers there are proprietary sentence that contain more information on the status of the receiver. One of these proprietary sentences $PGRMZ, contains the current altitude information. For example; ***$PGRMZ,888,f,3*13***, indicates an altitude of 888 feet. The problem with lack of information on the position error can be resolved by reading in the proprietary sentence $PGRME. This sentence contains information on three different position errors: horizontal, vertical, and overall spherical error. For example; ***$PGRME,6.5,M,10.2,M,13.6,M*2A***, indicates an estimated horizontal position error of 6.5 meters, a vertical position error of 10.2 meters and an overall spherical error of 13.6 meters. One problem however, with both $PGRMZ and $PGRME, is that the sentences are not available on all GPS systems.

The global positioning system fix data sentence, GPGGA, contains some of the most critical pieces of information required for 3-D navigation. This sentence not only contains information on the receiver's current latitude, longitude, and altitude, but also contains information regarding the current horizontal dilution of precision. The current status of the HDOP can be used to determine the reliability of the reported latitude and longitude. One of the only problems with this sentence is that the fields are not of fixed length, instead each field is separated by a comma and therefore the entire message must be parsed. The other problem with this sentence is that it does not contain any information on the current magnetic variation of the area. Figure 111 shows how the GPGGA sentence is filled with each segment of data.

$GPGGA,153719,3559.053,N,09701.304,W,1,06,1.8,270.1,M,-26.3,M,,*7D		
Name	Example Data	Description
Sentence Identifier	$GPGGA	Global Positioning System Fix Data
Time	153719	15:37:19 Zulu
Latitude	3559.053, N	35º59.053' N
Longitude	09701.304,W	97 º 01.304' W
Fix Quality: - 0 = Invalid - 1 = GPS fix - 2 = DGPS fix/WAAS fix	1	Data is from a GPS fix
Number of Satellites	06	6 Satellites are in view
Horizontal Dilution of Precision (HDOP)	1.8	Relative accuracy of horizontal position
Altitude	270.1, M	270.1 meters above mean sea level
Height of geoid above WGS84 ellipsoid	-26.3, M	-26.3 meters
Time since last DGPS update	null	No last update
DGPS reference station id	null	No station id
Checksum	*7D	Used by program to check for transmission errors

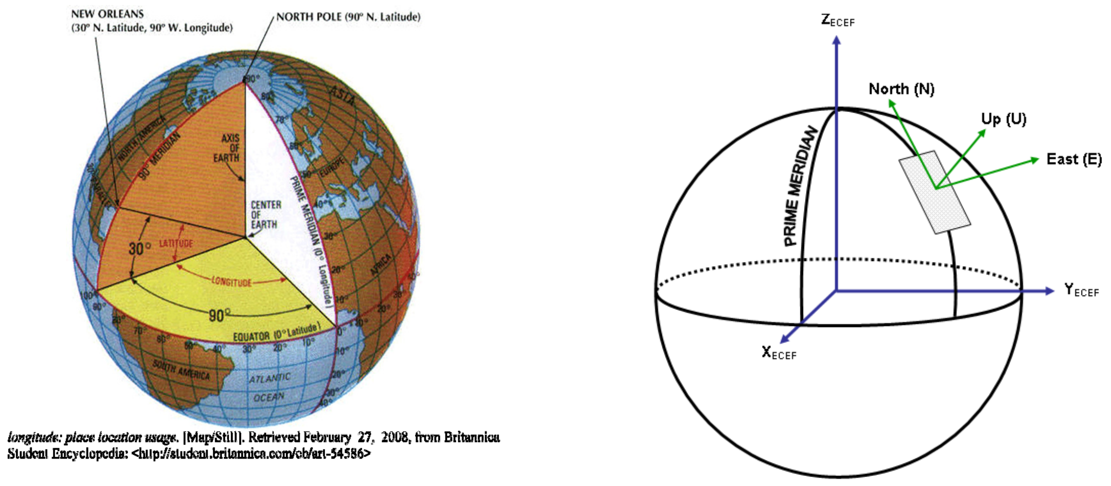
Figure 111 Global Positioning System Fix Data

Both the GPGGA and GPRMC are not readily useable alone as both are missing small pieces of information that might be required for full 3-D navigation. Once the current location of the system is found, one has to understand how to utilize this information to travel to a designated location or waypoint. In order to fully understand how to create this path, an understanding of the different coordinate systems used for navigation on the Earth is required.

COORDINATE SYSTEMS

Most path-planning navigation algorithms are designed in the local tangential frame with axes north, east, and up. However, long range navigation cannot be readily done using just the local tangential frame. It is necessary to perform a coordinate transformation from the local tangential frame to other frames, since various sensors provide information in different coordinate frames. Figure 112 shows the various coordinate frames involved in this project.

The GPS system provides the systems current location information in the Geodetic coordinate frame. In the Geodetic frame, the Earth is represented in a polar coordinate system that is longitude (λ), latitude (ψ), and height (h) above the Geodetic surface. This system is more commonly known as the Longitude, Latitude, Altitude (LLA) coordinate system. In order to make calculations of range and heading between two points easier, a Cartesian coordinate system is used. This Cartesian coordinate system, when applied to the Earth, is called Earth Center Earth Fixed frame (ECEF). The ECEF axis system is fixed to the Earth so that the X and Y axes rotate around the Z axis at a rotational velocity that is equal to that of the Earth's rotational velocity Q.


NEW ORLEANS (30° N. Latitude, 90° W. Longitude)
NORTH POLE (90° N. Latitude)
NORTH AMERICA
90° MERIDIAN
AXIS OF EARTH
CENTER OF EARTH
PRIME MERIDIAN (0° Longitude)
ASIA
EUROPE
AFRICA
30°
LATITUDE
LONGITUDE
90°
EQUATOR (0° Latitude)
SOUTH AMERICA
ATLANTIC OCEAN
longitude: place location usage. [Map/Still]. Retrieved February 27, 2008, from Britannica Student Encyclopedia: <http://student.britannica.com/eb/art-54586>
Z_ECEF
North (N)
Up (U)
East (E)
PRIME MERIDIAN
Y_ECEF
X_ECEF

a) Geodetic Coordinate Systems (LLA) **b) ECEF and Local Tangential Coordinate Systems**

Figure 112: Major Coordinate Systems for Navigation on the Earth

Coordinate Systems Transformations

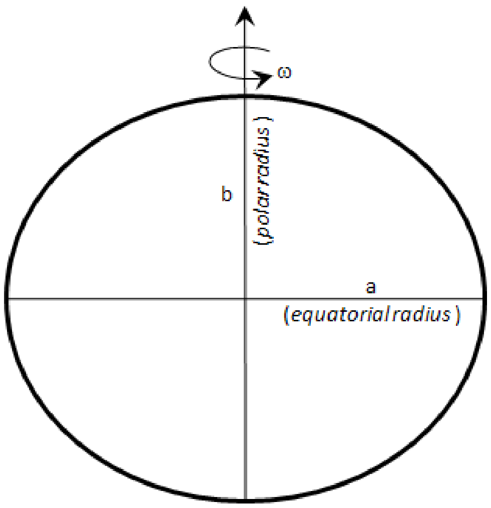
The conversion from the geodetic coordinate system (LLA) to ECEF is relative, straight forward. The basis behind this conversion is that the LLA system is based a spherical coordinate system while the ECEF is based on a Cartesian coordinate system. The conversion from LLA to ECEF uses equations [70] through [73], along with parameters from WGS84.

[70]

[71]

[72]

[73]



WGS84 Parameters		
Parameter	Description	Value
ω	Rotation Rate	7.292115 x 10^{-5} rad/s
a	Equatorial Radius	6,378,137 m
b	Polar Radius	6,356,752.3142 m
f	Flatting	3.35281066475 x10^{-3}
e^2	First eccentricity	6.69437999014x10^{-3}
e'^2	Second eccentricity	6.73949674228x10^{-3}

Figure 113 Reference Earth Model - WGS84

The conversion of ECEF coordinates back to the LLA coordinate system is a more involved process when compared to the conversion from LLA to ECEF. While a number of techniques have been put forth, the most accurate according to Zhu (8), is the following 15 step procedure.

$$\mathbf{r = \sqrt{X^2 + Y^2}} \quad [74]$$

$$\mathbf{E^2 = a^2 - b^2} \quad [75]$$

$$\mathbf{F = 54b^2Z^2} \quad [76]$$

$$\mathbf{G = r^2 + (1 - e^2)Z^2 - e^2E^2} \quad [77]$$

$$\mathbf{C = \frac{e^4Fr^2}{G^3}} \quad [78]$$

$$\mathbf{S = \sqrt[3]{1 + C + \sqrt{C^2 + 2C}}} \quad [79]$$

$$\mathbf{P = \frac{F}{3\left(S + \frac{1}{S} + 1\right)^2 G^2}} \quad [80]$$

$$\mathbf{Q = \sqrt{1 + 2e^4P}} \quad [81]$$

$$\mathbf{r_0 = \frac{-(Pe^2r)}{1 + Q} + \sqrt{\frac{1}{2}a^2\left(1 + \frac{1}{Q}\right) - \frac{P(1 - e^2)}{Q(1 + Q)} - \frac{1}{2}Pr^2}} \quad [82]$$

$$\mathbf{U = \sqrt{(r - e^2r_0)^2 + Z^2}} \quad [83]$$

$$\mathbf{V = \sqrt{(r - e^2r_0)^2 + (1 - e^2)Z^2}} \quad [84]$$

$$\mathbf{Z_0 = \frac{b^2Z}{aV}} \quad [85]$$

$$\mathbf{h = U\left(1 - \frac{b^2}{aV}\right)} \quad [86]$$

$$\mathbf{\psi = atan\left(\frac{Z + e'^2Z_0}{r}\right)} \quad [87]$$

$$\mathbf{\lambda = atan2(Y, X)} \quad [88]$$

Transformation from ECEF coordinates to the local tangent coordinates (NEU), also called the navigation frame, is performed with the following coordinate transformation matrix, it should be noted that the current longitude (λ) and latitude (ψ) are required in order to obtain a solution.

$$\begin{bmatrix} E \\ N \\ U \end{bmatrix} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\psi\cos\lambda & -\sin\psi\sin\lambda & \cos\psi \\ \cos\psi\cos\lambda & \cos\psi\sin\lambda & \sin\psi \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad [89]$$

To transform from local coordinates to the ECEF coordinate system is simply the inversion of the ECEF to NEU transformation:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -\sin\lambda & -\sin\psi\cos\lambda & \cos\psi\cos\lambda \\ \cos\lambda & -\sin\psi\sin\lambda & \cos\psi\sin\lambda \\ 0 & \cos\psi & \sin\psi \end{bmatrix} \begin{bmatrix} E \\ N \\ U \end{bmatrix} \qquad [90]$$

With these two matrix transformations, one is now able to fully transform from the geodetic coordinates (LLA) to local tangent coordinates (NEU). Most navigation schemes, as well as kinematic models, are based in the local tangent coordinate system. This transformation to a local x, y, and z system can easily be assigned to a traditional North, East, and Up system. Transformation from GPS information to the NEU system is a multi-step process. First, the current geodetic system is transferred to the ECEF system and then transformed into the NEU system. At this point, the current states of the system can be introduced to a kinematic model, and the resulting changes in state can be calculated. After the results of the kinematic model have been found, the states are transformed back to the ECEF system and are finally transformed to the LLA system. The results can then easily be displayed on a map of the world.

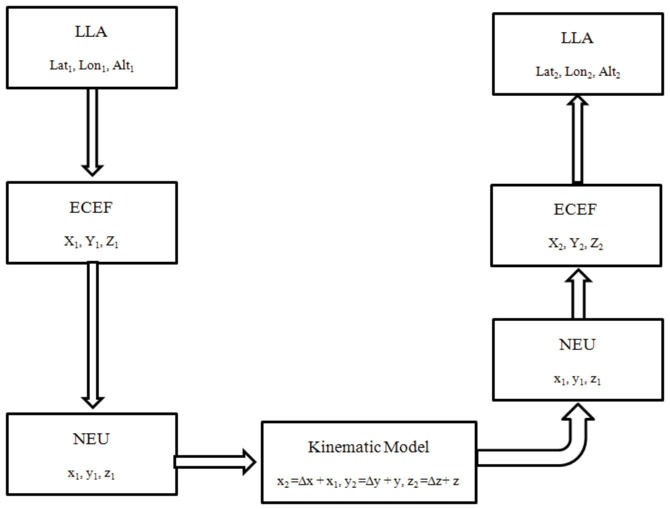


Figure 114 Conversion Process between LLA and NEU

LITERATURE REVIEW

One of the earliest attempts to control a small scale parafoil was presented in 1995 by Smith (9). In this paper the author shows the development of a 6-DOF computer simulation of a small parafoil. In this model, the aerodynamics were simplified, but still contained both the non-linearity's and severe constraints that are common to parachutes. The control algorithms put forth used position and velocity provided by a GPS to generate commands at a rate of 1 Hz, to control the steering lines of the chute. The author then goes on to discuss the development of two different modes that parafoils can be commanded to perform, homing and circling. The homing mode was included to address the problem of moving the parafoil in a favorable direction that would allow it to move toward a target point, while allowing for an offset from this point. The homing mode was based on a two-axis proportional navigation and also on an upward-biased rate-integrating navigation in the glide angle.

Once the parafoil reached a predetermined offset from the target, it would enter into the circling mode and then attempt to maintain an offset circling of the target until either the winds pushed it off target, at which time it would switch back to the homing mode, or if the parafoil was near the ground it would produced a landing flare to kill off all remaining lift and attempt to produce a minimal velocity on landing.

One interesting use of a parafoil was proposed to deliver airborne cargo, precisely and autonomous (10). This paper presents a high-fidelity modeling and simulation of a powered parafoil-payload system and how the system could be applied to an autonomous precision airborne cargo delivery. In the proposed concept, the cargo transfer is accomplished in two phases: Initial towing phase, when the glider follows the towing vessel in a passive lift mode;and the autonomous gliding phase, when the system is guided to the desired point. During the towing phase, the system gains as much altitude as possible by taking the angle-of-attack that will provide the best lift. Once sufficient altitude is attained, the gliding phase starts. The system is steered to the desired location by controlling the lengths of the rear suspension lines using two control inputs. The paper presents the concept of the system, its 6DoF model, the control algorithm at the stage of passive glide and the simulation results.

In a paper presented by Hogue and Jex (11), the authors discussed a novel approach to autonomous control of a parafoil. The main objective of their approach was to employ the use of pseudo-jumper guidance and control scheme, which was based in part on professional parafoil jumpers. After surveying professional jumpers, the authors where able to categorize a typical parafoil mission into five distinct phases, which were defined as the following:

- "Cruise" phase: The cruise phase consisted of aiming for a landing zone, designated by several waypoints, by use of one or more homing algorithms.
- "Hold" phase: As the parafoil nears the landing zone, it switches to a mode that commands S-turns just downwind of the approach zone. The pattern is adjusted to according to the winds at the site so that the parafoil stays near the landing zone and also has enough room remaining to perform a final approach and flare at landing.
- "Center" phase: As the parafoil nears the ground the controller aims the parafoil for the center of the landing zone.
- "Approach" phase: As the parafoil gets even closer to the ground, the control switches from trying to align with the center of the landing zone and instead aligns itself with the surface winds to minimize control line actions.
- "Flare" phase: The final phase of the missions is triggered with a rapid full braking to arrest the vertical velocities of the system just before touchdown.

While this project will develop a controller to return the parafoil from near space conditions, over 80,000 ft, there have been many attempts at implementing parafoil controllers from lower altitudes, under 25,000 ft. One example of this implementation was an autonomous parafoil that has been tested in the field at the U.S. Army Yuma Proving Grounds (12). The results were obtained from airdropping the system from an aircraft at an altitude of 10,000 AGL and then allowing the onboard flight control to fly autonomously until it reached approximately 1500 AGL. At this time, an operator on the ground assumed control and manually landed the system. One of the more unique topics addressed by this paper was the establishment of several safety zones, called ‘safety fans’, around the landing zone. The safety fans that are established are actually three zones that are composed of the three different modes of failure that the parafoil system might encounter. The three modes of failure that are discussed are: Ballistic, Flight Termination System (FTS), and Uncontrolled Flight. The Ballistic Safety Fan is based upon a failure, where the parafoil does not correctly deploy, and the system descents under a drogue chute with minimal drag. This mode of failure will result in the smallest safety fan but also the highest impact velocity. The second mode of failure discussed, the Flight Termination System, is designed to be activated by the flight computer in the event of failure of the guidance system. When FTS is activated, the controller commands a tight right-hand turn until either the guidance system recovers, or the system impacts the ground. With knowledge of the turn rate of the parafoil and a local wind profile, the size and location of the FTS safety fan can be found. Finally, the last mode of failure, Uncontrolled Flight, is designed for situations, where the parafoil has deployed correctly but is no longer under commanded flight. In this failure mode, the parafoil is stuck in forward flight and as a result has the potential to travel extremely long distance before the system impacts on the ground.

An example of an extremely large delivery system was presented in references (13) and (14). In these papers, the status of several large parafoil projects from 2005 where discussed. The first test flown was a 10,000 pound-class parafoil system, which was then later extended to accommodate payloads up to 30,000 pounds. The initial avionics applied by the guidance controller software used GPS to obtain position, velocity, and heading data. Later versions where upgraded to not only utilize a GPS receiver to provide position and velocity, but also added the addition of inertial sensors to provide full six-axis inertial data to the controller. The papers go on to discuss the use of a third system that would utilize only GPS measurements and a single rate gyro on the yaw axis. The guidance algorithm is partitioned into homing, energy management, and an optimized table-lookup terminal flight phase. In the homing stage, the parafoil is commanded by use of simple heading rate commands so that the parafoil is pointing toward the landing zone. Once the parafoil is close to the landing zone, it switches to an energy management phase, where the guidance system commands the parafoil to fly in a figure-eight pattern that is centered over the landing zone. This phase of the flight is used to slow the forward velocity of the parafoil to a speed that is acceptable for landing. Once the parafoil's forward velocity has slowed down to an expectable rate, the controller switches to the final phase of landing. In this phase, the guidance system steers a trajectory that intersects the desired landing point at a heading, which is determined from current wind conditions at the site. This final landing phase is implemented by use of a table-lookup algorithm, which uses a large family of pre-computed heading rate commands. All the phases of the controller have been designed to accept and used the following system states: position, velocity, heading, heading rate, and a table of wind velocity as a function of altitude.

An autonomous parafoil that has been successful developed is the MMIST Sherpa (15) . The Sherpa, as of the time of this writing, is being used in Iraq to provide precision aerial re-supply to troops in Iraq. The system has a cargo capacity of nearly five tons and vertical standoff capability of up to 25,000 ft. Upon leaving the cargo aircraft, the Sherpa deploys the parafoil and then autonomously flies the cargo to the pre-programmed landing point and lands within a circular error probability (CEP) of 100 m (328 feet) of the target. To aid in the guidance and control of landing locations a handheld remote is utilized. This controller provides two main functions: Manual over-ride and in-flight landing point reprogramming. The manual control capability allows a user on the ground to over-ride the autonomous mission, if desired. In-flight reprogramming of the target point is quickly accomplished using the same handheld remote controller. The Sherpa can be reprogrammed to deliver cargo to the present location of the ground controller through a short communication signal from the handheld remote controller.

In a project presented by D'Souza in 1997, an optimal controller for planetary landing was presented (16). The guidance law presented attempts to minimize the commanded acceleration along with a weighted final time. This guidance law is a linear function of the states of the system with respect to the landing zone and a nonlinear function of the time-to-go. The time-to-go is obtained as a solution to a quadric equation that can be solved analytically. The advantage of this guidance law is that it does not involve any iteration to arrive at a solution. It is the exact solution to the two-point boundary value problem associated with the first variation necessary conditions, which also satisfies the second variation necessary conditions for a minimum.

Path-Planning

One of the most interesting aspects of this project is the development of a path-planning scheme that will allow the parafoil to be released from altitude and then autonomously travel to a pre-designated landing zone, which the RPV determines can be reached. This path-planning scheme will differ from what most do, to the nature of the environment that the system will be passing through. The RPV will start from a near-space altitude, over 80,000 ft, and then traverse to one of several pre-assigned landing locations. While the RPV advances toward the target, several major influences will be encountered, such as changes in air density and wind velocity as a function of altitude. In order to gain an understanding of the type of problems that might be encountered, several different path-planning schemes have been looked at.

One of the problems that will be faced during the design and implementation of a path planner for the parafoil will be that the winds aloft can change rapidly not only in magnitude but also direction. The problem then becomes how to successfully plan a path that will allow the vehicle to reach a designated landing zone as it descends through varying conditions. One paper of interest, (17) "*Strategies of Path-Planning for a UAV to Track a Ground Vehicle*" , does not directly address this issue of varying winds and density, it does address how to modify the path that is dependent upon the relationship between the velocity of a UAV and the velocity of ground target. In this paper, the authors put forth a path-planning scheme that would allow an unmanned aerial vehicle (UAV) to follow a ground vehicle. The ground vehicle was allowed to change its heading and vary its speed from a standstill up to the velocity of the UAV, while the UAV maintained a fixed airspeed and maneuvered itself to track the ground vehicle. In order to perform the required task, the scheme utilized a hybrid controller that allows the aircraft to switch from one mode of tracking to anther mode of tracking that is directly dependent upon the relative velocities of the two vehicles. The two modes that are put forth for this scheme are a loitering mode and a sinusoidal mode. The loitering mode is where the UAV would enter into a holding pattern above the ground vehicle of interest until the target leaves a fixed zone. If, however, the ground vehicle is moving at a higher speed, the UAV would enter the sinusoidal mode. In the sinusoidal mode, the UAV would stand-off at a given distance and then slew back and forth in a sinusoidal pattern with an amplitude that was directly a function of the relative velocity of the UAV and the ground vehicle. The path that the UAV follows in the sinusoidal mode is defined by a simple sin-wave that has an amplitude (A) that varies directly in relation to the ratio between the velocity of the UAV (v_p) and the velocity of the ground vehicle (v_b).

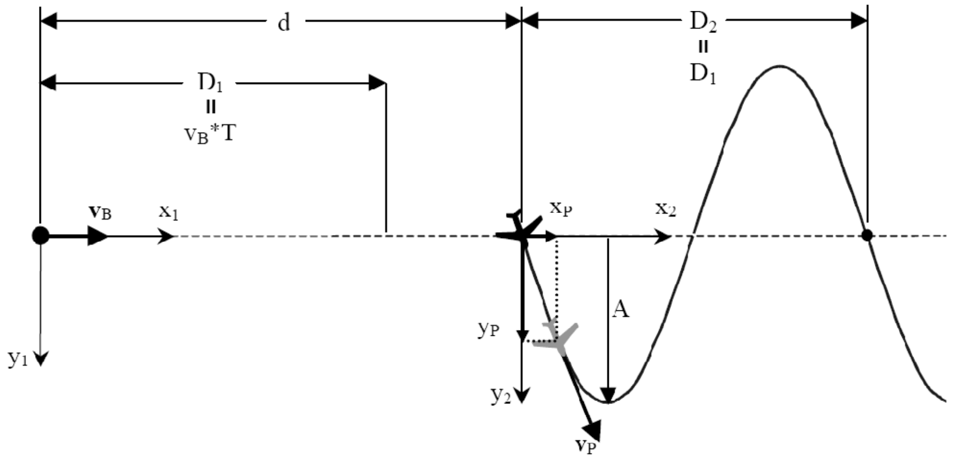


Figure 115 Path following Scheme for UAV tracking a Ground Vehicle

In the loitering mode, the UAV enters into a holding pattern at a reassigned location that is in direct relation to the ground vehicle. The UAV can enter and hold this location by either entering into a circle or rosette trajectory, see Figure 116 for the trajectory that is sketched out by each of the loitering modes. In the circular trajectory, the UAV simply orbits a waypoint that places the UAV in the desired location and then after entering this orbit, maintains a constant bank angle to circle around a waypoint. One of the other methods presented for entering a holding pattern above the ground vehicle is a rosette trajectory. In the rosette trajectory, the UAV is given several waypoints that define a line, which passes over the target. Once the UAV has completed this set of waypoints, a new line segment is generated by rotating the previous set of waypoints, or line segment, about the target and then recomputing a new set of waypoints. One of the advantages of the rosette over circular orbiting is that in the rosette trajectory the UAV is able to maintain an attitude that allows the bottom of the UAV to remain facing the ground for longer time periods, when compared the aircraft remaining in a constant bank angle.

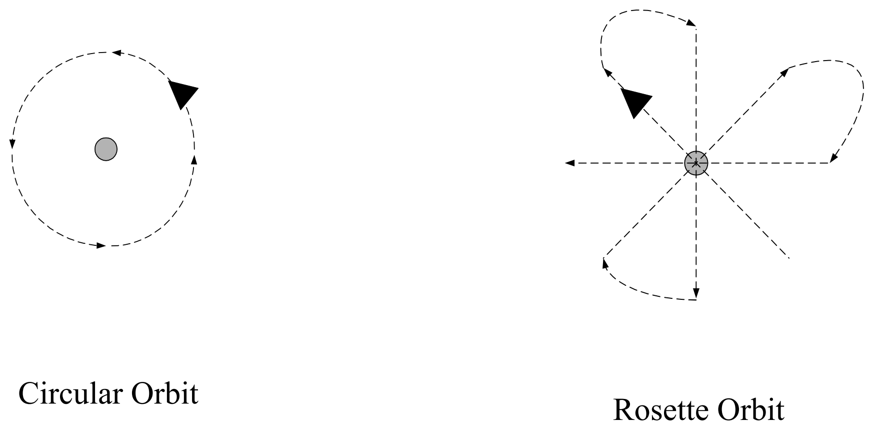


Figure 116 Loitering modes

A second path planning scheme originally designed for a UAV was put forth in reference (18). The main propose of this scheme was to design a relatively simple controller that would allow lateral tracking between waypoints for the Aerosonde as it traveled across the Atlantic Ocean. While the main purpose of this paper was to discuss a lateral track controller that can be used by a UAV, the overall idea can easily be modified for use by a parafoil as the controller was based on a kinematic model of the Aerosonde UAV. The paper showed that the controller was able to stably track a flight plan segment from any initial condition, while not relying on complex switching logic as other lateral controls do. The author also showed that the controller could be slightly modified to handle high wind situations. While the objective of this scheme was to develop a controller that would allow the UAV to intercept a track line between to waypoints from an arbitrary position and heading relative to the track line, the controller can be modified for use by a parafoil to maintain a desired heading which favors traveling toward the landing zone.

In a paper presented by authors in reference (19), a path finding routine was introduced that can be used to create an optimal trajectory for a given area. While the core of the paper was developed for use by a UAV, the concepts presented can be transferred to an autonomous parafoil. In this paper one formulation was the utilization of local tangent plane equations of motion, and the other was, use of simplified equations of motion. In addition, the effects of wind, moving threats and a moving target are added. The moving threat could consist of a known object that should be avoided during flight, as in a lake or large wooded area. The moving target would be a moving destination, or a rendezvous problem. The last addition to the problem was an examination of a pop-up threat that causes a reformation of the optimal path in mid-flight. In order to gain more realistic terrain, information from the United States

Geological Survey was incorporated into the model. The data obtained from USGS can be found in tabular format relating the altitude to the locations longitude and latitude. This data was then converted to matrix form, from which it could then be used as $f(x, y)$.

In a paper presented by Rafi (20), an algorithm for path generation for an UAV following a moving target is addressed. The UAV in consideration is a fixed wing aircraft that has physical constraints on airspeed and maneuverability. The author shows that a simple circular intercept pattern navigation algorithm works extremely well under most all conditions encountered during simulation. One of the other interesting aspects of this paper is that the tracking of the target is done using a mounted camera. The paper discusses how to utilize the information obtained from the camera in order to transform the location of the target in the frame of the camera, into coordinates that are with respect to the aircraft.

The focus of the work presented by Mcgee (21) was the exploration of a method for finding the shortest path from an initial position, orientation to a final position, and orientation in the two-dimensional plane for an aircraft with a bound turning radius in the presence of a constant wind. This work was based on a small autonomous aircraft under the combined control of an off-the-shelf Piccolo avionics package for low level control, and an onboard PC104 computer for higher level tasks. The primary interest in the paper, with respect to this project, was the development of the path planner. The path planner generated an overall path that was then broken down into several smaller segments. The path segments are generally composed of a mixture of the following: right and left turns with minimum radius of curvature that are composed of an arc length less then 180 degrees, right and left turns composed of an arc length greater then 180 degrees, and then finally, straight line segments. These paths are then combined into a single path that is composed of any combination of any three path segments discussed above. The path finding routine then determines the combination that will result in the shortest path to the target point of interest. One of the major advantages of this type of path creation is that only very simple behaviors are required from the system. In general, the only behavior the vehicle would have to exhibit would be an ability to turn clockwise and counter-clockwise as well as travel in a straight line.

LOW COST AUTONOMOUS RECOVERY SYSTEM

One of goals of this project was to demonstrate the feasibility of using a parafoil as the basis of a low cost autonomous recovery system. Parafoil based recovery systems have several advantages over the traditional spherical chute for recovery of payloads for near-space environments. Some major advantages of the parafoil is that you are able to control the system in the horizontal plane, and thus control heading, as well as the parafoil is able to produce a forward driving force, which allows it to establish a forward velocity independent of the winds current speed. In order to demonstrate the feasibility of this idea, it was decided to create a system that utilized a simple waypoint seeker as the control. In order for this system to be utilized by near-space mission the controller had to meet the following requirements: low cost, light weight, low power consumption, the ability to control the parafoil in heading control with simple 'Pull-Left' and 'Pull-Right' commands, and be able to determine its current location and velocity with respect to the ground.

Autopilot

The controller that was selected for this project is the ArduPilot, designed by Chris Anderson and Jordi Muñoz of DIY Drones built around the new ATMega328 microcontroller. This controller is based on a 16MHz Atmega328 processor, which has a total onboard processing power of approximately 24 MIPS. To decrease the overhead on the core processor, the unit contains the following: a hardware-driven servo controller, which means less processor overhead and increases response of servos and a built-in hardware failsafe that uses a separate circuit with a multiplexer chip and ATTiny processor, that can be used as a failsafe to allow servo commands to still be passed to the system in the event of a processor error. This failsafe allows the controller to be rebooted during a mission if needed, without losing servo commands during the reboot. The unit has been designed to be open source-code and thus allows full access to customize the software as needed.

Atmel Atmega328 processor

The autopilot system is centered on the Atmel Atmega328 microcontroller, which provides the following features (22): 32K bytes of In-System Programmable Flash with Read-While-Write capabilities, 1K bytes of EEPROM, 2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters, two 8-bit and one 16-bit, with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, an 8-channel 10-bit ADC, a programmable Watchdog timer with internal oscillator, and six software selectable power saving modes: idle, ADC noise reduction, power save, power down, standby, and extended standby. The idle mode stops the CPU, while allowing the SRAM, timer/counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The ADC noise reduction mode stops the CPU and all I/O modules, except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In power saving mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. Power down mode saves the register contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset. In standby mode, the oscillator is kept running while the rest of the device is sleeping. Extended standby mode is identical to the power savings mode with the exception that oscillator is kept running. Both standby modes allows very fast start-up, as the device wakes in six clock cycles, and produce very low power consumption on the order of 0.1 μA. The Atmega328 is rated for a large operating voltage range from 1.8 to 5.5 volts with power consumption 0.2mA in active mode, 0.75 μA in power saving mode. This along with its operating temperature range from -40° to 85°C and operating weight of 25 grams makes it a perfect match for near-space environment missions.

Programming

The ATmega328 utilizes a boot loader to provide direct access to the Read-While-Write Self-Programming mechanism. This feature allows flexible access to provide quick software updates, as well as the ability to read code and write program code into the device. The boot loader can thus even modify itself, and it can also erase itself from the code, if the feature is not needed anymore.

The writing of the programs is done performed in the open-source Arduino environment. This environment allows the user to write programs in a 'C' type format and thus making it easy to write and implement code for use on the ATmega328 board. To interface with the microcontroller, a Universal Serial Bus device, from Future Technology Devices International (FTDI), is used. This device allows rapid deployment of code, as well as providing a serial interface for debugging options. One of the other advantages of this USB interface is that provides direct access to the auto-reset on ATmega328 when a new program is downloaded.

Waypoint Navigation Scheme

The simplest approach to waypoint navigation is to calculate a bearing, which points directly toward the target waypoint from the vehicles' current position.

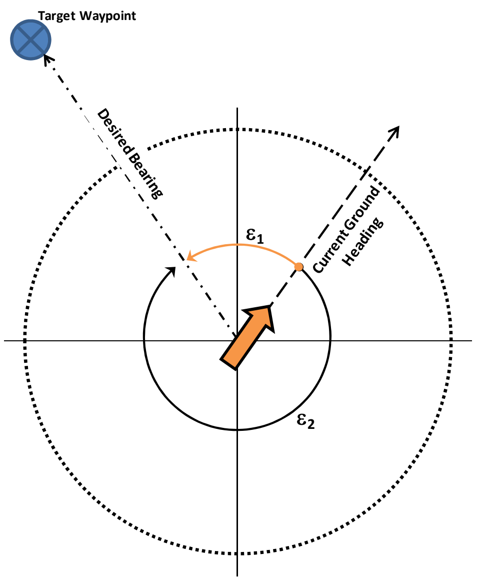


Figure 117 Path Planning Scheme

Since it safe to assume that the vehicle will have access to its current location and the location of the desired waypoint, the autopilot can compute the desired bearing to the target by using the FAI bearing calculation as follows:

$$\varphi_{bearing} = atan\left[\frac{cos(lat_0) * sin(lat_1) - sin(lat_0) * cos(lat_1) * cos(lat_1 - lat_0)}{sin(lon_1 - lon_0) * cos(lat_1)}\right] \quad [91]$$

Once the bearing is know, the difference between the current ground heading and desired bearing is found by:

$$\epsilon_{\varphi} = \varphi_{bearing} - \varphi_{heading_{GND}} \quad [92]$$

The heading error indicates the number of degrees the system would need to turn in order to correct its current heading to that of the desired bearing. Unfortunately, this solution does not guarantee that the vehicle with close the error by turning in direction of minimum angle. To determine the minimum angle for the vehicle to turn the avoid this the error is correct by either, subtracting 360 degrees from the heading error if it is greater than 180 degrees or adding 360 degrees to the heading error if it is less than -180 degrees. A negative heading error will produce command a left turn to correct the vehicles, and a positive heading error produces a right turn.

One problem with this method is that as the distance to the target becomes smaller, very large changes in heading error can occur for a small offset in position, with respect to the target. A simple and robust correction involves defining a circle of specified radius around the waypoint. As soon as the vehicle enters this circle, its behavior can be modified to switch from a waypoint seeking mode to a circular orbit.

Typically the commanded turn angle and the heading error cannot be one in the same. There are several reasons for this: The system may not be able to physically be commanded to turn such a tight angle, and second, even if the system can command a large turn rate, the system will start to overshoot. In order to determine the steering angle of the vehicle, a simple proportional gain controller is used. This controller takes on the form:

$$\textbf{Steering Angle} = \boldsymbol{\epsilon_{\varphi} * K_P} \qquad [93]$$

However the steering command may still need to be limited to:

$$\textbf{Steering Angle} = \begin{bmatrix} \geq 10^{\circ} \text{ then Steering Angle} = 10 \\ \leq 10^{\circ} \text{ then Steering Angle} = -10 \\ \text{else Steering Angle} \end{bmatrix} \qquad [94]$$

Once a steering angle was found, a servo command was generated by the code to drive the overboard servo hardware found on the ArduPilot. The following is the basic pseudo code used to create the autopilot:

```
1. Initialize the system:
    a. During this stage the onboard microcontroller checks the status of the GPS
       and waits until a lock is obtained.
    b. Determine 'Return to Home' location
    c. Load Waypoint(s) into active memory
2. Wait until Autopilot start is given
3. Decode GPS message to determine current states
4. Determine Turn Angle
    a. Determine Bearing to current waypoint
    b. Determine Current Heading of system
    c. Determine Error between Bearing and Heading
    d. Calculate servo turn angle
5. Send servo angle command to controller hardware
6. Await update of GPS status
7. Repeat Steps 2 through 6
```

Hardware Implementation

For initial debugging of the system, we decided to implement the autopilot on a ground vehicle. The ground vehicle chosen was Losi's HIGHroller Lifted Truck. This vehicle was selected for several reasons: The ground vehicle allowed safe testing and debugging of the autopilot hardware and software. The ground system could be tested regardless of wind conditions. The vehicle was large enough to support all the gear required for the autopilot, as well as external data logging devices. And the vehicle turned by use of a single servo with ‘Turn_right’ and ‘Turn_left’ commands.

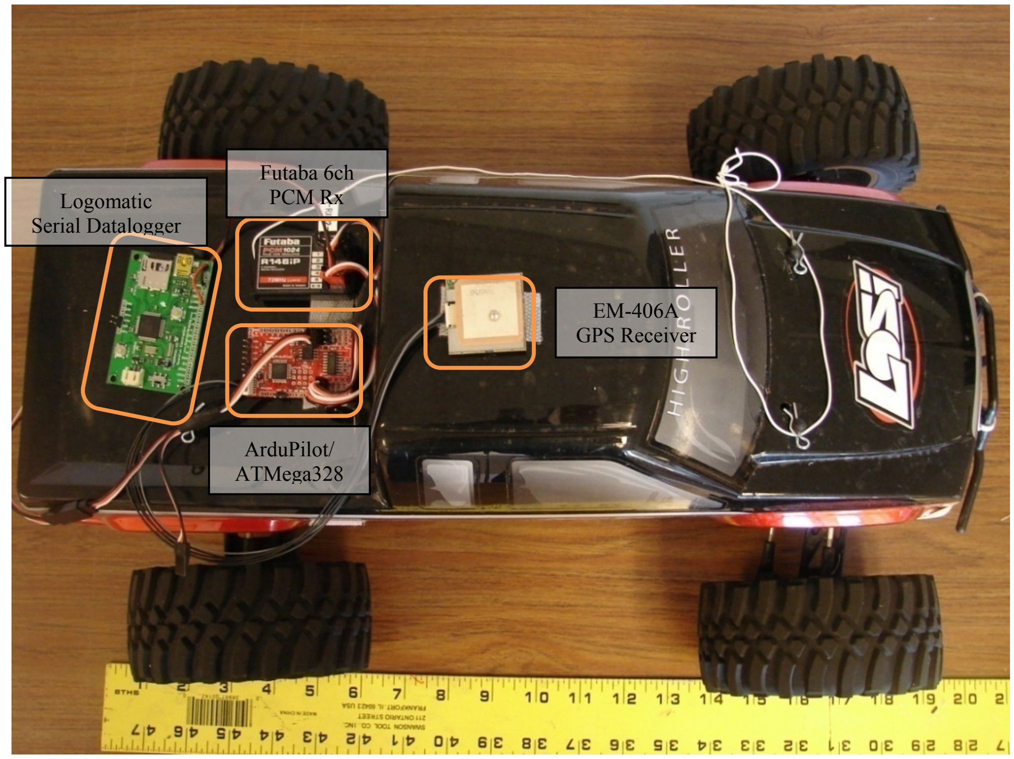


Figure 118 ParaCar Autopilot Testbed

In order to test the autopilot, it was decided to start with a very small gain of 0.1 and then observe the ground track to determine the systems behavior. Before the test was initiated, the vehicle was assigned a home waypoint (36.132332°N, 97.081171°W) and then moved to a start location 150 feet to the southeast, and given a heading of 110°. After placement at this location, the autopilot was started, and the vehicle was given a small forward velocity, approximately 5 mph, and then allowed to autonomous navigate its way back to the home target. Once the vehicle reached to within a 2 meter radius of the target, the logger was stopped and the data from the vehicle was processed. This process was then repeated for a total of six gains ranging from 0.1 to 0.6. We decided to plot the results of three of the gains to show the effects of the gain on the tracking of the system to the assigned home. The results for gains 0.1, 0.3, and 0.6 were selected as they demonstrated the three basic cases of tracking a single point with only a proportional controller. The resulting ground tracks were overlaid on an aerial map of the field using the website GPS Visualizer and are shown in Figure 119. The resulting behavior of each of the gains indicates how the vehicles search for the waypoint is directly affected by its gain setting.

The ground track for the 0.1 gain shows a tendency for the vehicle to ‘spiral’ in toward the target. If one follows the ground track, they will note that for each pass of the target the distance from the target decreased. The overall time for the ‘Para-Car’ to reach the target was well over 440 seconds; and on the first circuit around the target, the maximum error was on the order of 150 feet. The gains of 0.3 and 0.6 both dramatically improved the time to target as both were able to reach the target in just over 60 seconds. The major difference between the two gains is that the 0.6 gain resulting in some over correction to the target, which can be seen in the serpentine ground track. Since gains below 0.3 tended to produce a spiral ground track and gains above it serpentine, it was decided to use the gain of 0.3 for the parafoil runs.

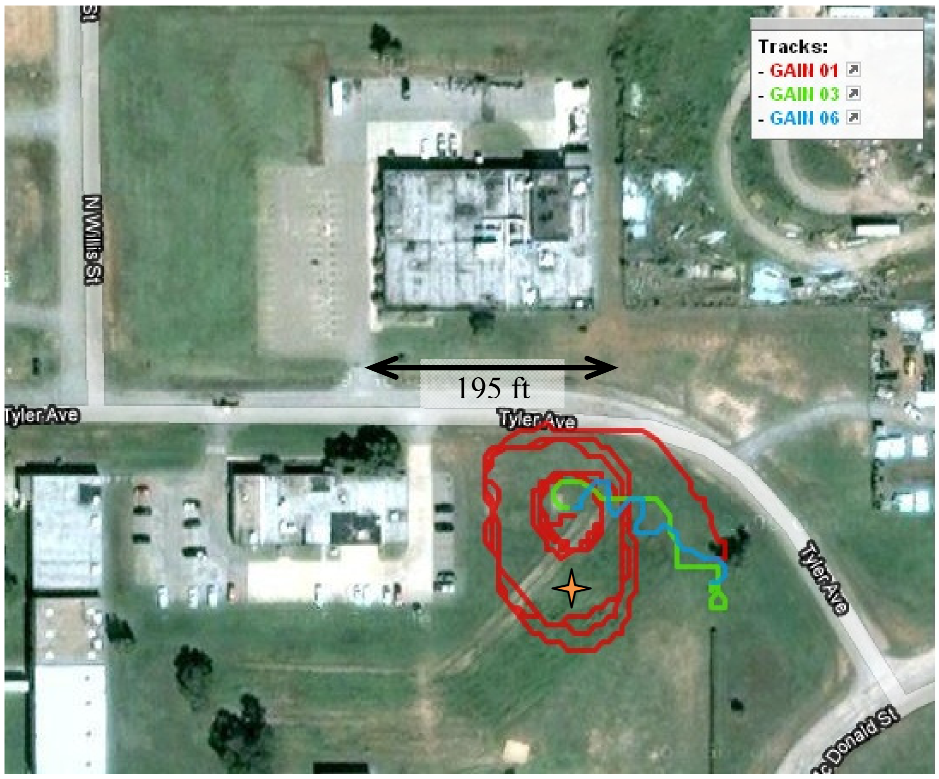


Figure 119 ParaCar Autopilot Ground Tests with Varying Gains

Now that the system had been tested safely on the ground, we decided to move to the next stage of testing the autopilot on board a powered parafoil. The parafoil selected was a COTS powered parafoil produced by Gold Rosita. This parafoil has the following dimensions: a span of 50 inches, a chord of 18 inches, an aspect ratio of 2.78, and a flying weight of 0.68 lbs. The system selected is actually a powered parafoil, but this selection of powered does not affect the behavior of the system in heading. The only advantage the powered system has over a non-powered parafoil is that the system can be hand launched and then use power to gain altitude. In a near-space mission, the balloon would actually provide the system with its initial altitude and thus would necessitate the need for power.

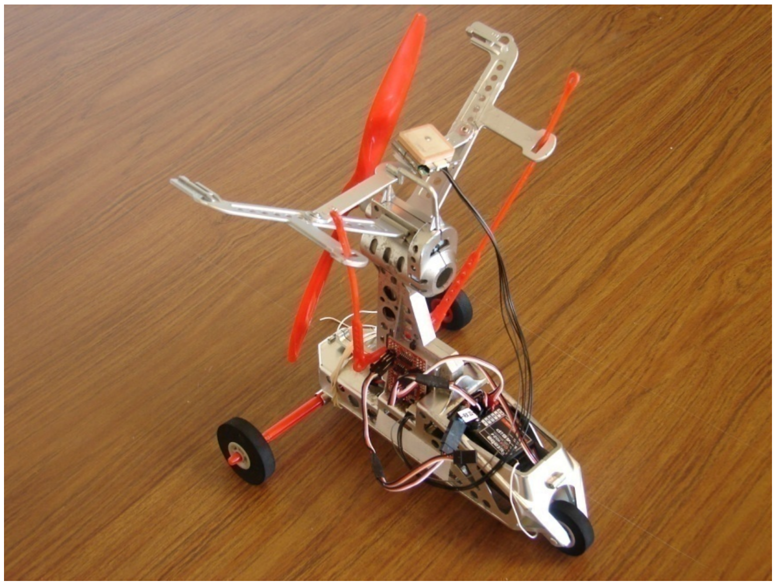


Figure 120 Parafoil Gondola Testbed

Parafoil in Flight

Once the parafoil had been selected several modifications had to be made in order for it to correctly interface with the autopilot. The first major modification made was to remove the controller that came with the system. This controller managed the power settings for the motor as well as translated the pilots turn commands into servo commands. Since it was still desirable to preserver both motor and servo commands, the original controller was replaced by a Futaba six channel PCM receiver and a speed controller. The power for the autopilot and receiver came from the speed controllers built in Battery Elimination Circuit or BEC.

Once the autopilot had been interfaced with the parafoil it was time to test fly the unit in the field. Since it had been found through ground tests that a gain of 0.3 produced desirable results, the gain was left at this setting. Again the system was taken outside and assigned a landing waypoint (36.132332°N, 97.081171°W). To conduct this test, we decided to test how well the parafoil behaved under two different conditions: upwind and downwind. The first test was conducted with the launch location approximately 150 feet downwind of the landing point. The parafoil was launched and then just afterwards the autopilot was turned on. The ground track, shown below in red, indicates that while the parafoil was not able to penetrate the winds, it was able to point in the direction of the home. The reason the parafoil never advanced toward the target was the during the tests the wind speed was noted to be approximately 10 MPH out of NNE and the parafoil's maximum flight speed is only 10MPH, so it was not physically able to penetrate the winds to reach the target.

The next two tests were conducted upwind of the landing point at distances of approximately 400 and 460 feet upwind of the landing point. The first upward flight, shown in the green ground track, was launched from an upward distance of 400 feet from the landing zone and again just after launch the autopilot was switched on. During this flight the autopilot initiated the flight with a counterclockwise turn until it had aligned itself with the landing point and then attempted to maintain this heading for the remaining flight. Unfortunately before the parafoil was able to reach the target landing zone, its flight path intercepted a tree which abruptly ended the flight attempt. Shortly after retrieving the parafoil from the tree the second upwind flight was made. This flight was initiated at a distance of 460 feet upwind from the landing point. Again shortly after launch, the autopilot was started and took control of the parafoil. This time the autopilot initiated a clockwise turn toward the landing zone and then continued correct the heading until it had actually overshot the landing zone at which time it commanded a clockwise turn. This correction to the ground track continued until the system had locked until a relative straight line solution to the landing point. This flight was terminated shortly after the parafoil's glide slope intercepted the roof of the DML, which was the last recorded position as seen in the figure below.

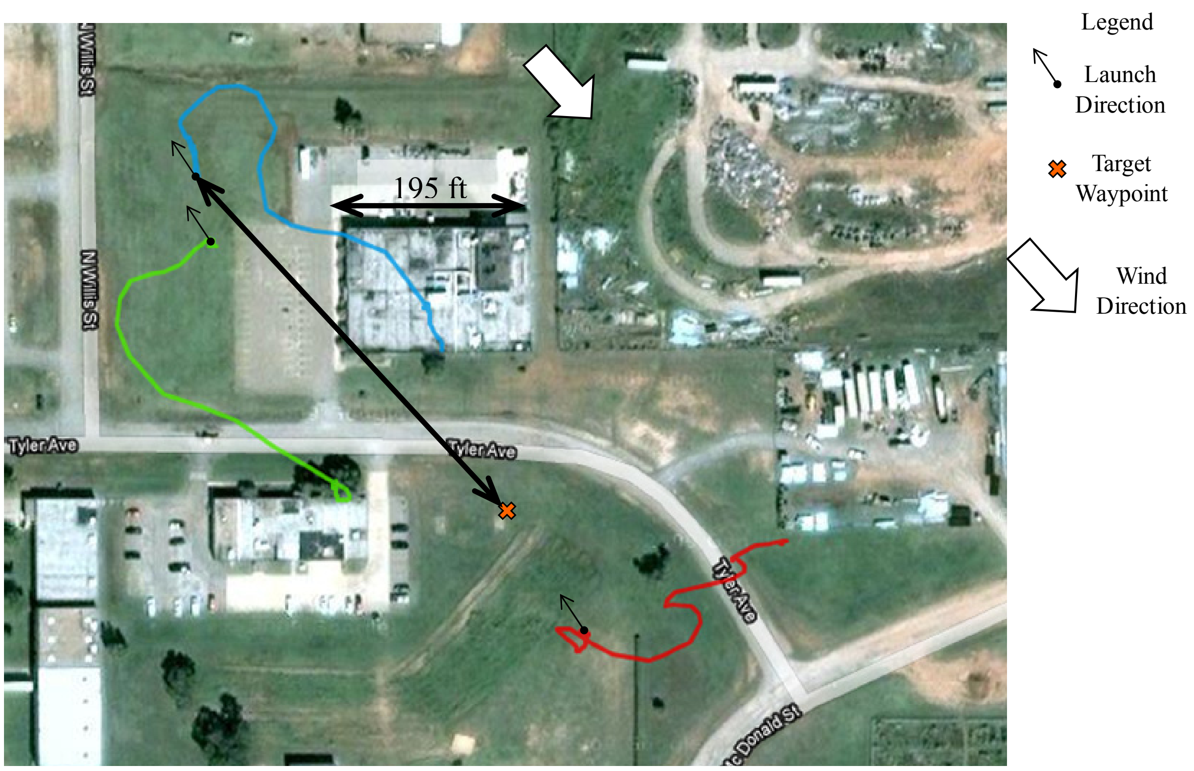


Figure 121 Parafoil Autopilot Aerial Ground Tracks

CONCLUSION AND RECOMMENDATIONS

The major goals for this project have been to improve the ability to predict and track near-space mission payloads, and to demonstrate the feasibility of a low cost autonomous recovery system. To fulfill the first goal, several pieces of custom written software were developed and tested. The Real Time Mission Predictor has the ability to detect and correct for difference in the winds between pre-mission and the ascent phase, as well as correct for the major physical parameters that effect both the ascent and descent phase. On average the error between the finally landing zone that that predicted thirty minutes before landing was found to be less than two miles.

During the process of development of the software a better understanding of the dynamics of the ascent phase were found. Throughout the study it was found that several assumptions that have been made in the past simply do not hold up for near-space missions in the weight and size category, typically ten to twenty pounds. The assumption that the system maintains a near linear ascent rate does not all ways hold true. While there are cases of the ascent rate being nearly linear, not all flight conditions will lead to linear ascent rates. The differences in buoyancy and weight, that cause these non-linear ascent rates, can be relatively small. The assumption of the balloon having a spherical drag model was also not correct. This error leads directly to the problems of assuming a linear ascent rate. By experimentally results, the actual balloon drag does not transition as quickly as was previously assumed. Instead the transition occurs at a more gentle rate than when compared to the classic drag on a sphere, assumed by others.

To fulfill the second goal, two systems have been used. The first system was a ground based vehicle that was used to tune and safely test the autopilot system before porting it over to the flight unit. The second was a powered parafoil that has the same handling qualities of a non-power parafoil. This parafoil was successfully tested under real wind conditions and demonstrated the ability of a low cost autopilot to determine the location and bearing of a target waypoint and then correctly determine a heading to reach it. The unit has fully demonstrated the feasibility of a low cost autonomous parafoil based system to perform a waypoint seeking scheme.

The following are my recommendations for future work are based on the experience that has been gained while conducting this project:

- Determine a waypoint scheme that utilizes the lack of wind penetration of the parafoil recovery system.
- Determine minimum sensor requirements required for optimum path planning
- Upgrading of current power parafoil to a brushless system to improve endurance and increase power available for initial climb out.
- Determine the effect of varying the gains for the system as a function of distance from the target.
- Creation of a wind detect scheme to be used to allow the autopilot to learn the winds during the ascent phase.
- Determine the effects and how to detect changes in the winds between ascent and descent phases.

BIBLIOGRAPHY

1. Fox, John N. The Baffling balloons! *Physic Education.* 1993.

2. NASA Glenn Research Center. Earth Atmosphere Model.

3. Hoerner, S. F. *Fluid Dynamic Drag.* Bakersfield : authors family Hoerner Fluid Dynamics, 1965.

4. Nicolaides, J. *Parafoil Wind Tunnel Tests.* Wright-Patterson AFB : U.S. Air Force Flight Dynamics Laboratory, 1971.

5. Wade, Ian. *APRS Protocol Reference.* Tucson : APRS Working Group, 2000. ISBN 0-9644707-6-4.

6. 4-character ICAO station identifier or a 6-digit WMO index number. *Air Resources Laboratory - READY Current Meteorology.* [Online] NOAA. [Cited: 6 12, 2009.] http://www.arl.noaa.gov/data/weather/text/sfstns.txt.

7. WAAS Performance Analysis Report. *FAA William J. Hughes Technical Center WAAS Test Team.* [Online] January 2008. [Cited: April 18th, 2008.] http://www.nstb.tc.faa.gov/reports/waaspan23.pdf.

8. *Conversion of Earth-centered Earth-fixed coordinates to geodetic coordinates.* Zhu, J. 1994, IEEE Transactions on Aerospace and Electronic Systems , pp. 957-961.

9. *Steering Algorithms for GPS Guidance of RAM-AIR Parachutes.* Smith, B. 1995, Proceedings of the 8th International Technical Meeting of the Institute of Navigation, pp. 12-15.

10. *Modeling and simulation of a ship launched glider cargo delivery system.* Puranik, A. , Parker, G., Passerelle, C., Bird III, J., Yakimenko, O., Kaminer, I. 2006, AIAA Guidance, Navigation, and Control Conference, pp. 5332-5341.

11. *Applying Parachute Canopy Control and Guidance Methodology to Advanced Precision Airborne Delivery Systems.* Hogue, J., and Jex, H. 1995, AIAA Aerodynamic Decelerator Systems Technology Conference, pp. 49-59.

12. *Testing of Guided Parafoil Cargo Delivery Systems.* Kaesemyer, S. 2005, 18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar. AIAA 2005-1668.

13. *Autonomous Large Parafoil Guidance, Navigation, and Control System Design Status.* Carter, D., George, S., Hattis P., McConley, M., Rasmussen, S., and Singh, L. Williamsburg, VA : AIAA, 2007. 19th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar.

14. *Autonomous Large Parafoil Guidance, Navigation and Control System Design Status.* Hattis, P., Carter, D., George, S., McConley, M., Rasmussen, S., Singh, L., Tavan, S. 2007, 19th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar.

15. MMIST Sherpa - GPS Guided Parafoil. *MMIST.* [Online] Mist Mobility Integrated Systems Technology Inc. [Cited: March 14, 2008.] http://www.mmist.ca/Sherpa.asp.

16. *An Optimal Guidance Law for Planetary Landing.* D'Souza, C. 1997, Proceeding of the AIAA Guidance, Navigation, and Control Conference.

17. *Strategies of Path-Planning for a UAV to Track a Ground Vehicle.* Lee, J., Huang, R., Vaughn, A., Xiao, X., and Hedrick, J. 2003. The Second Annual Symposium on Autonomous Intelligent Networks and Systems.

18. *Lateral Track Control Law for Aerosonde UAV.* Niculescu, M. s.l. : AIAA, 2001. 39th AIAA Aerospace Sciences Meeting and Exhibit .

19. *On-line Trajectory Optimization Including Moving Threats and Targets.* Twigg, S., Calise, A.J., and Johnson, E.N. August 2004, Proceedings of AIAA Guidance, Navigation, and Control Conference.

20. *Autonomous Target Following by Unmanned Aerial Vehicles.* Rafi, F., Khan, S, Shafiq, K., Shah, M. 2006, SPIE Defense and Security Symposium.

21. *Optimal Path Planning in a Constant Wind with a Bounded Turning Rate.* Mcgee, T., Spry, S., and Hedrick, J. 2005, AIAA-2005-6186.

22. http://www.atmel.com/dyn/resources/prod_documents/doc8161.pdf. *ATmel Products.* [Online] [Cited: June 15, 2009.] http://www.atmel.com/dyn/resources/prod_documents/8161S.pdf.

23. *Optimal Path Planning for Unmanned Air Vehicles with Kinematic and Tactical Constraints.* Yang, G., Kapila, V. 2002, IEEE, pp. 1301-1306.

24. *Griffon: A Man-Portable Hybrid UGV/UAV.* Yamauchi, B., Rudakevych, P. 2004, Industrial Robot, pp. 443-450.

25. *On the Development of a Scalable 8-DOF Model for a Generic Parafoil-Payload Delivery System.* Yakimenko, O. 2005, AIAA, pp. 642-654.

26. *Applications of Scientific Ballooning Technology to High Altitude Airships.* Smith, M., Rainwater, E. 2003, AIAA.

27. *The Return of the Balloon as an Aerospace Test Platform.* Smith, M. and Allision, G. Arcachone, France : AeroStar, 1999. ESA Reentry Vehicles Symposium.

28. *Aspects of Control for a Parafoil and Payload System.* Slegers, N., Costello, M. 2003, Journal of Guidance, Control, and Dynamics, pp. 898-905.

29. *Development of Flight Test of a Deployable Precision Landing System.* Sim, A., Murray, J., and Neufeld, D. 1994, AIAA Journal of Aircraft, pp. 1101-1108.

30. *Guidance and Control for Flat-Circular Parachutes.* Scott, D., Richard, B., and Glen, B. 2001, Journal of Aircraft, pp. 809-817.

31. *Unmanned Aerial Vehicle Path Following for Target Observation in Wind.* Rysdyk, R. 2006, AIAA Journal of Guidance, Control, and Dynamics.

32. *The Trans-Pacific Crossing: Long Range Adaptive Path Planning for UAVs Through Variable Wind Fields.* Rubio, J. , Kragelund, S. 2003. AIAA/IEEE Digital Avionics Systems Conference.

33. *Unmanned Aerial Vehicle Path Following for Target Observation in Wind.* Rolf, R. 2006, Journal of Guidance, Control, and Dynamics , pp. 1092-1100.

34. *A Flight Simulation Algorithm for a Parafoil Suspending an Air Vehicle.* Redelinghuys, C. 2007, Journal of Guidance, Control, and, Dynamincs, pp. 791-803.

35. *Multibody Dynamics of Parachute and Balloon Flight Systems for Planetary Exploration.* Quadrelli, M., Cameron, J., and Kerzhanovich, V. 2004, Journal of Guidance, Control, and Dynamics, pp. 564-571.

36. Prakash, O. *Aerodynamics, Longitudinal Stability and Glide Performance of Parafoil/Payload System.* Bombay : Department of Aerospace Engineering, Indian Institute of Technology, 2004.

37. *Entry Guidance and Trajectory Control for Reusable Launch Vehicle.* Ping, L. 1997, Journal of Guidance, Control, and Dynamics , pp. 143-149.

38. *Vector Field Path Following for Miniature Air Vehicles.* Nelson, D.R., Barber, D.B., McLain, T.W., and Beard, R.W. 2007, IEEE Transactions on Robotics, pp. 519 - 529.

39. *Entry Guidance and Trajectory Control for Reusable Launch Vehicle.* Lu, P. 1997, Journal of Guidance, Control, and Dynamics, pp. 143-149.

40. *Novel Method for Identification of Aircraft Trajectories in Three-Dimensional Space.* Korgul, A. 2000, Journal of Guidance, Control, and Dynamics, pp. 1021-1029.

41. *On the Development of GNC Algorithm for a High-Glide Payload Delivery System.* Kaminer, I., Yakimenko, O. 2003, Proceedings of the 41st IEEE Conference on Decision and Control Volume 5, pp. 5438 – 5443.

42. *Development of Control Algorithm for the Autonomous Gliding Delivery System.* Kaminer, I., Yakimenko, O. 2003, AIAA.

43. *Trajectory Tracking for Autonomous Vehicles: An Integrated Approach to Guidance and Control.* Isaac, K., Antonio, P., Eric, H., and Carlos, S. 1998, Journal of Guidance, Control, and Dynamics, Vol. 21, pp. 29-38.

44. *Lateral-Directional Aerodynamics from a Large Scale Parafoil Test Program.* Iacomini, C, and Cerimele, C. 1999, AIAA, pp. 218-228.

45. *Optimal Control of a Gliding Parachute System.* Gimadieva, T. 2001, Journal of Mathematical Sciences, pp. 54-60.

46. *Guidance and Control for Flat-Cirular Parachutes.* Dellicker, S., Benney, R., and Brown, G. 2001, Journal of Aircraft, pp. 809-817.

47. *Autonomous Guidance, Navigation, and Control of Large Parafoils.* Carter, D., George, S., Hattis, P., Singh, L., and Tavan, S. 2005, AIAA Aerodynamic Decelerator Systems Conference .

48. Bryson, A. *Dynamic Optimization.* s.l. : Addison Wesley Longman, 1999.

49. *Principles of Guidance-Based Path Following in 2D and 3D.* Breivik, M. and Fossen, T. 2005, 44th IEEE Conference on Decision and Control, pp. 627 – 634.

50. *Guidance-Based Path Following for Autonomous Underwater Vehicles.* Breivik, M. and Fossen, T. 2005, IEEE.

51. *Aerodynamic Improvements of Paraglider Performance.* Babinsky, H. 1999, AIAA, pp. 362-367.

52. *Novel Method for Identification of Aircraft Trajectories in Three-Dimensional Space.* Artur, K. 2000, Journal of Guidance, Control, and Dynamics, pp. 1021-1029.

53. *Trajectory-Tracking and Path-Following of Underactuated Autonomous Vehicles With Parametric Modeling Uncertainty.* Aguiar, A., Hespanha, J. 2007, IEEE Transactions on Automatic Control, pp. 1362-1379.

APPENDIX A: SOFTWARE

Latitude/Longitude/Altitude to Earth Center Earth Fixed Conversion Code

```
Sub lla2ecef(ByVal lon As Object, ByVal lat As Object, ByVal alt As Object, ByRef Xecef As Object,_
              ByRef Yecef As Object, ByRef Zecef As Object)

    ' Xf = ECEF X-coordinate (m)
    ' Yf = ECEF Y-coordinate (m)
    ' Zf = ECEF Z-coordinate (m)
    ' lat = geodetic latitude (radians)
    ' lon = longitude (radians)
    ' alt = height above WGS84 ellipsoid (m)
    '
    '% Notes: This function assumes the WGS84 model.
    '       Latitude is customary geodetic (not geocentric).

    ' Source: "Department of Defense World Geodetic System 1984"
    '        Page 4-4
    '        National Imagery and Mapping Agency
    '        Last updated June, 2004
    '        NIMA TR8350.2

    Dim n As Double 'Prime Vertical Radius of Curvature

    lon = lon * degtorad
    lat = lat * degtorad

    n = a / Math.Sqrt(1 - e ^ 2 * Math.Sin(lat) ^ 2)

    Xecef = (n + alt) * Math.Cos(lat) * Math.Cos(lon)
    Yecef = (n + alt) * Math.Cos(lat) * Math.Sin(lon)
    Zecef = ((1 - e ^ 2) * n + alt) * Math.Sin(lat)

End Sub
```

Earth Center Earth Fixed to Latitude/Longitude/Altitude Conversion Code

```
Sub ecef2lla(ByVal Xf As Double, ByVal Yf As Double, ByVal Zf As Object, ByRef lon As Object,_
            ByRef lat As Object, ByRef alt As Object)

    ' Xf = ECEF X-coordinate (m)
    ' Yf = ECEF Y-coordinate (m)
    ' Zf = ECEF Z-coordinate (m)
    ' lat = geodetic latitude (radians)
    ' lon = longitude (radians)
    ' alt = height above WGS84 ellipsoid (m)
    '
    '% Notes: This function assumes the WGS84 model.
    '       Latitude is customary geodetic (not geocentric).

    ' Source: "Department of Defense World Geodetic System 1984"
    '        Page 4-4
    '        National Imagery and Mapping Agency
    '        Last updated June, 2004
    '        NIMA TR8350.2

    Dim n As Double 'Prime Vertical Radius of Curvature
    Dim b As Double
    Dim p As Double
    Dim ep As Double
    Dim th As Double


    b = Math.Sqrt(a ^ 2 * (1 - e ^ 2))
    ep = Math.Sqrt((a ^ 2 - b ^ 2) / b ^ 2)
    p = Math.Sqrt(Xf ^ 2 + Yf ^ 2)
    th = My_Math.Atn2(a * Zf, b * p)

    lon = My_Math.Atn2(Yf, Xf)
    lat = My_Math.Atn2(Zf + ep ^ 2 * b * Math.Sin(th) ^ 3, p - e ^ 2 * a * Math.Cos(th) ^ 3)

    n = a / Math.Sqrt(1 - e ^ 2 * Math.Sin(lat) ^ 2)
    alt = p / System.Math.Cos(lat) - n

    lon = lon * radtodeg
    lat = lat * radtodeg
    alt = Format(alt, "#0.00")

  End Sub
```

Earth Center Earth Fixed to Local Tangential Conversion Code

```
Sub ecef2neu(ByVal Xf As Double, ByVal Yf As Double, ByVal Zf As Object, ByRef n As Object, _
            ByRef e As Object, ByRef u As Object, ByVal lat As Object, ByVal lon As Object)
    lat = lat * degtorad
    lon = lon * degtorad
    n = -Math.Sin(lat) * Math.Cos(lon) * Xf - Math.Sin(lat) * Math.Sin(lon) * Yf + Math.Cos(lat) * Zf
    e = -Math.Sin(lon) * Xf + Math.Cos(lon) * Yf
    u = Math.Cos(lat) * Math.Cos(lon) * Xf + Math.Cos(lat) * Math.Sin(lon) * Yf + Math.Sin(lat) * Zf
  End Sub
```

Local Tangential to Earth Center Earth Fixed Conversion Code

```
  Sub neu2ecef(ByRef Xf As Double, ByRef Yf As Double, ByRef Zf As Object, ByVal n As Object,_
            ByVal e As Object, ByVal u As Object, ByVal lat As Object, ByVal lon As Object)
    lat = lat * degtorad
    lon = lon * degtorad
    Xf = -Math.Sin(lon) * e - Math.Cos(lon) * Math.Sin(lat) * n + Math.Cos(lon) * Math.Cos(lat) * u
    Yf = Math.Cos(lon) * e - Math.Sin(lon) * Math.Sin(lat) * n + Math.Cos(lat) * Math.Sin(lon) * u
    Zf = Math.Cos(lat) * n + Math.Sin(lat) * u
  End Sub
```

FAI Distance Calculator

```
Sub distance(ByVal lat0 As Double, ByVal lat1 As Double, ByVal lon0 As Double, ByVal lon1 As Double,_
           ByRef FAIdist As Double)
    'FAI(Sphere) Source:
    Dim FAIradius As Double = 6371.0 'WGS84 Radius km
    lat0 = lat0 * degtorad
    lat1 = lat1 * degtorad
    lon0 = lon0 * degtorad
    lon1 = lon1 * degtorad
    FAIdist = Math.Acos(Math.Sin(lat1) * Math.Sin(lat0) + Math.Cos(lat1) * Math.Cos(lat0) * Math.Cos(lon1
             - lon0))
    FAIdist = FAIdist * 1.852 * 10800 / pi

  End Sub
```

FAI Bearing Calculator

```
Sub bearing(ByVal lat0 As Double, ByVal lat1 As Double, ByVal lon0 As Double, ByVal lon1 As Double, _
            ByRef FAIbearing As Double)
  lat0 = lat0 * degtorad
  lat1 = lat1 * degtorad
  lon0 = lon0 * degtorad
  lon1 = lon1 * degtorad

  FAIbearing = Math.Atan2(Math.Sin(lon1 - lon0) * Math.Cos(lat1), _
               Math.Cos(lat0) * Math.Sin(lat1) - _
               Math.Sin(lat0) * Math.Cos(lat1) * Math.Cos(lon1 - lon0))
  FAIbearing *= radtodeg
  FAIbearing += 360
  FAIbearing = FAIbearing Mod 360
  FAIbearing = Math.Round(FAIbearing)


End Sub
```

Earth Atmosphere Model

```
Sub atmosphere_SI(ByVal alt As Double, ByRef density As Double, ByRef Temperature As Double,_
                  ByRef Pressure As Double, ByRef Viscosity As Double)
    'Earth Atmosphere Model
    'NASA Glenn Research Center
    'Altitude meter
    'Temperature K @ Sea Lvl 288.2
    'Pressure Pa  @ Sea Lvl 1.01E+05
    'Density  @ Sea Lvl 1.225
    Const beta = 0.000001458
    Const S = 110.4

    If alt >= 25000 Then
      Temperature = -131.21 + 0.00299 * alt
      Pressure = 2.488 * ((Temperature + 273.1) / 216.6) ^ -11.388

    ElseIf alt >= 36152 Then
      Temperature = -56.46
      Pressure = 22.65 * Math.Exp(1.73 - 0.000157 * alt)

    Else
      Temperature = 15.04 - 0.00649 * alt
      Pressure = 101.29 * ((Temperature + 273.1) / 288.08) ^ 5.256
    End If

    density = Pressure / (0.2869 * (Temperature + 273.1))
    Pressure = Pressure * 1000 'Change to Pa from K-Pa
    Temperature = Temperature + 273.1 'Change to K from C
    Viscosity = (beta * Temperature ^ (3 / 2)) / (Temperature + S)
  End Sub
```

APPENDIX B: HARDWARE

Losi 1/10 HIGHroller Lifted Truck RTR

KEY FEATURES
Front and rear skid plate for scale appearance and protection
Reliable planetary gear differential for increased durability
LM-32K performance motor for great speed acceleration
12RB Electronic Speed Control with reverse to back out of undesired situations
Drive Train Sealed 3-Gear Transmission
Motor: LM-32K Performance Motor
Tire Type Low-Profile, All-Terrain
Length: 18.5 in (470mm)
Wheelbase: 12 in (305mm)
Width: 12.75 in (324mm)
Ground Clearance: 2.0 in (51mm)
Weight: 5.35 lbs (2440grams)
Top Speed: 15mph (24kph)

Gold Rosita’s Paracopter

KEY FEATURES
Parafoil Span: 50 in(1270mm)
Parafoil Chord: 18 in(458mm)
Steering: Pull left/Pull right
Weight with full systems: 0.68 lbs (310grams)
Gondola Length: 9in(229mm)
Gondola Max Width:2.25in(57mm)

VITA

Joseph P. Conner Jr.

Candidate for the Degree of

Doctor of Philosophy

Thesis: DEVELOPMENT OF A REAL-TIME PERFORMANCE PREDICTOR AND AN INVESTIGATION OF A RETURN TO POINT VEHICLE FOR HIGH ALTITUDE BALLOONING

Major Field: Mechanical and Aerospace Engineering

Biographical:

Personal Data: Born in Tulsa, Oklahoma, On December 31, 1970 to Joseph and Sandy Conner. Married Mandy Sheree Conner and reside with our children, Gabriel and Piper, in Perkins, Oklahoma.

Education:
Completed the requirements for the Doctor of Philosophy in Mechanical and Aerospace Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2009.

Completed the requirements for the Master of Science in Mechanical Engineering at Oklahoma State University, Stillwater, 2000.

Completed the requirements for the Bachelor of Science in Mechanical Engineering at Oklahoma State University, Stillwater, 1995.

Experience:
Lab Director for Mechanical and Aerospace Engineering, Appointed 2009
Lecturer for Mechanical and Aerospace Engineering, Appointed 2000
Taught the courses Measurements & Instrumentations and Aerospace Structures I
Assist with Aerospace Engineering Capstone Design Course, Spring 2000 - Current
Assist with Introduction to Engineering (Aerospace) Course, Fall 2004 - Current
Assist with ASTRO Oklahoma Space Grant Consortium Workshop, Summer 2007

Professional Memberships:
Association for Unmanned Vehicle Systems International
American Institute of Aeronautics and Astronautic

Name: Joseph P. Conner Jr. Date of Degree: December, 2009

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: DEVELOPMENT OF A REAL-TIME PERFORMANCE PREDICTOR AND AN INVESTIGATION OF A RETURN TO POINT VEHICLE FOR HIGH ALTITUDE BALLOONING

Pages in Study: 171 Candidate for the Degree of Doctor of Philosophy

Major Field: Mechanical and Aerospace Engineering

Scope and Method of Study: The goal of this study was twofold: First, to gain a better understanding of the governing dynamics in near-space missions and to investigate the feasibility of a low cost autonomous parafoil for control during the descent phase. A study of balloon expansion and drag was conducted to investigate the ascent phase of a flight. The next phase of the project created of a custom prediction program used in real-time to accurately predict the landing location of payloads well before they actually landed. The program was designed to detect and adjust the major dynamic parameters as seen during the flight, and then to use this information to increase the accuracy of predictions. The second goal of this project was to determine the feasibility of using a low cost autonomous parafoil to increase the control over a desired landing zone. This study was conducted in two phases: The first phase tested the autopilot on a ground vehicle to gain confidence in the system. The second phase tested autopilot on an actual parafoil.

Findings and Conclusions: The major findings for this project are: An increase in understanding of the balloon dynamics in the ascent phase was required in order to correctly predict the reaction of the system. In the process of gaining this understanding, two major items were found: First, near-space balloons can be modeled as zero-pressure balloons and thus their expansion is governed by the Ideal Gas Law. Second, the assumed drag model for near-space missions in the size range of 10 to 20 lbs and 600 cu ft of Helium was grossly misunderstood. While the system does transition from a high to low drag coefficient, it does so at a much slower rate than that a classical sphere. The method of detecting and correcting parameters in real-time lead to a major increase in accuracy in the predicted landing zone just after burst. Typically, the system has an accuracy of 2 miles or better, two minutes after burst has occurred. The second part of this study determined that it is feasible to utilize a low cost autonomous parafoil to increase control during the descent phase. The parafoil allows the system to reach a designated landing zone instead of landing at the whim of the winds.

ADVISER'S APPROVAL: Dr. A.S. Arena Jr