Sub-Reflector Proof of Concept

Taylor Bradham

14 April 2011

Abstract

I participated in the Mechanical Engineering Capstone. The 200+ students who participate are split into group of 5-7 members. Each group bids on three projects they find interesting and feel their unique skill set would allow them to complete to project better than any other group. The group I was placed in won the bid for a project sponsored by L-3 Communications based out of Salt Lake City. The project objective was to design a device that reflects a laser beam onto a wall and be able to move quickly and accurately according to user input. I personally was given ownership of testing the accuracy, speed, and acceleration specs of the project, as well as oversee the development of algorithms for the movement of the reflective surface. Based on the criterion set forth by our sponsor the project was a success, and the concept has been approved for further development.

Introduction

I participated in the Mechanical Engineering Capstone at BYU. Capstone is a two-semester team program with an industry-sponsored projects used in the real world. For Capstone I was placed on a team of seven members with a full-time faculty as a coach. For our group the coach was Dr. David Long of the Electrical Engineering department. Through a bidding process our group was assigned a project sponsored by L-3 Communications. L-3 Communications does contracting in command, control and communications, intelligence, surveillance and reconnaissance, government services, and aircraft modernization and maintenance.

L-3 Communications is in need of a highly precise positioning system. The project for the capstone team was to provide a proof of concept for the feasibility of such a device. The device had to meet very specific velocity, acceleration, and positional accuracy specifications. The purpose of proof of concept was to present evidence to L-3 Communications management that they can develop this concept for one of their products; however, the future development of the concept is proprietary and L-3 Communications has asked that it not be discussed in this report.

The official Project Objective Statement is the following: Create a proof of concept that accurately and quickly reflects a laser in response to user defined input according to L-3 Communications specifications by the 14th of April 2011 with a budget of $8,000.

In order to provide the proof of concept L-3 asked our group to design and manufacture a device that would reflect a laser off a mirror to a wall. The device creates an image or shape on the wall based on user-defined input. See Figure 1 on the following page.

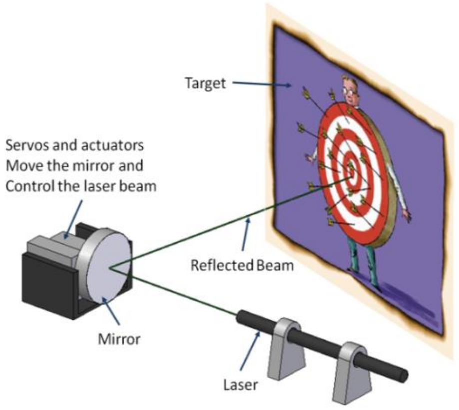


Figure 1 – The device reflects a laser beam onto the wall in a predesigned pattern.

The main goal of the project was to obtain approval for further development of the concept from the management of L-3 Communications. It was determined by the group that the best way to achieve this was to meet the functional specifications given by L-3.

Specifications

The specifications of the project are very tight and are listed in Table 1 below:

Table 1: Functional Specifications

	Metric	Unit	Marginal	Ideal
1	Rotational acceleration	deg/sec^2	8900	12000
2	Rotational velocity	deg/sec	47	60
3	Size	inch	3	3
4	Positional accuracy	deg	0.06	0.025
5	Tilting angle	deg	±4	±5
6	Voltage	volts	28	24
7	Current	amps	3A continuous, 5A peak	1
8	Rotation	rotational degrees of freedom*	2	2
9	Center of rotation about top of reflective surface	Binary	Pass	Pass
10	Functions consistently independent of user	binary	Pass	Pass
11	Appearance	binary	Pass	Pass
12	Flat mirror	binary	Pass	Pass
13	Vertical displacement of mirror	meter	.0064 (.25in)	0.0005
14	Displacement of x-y plane	meter	.00076	0
15	Contains complete design documentation	binary	Pass	Pass

Design

After weeks of careful concept development and screening the group decided on pursuing a design based on the push and pull of the mirror surface with two high-precision linear actuators pictured in Figure 2. The specifications of these actuators, the Zaber NA08A16 actuator, are well within the need of the project. The product specifications can be found in Appendix A.

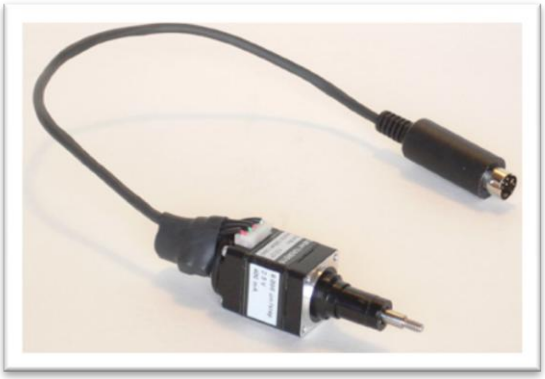


Figure 2 – Zaber NA08A16 actuators used in the project.

Two designs were pursued that used different methods of attaching the actuators to the surface; one uses compliant mechanisms and the other a gimbal.

Gimbal

The gimbal, pictured in figure 3 to the right, was designed in mimic of the puzzle game called "Labyrinth." The gimbal keeps the center of rotation about the top of the reflecting surface. The actuators are attached to the reflecting surface using springs connected to a roller attachment as seen in Figure 4 on the following page.

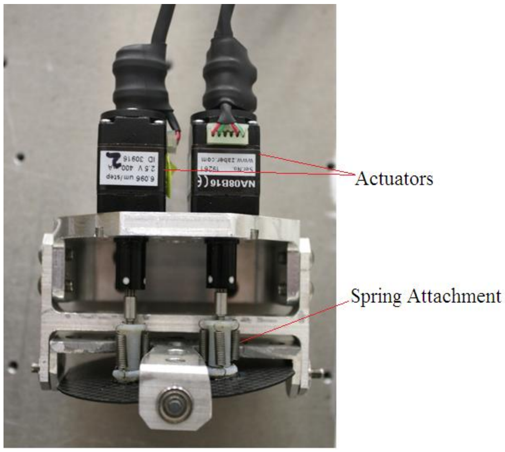


Figure 3 - Picture of Gimbal design

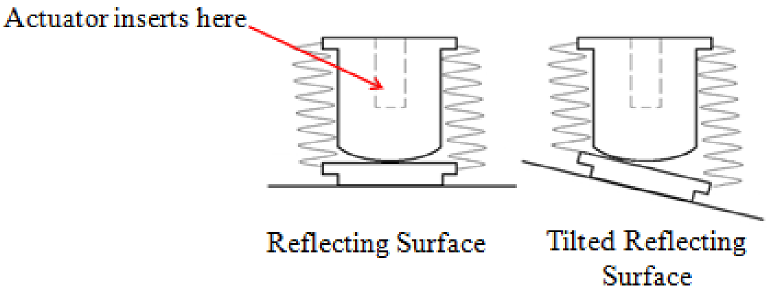


Figure 4 - Spring attachments for gimbal design

The gimbal is comprised of three main parts:

- X-bracket (Figure 5a): Holds onto the straight bracket and allows for the two axes of rotation necessary to achieve a full range of motion. The X-bracket has large holes along one axis that allows the straight bracket to rotate using bearings, and another axis into which pins may be pressed.
- The straight bracket (Figure 5b): Adheres to the carbon fiber piece and rotates about one axis as it is pressed into the bearings on the x-bracket.
- The H-bracket (Figure 5c): Although most of the design has little dynamic effect on performance, the straightness and diameter of its large hole must be precise to ensure correct performance.

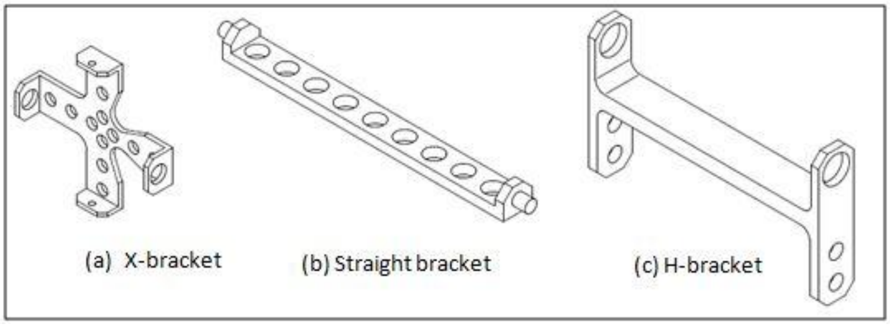

(a) X-bracket
(b) Straight bracket
(c) H-bracket

Figure 5 - Gimbal Part Drawings

Compliant

The compliant design is much simpler than the gimbal. It uses just two polypropylene compliant pieces to attach the actuators to the reflective surface and one additional compliant piece to keep the mirror centered and serve as a mechanical stop. The two advantages of compliant mechanisms are:

- Ideal for small movements. Since L-3 requires the reflector to tilt only four degrees, compliant mechanisms provide both sufficient stability and flexibility.
- Polypropylene material provides a high strength to modulus ratio, allowing for maximum strength and flexibility.

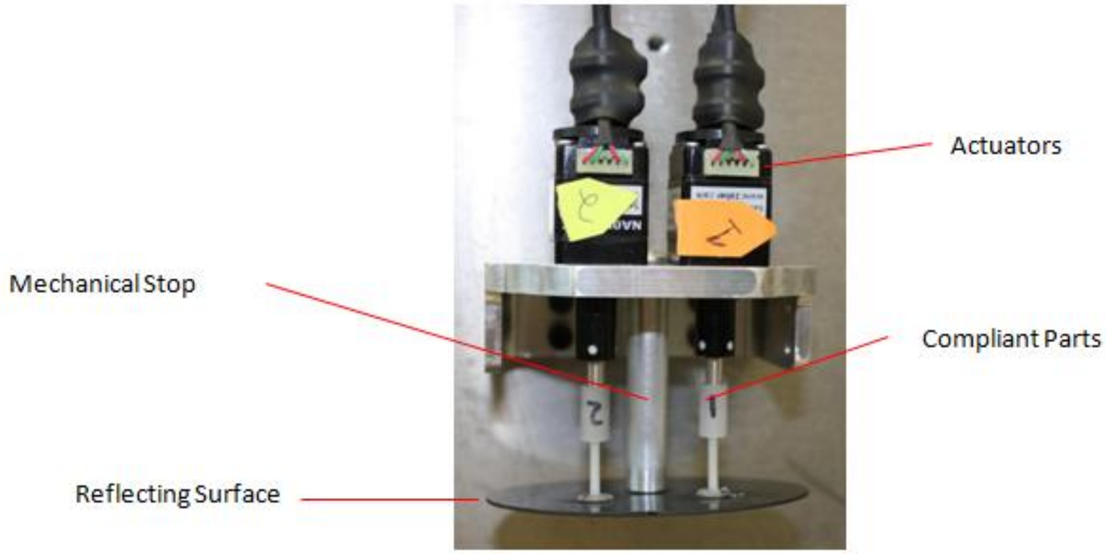


Figure 6 - Compliant Design

Methods

The specifications of the project are extremely tight, and I was given ownership of developing the testing methods. Most instruments that can test such specifications are very expensive and are beyond the budget of the project. Due to this limitation-3 stated that testing should remain within budget and whatever results the group can achieve should be reported to them. With the resources available at BYU it was determined that the best way to test the devices is by imaging them.

The Mechanical Engineering department has high-speed cameras, available for student use. The cameras take a 1 megapixel video at a rate of 1000 frames per second or a lesser quality image, about half a megapixel at a rate of up to 3000 fps. A set of extension tubes was attached to the lens of the camera to allow positioning the camera as close as possible to the device. We zoomed in as close as close as possible on a ruler in order to determine how tight of a resolution the camera can resolve. The resolvable area is about 1cm. When converted to degrees based on a 3-inch diameter circle, the displacement of the device can be resolved to within .013 degrees. With that resolution the positional accuracy of the device can be tested well within the .06 degree spec. The 3000 fps camera rate also allowed accurate testing of the velocity and the acceleration within spec as well.

It was important that both axes be tested at the same time in order to measure the affect that moving one axis has on the other. The mirror setup shown in Figures 6 and 7 provided the solution. With the mirrors arranged according to the Figures the image on the camera is split between each axis.

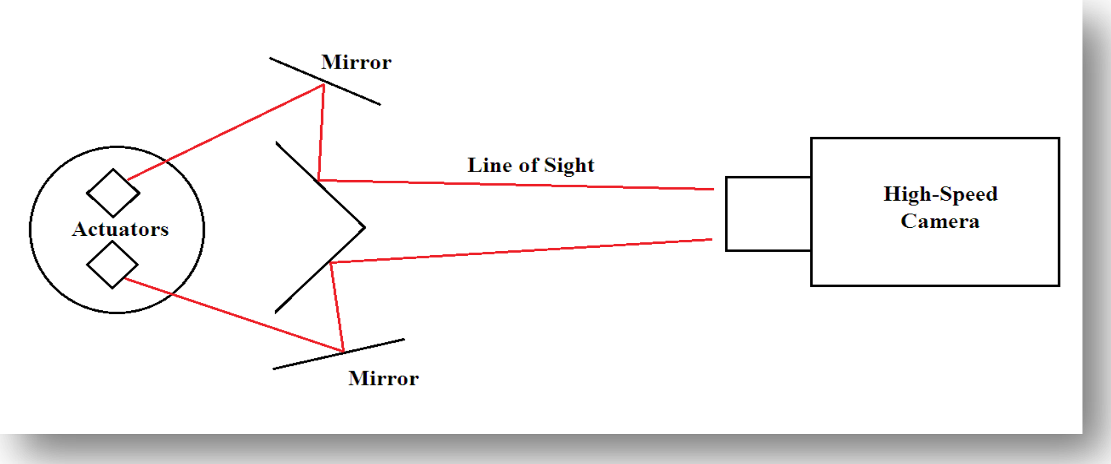


Figure 7 – Diagram of testing setup

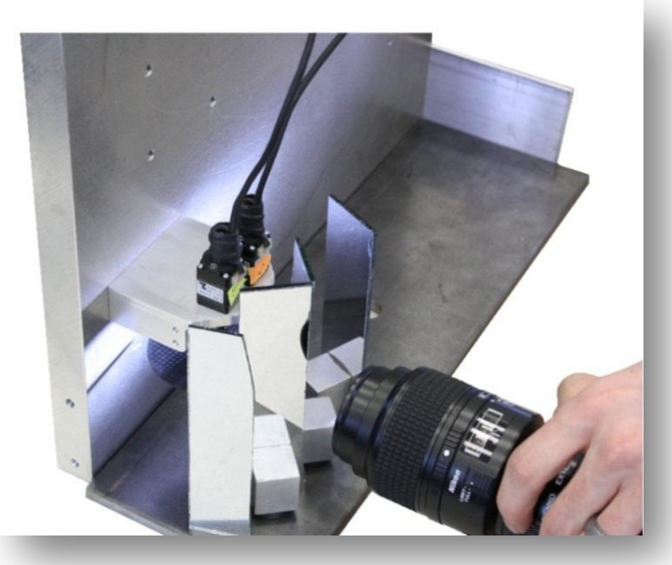


Figure 8 – Picture of testing setup

Testing Procedure

The following is the procedure followed to test the device using the setup previously described:

For velocity and acceleration:

1. Run one actuator between a target of +4.0° and -4.0° capturing with the high-speed camera.
2. Use script in Matlab written by Dr. David Long (Appendix B) to analyze the angular position of each axis as a function of time.
3. Use second curve fitting tool in Matlab (Appendix C) to find a fitting curve for the angular position
4. Take first and second derivative of position equation to find velocity and acceleration

- For positional accuracy:

1. Move the actuators to trace the perimeter of the calibration grid as shown in Figure 8. The target output is given in degrees. The script converts the degree input into a linear displacement for each actuator.
2. Use script in Matlab (Appendix B) to assess the position of the device after each move.
3. Place resulting data into new grid and compare with the predetermined grid.

- For planar displacement:

1. Take pictures using a static camera from bird's eye view of mirror surface in a flat position and then at each angular extreme.
2. Find pixel number at each end of the mirror surface and average them to find pixel at center of mirror surface
3. Compare resulting pixels and use pixel/inch conversion to calculate planar displacement

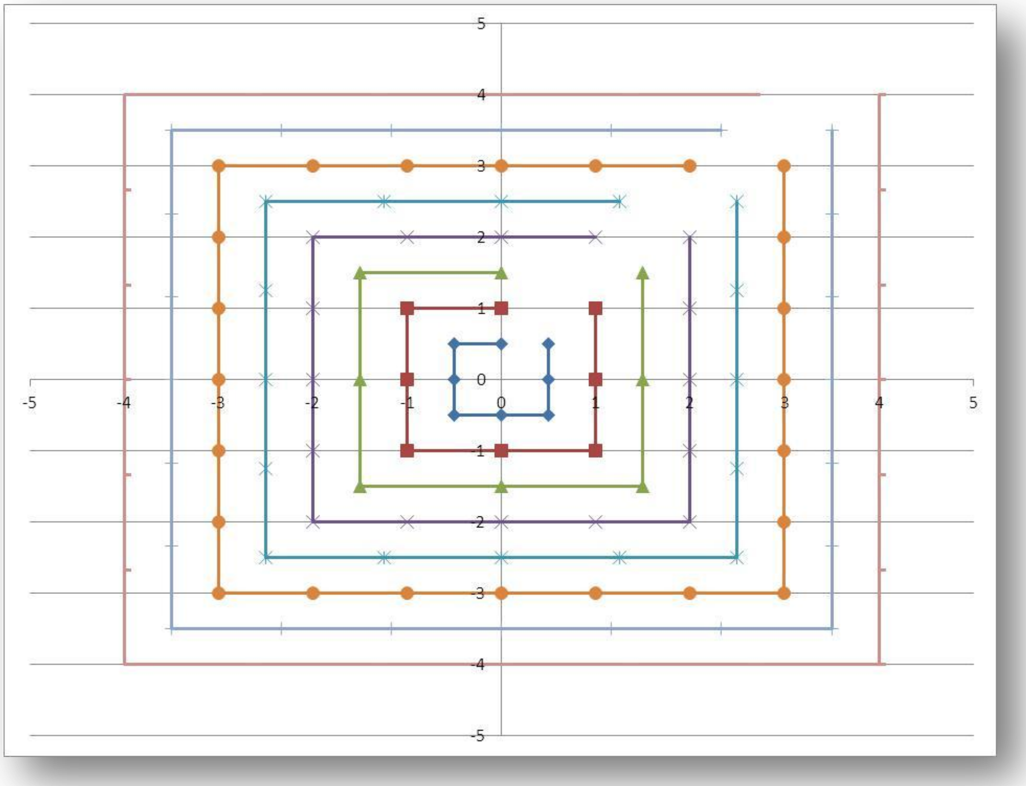


Figure 9 – Each axis represents an axis one of the actuators is positions on. Units are in degrees.

The same testing procedures were used for both the gimbal and compliant designs for consistency. The results were then compared to each other in order to determine which design best meets the specifications needed for the project.

Algorithms of Motion

I was also put in charge of the scripting of the motion algorithms. I worked closely with another group member on this part of the project. Listed below are the decisions made and reasoning for them for this portion of the project.

1) Use velocity based algorithm
    a. This decision was apparent. Even though controlling by points produces the highest precision, it forces the actuators to start, go to a point, and then stop. The movement is then not continuous and the speed of the demonstration is greatly hampered. After consulting the team coach and L-3 it was agreed that using points is impractical, and extrapolating velocities from target points is more effective.
2) Not use feedback for control program
    a. This was a key decision because without feedback, the device cannot be self-correcting. In order to obtain the actuator's current position the position of the actuator must be polled This leads to several problems:
        i. There is significant delay in polling the actuator (slower program run time)
        ii. It is impossible to determine exactly when the actuator reaches the sent position (immediately after the command is sent, or right before the data is received). If it is unknown exactly when the value for position is

accurate within a certain degree of certainty, the data value does more harm than good.

iii. Actually applying the feedback values requires calculating velocities in real time. More calculations lead to a slower program.

b. All of the problems listed above led to the conclusion that with our current resources using feedback from the actuators is impractical.

3) Calculate velocities beforehand

a. After deciding to not use feedback, this decision was straightforward. The values are the same if the velocities are calculated before or in real time because they are based solely on target point locations.

4) 2 point values per calculation

a. Dr. Lyon, one of the contacts from L-3, helped make this decision. He informed us that using many points for extrapolating a velocity is useful for devices using feedback, but is not really beneficial when using solely target values. When using target values, 2 points is as helpful as 10 points.

5) Use inches/second

a. Ultimately the actuators required a "data value" for a given velocity. Which units were used was relatively arbitrary, so with very little discussion it was decided to use on inches/second.

6) How many velocities per iteration to use

a. The number of values used has a high impact on the speed performance of the actuators. The more values per iteration, the slower it draws. That said, with too few points the shape looks very boxy (This works fine for designs with straight lines and corners, but very bad for any rounded edges).

   b. After some experimentation it was decided that the number of values used has to depend on the shape. A square requires only 4 values, a circle requires at least 8, and more complex images require many more values. The ultimate decision for each image comes down to a trade-off between speed and detail.

7) Correcting regular drift with offset

   a. In testing the code, it was found that the actuators were consistently drifting as they drew images. This was expected because the actuators require integer values. This meant the velocities had to be rounded.

   b. In order to correct the drift a combination of velocity absolute offsets and gains is being used to help prevent drifting of the image to any edge during the presentation.

A few difficulties arose in creating the desired shapes; however, these difficulties can easily be remedied. The algorithms were tested by simply running the script with a laser reflecting off the mirror surface and onto the wall. The resulting figures contained distinct "wiggles" as the script ran each loop. The shape itself also randomly shifts completely by as much as a half-inch. There were a few reasons for this:

1. We used the program provided by Zaber Industries for running the scripts. This program, which transmits data through the USB to a serial port on the controllers, is only single-threaded. It can therefore only send commands to one

actuator at a time. This created a slight lag in between each command resulting in the wiggles seen in the figures.

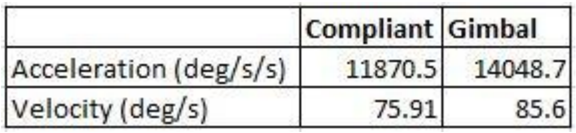
2. The program was run on a Windows platform, which has numerous services running in the background. Especially when Windows suddenly runs one of these services in the background, the rate at which the velocity commands are sent to the controllers is delayed slightly and causes the shift.

These issues were discussed with our contacts at L-3 and they informed us that they can easily overcome those problems by building a hardware-based platform that can handle double-threading. They can also build their own controllers that are much faster than those provided by Zaber. These issues therefore do not affect the outcome of the project.

Results and Discussion

The testing methods worked well for measuring the velocity and acceleration of the devices, but there were unexpected complications with measuring the positional accuracy of the device. The acceleration and velocities measure are found in Table 2 below.

Table 2 – Velocity and Acceleration Results



	Compliant	Gimbal
Acceleration (deg/s/s)	11870.5	14048.7
Velocity (deg/s)	75.91	85.6

There is a moderately high error margin on these results due to the potential errors in the curve fitting process. Changing just a few minor parameters, such as the range of points selected for the curve fit, changes the results significantly. This is especially the case with the acceleration profile. We can easily conclude, however, that the devices exceed the minimum requirements for velocity and acceleration. The graphs of these results can be found in Appendix D.

A grid of the measured positions was created modeling Figure 8. The results were not at all what was expected, and are found in Figures 9 and 10. The axes in the figures are in inches.

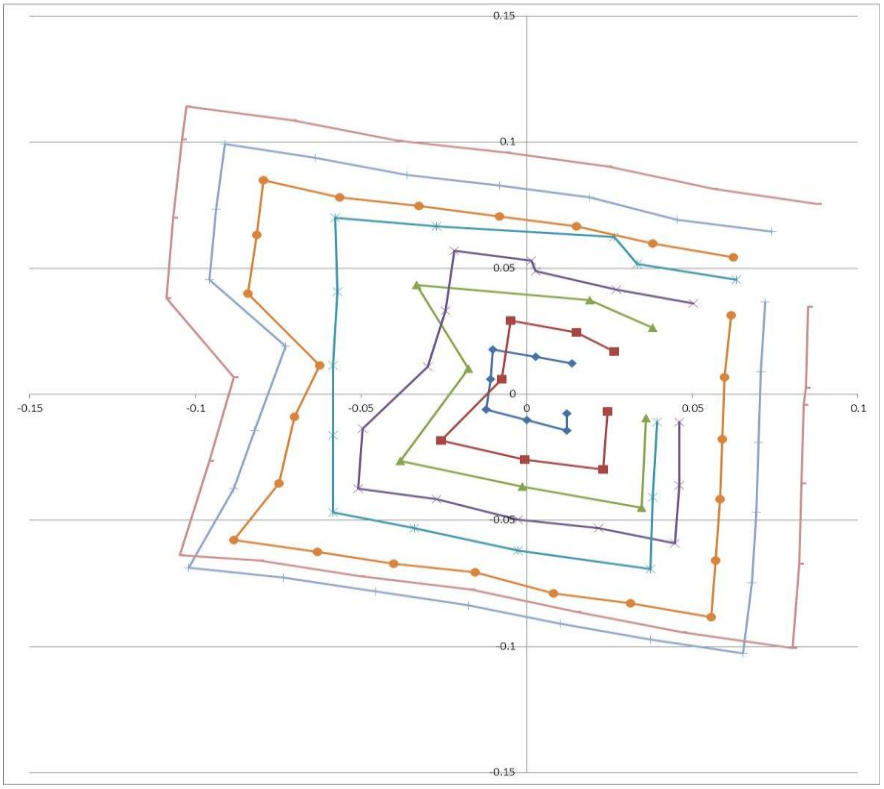


Figure 10 - Positional accuracy results for the compliant device.

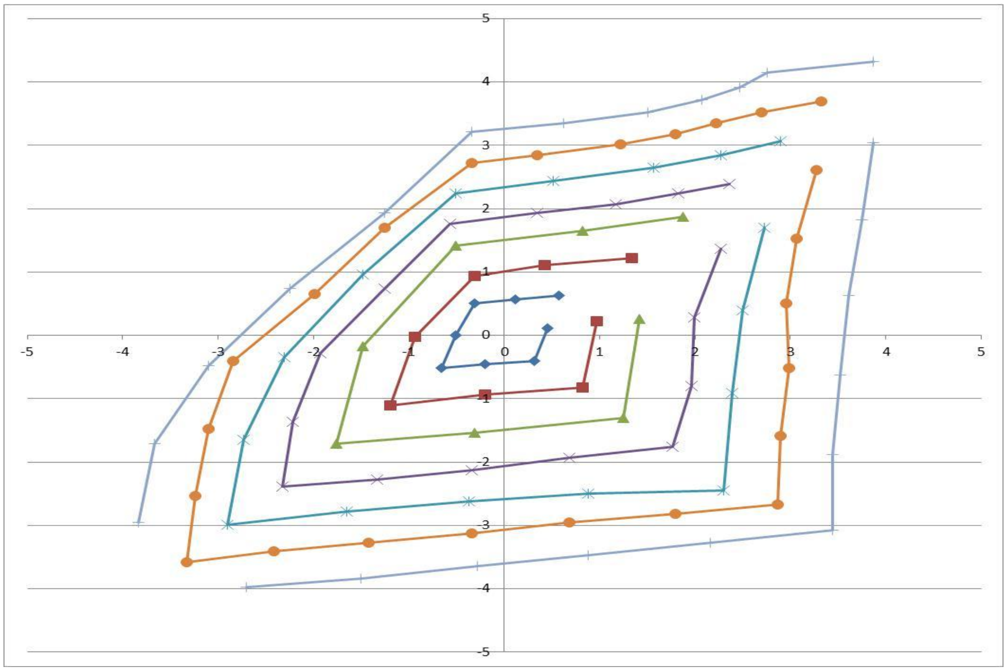


Figure 11 - Positional Accuracy results for the gimbal device.

After analyzing these results I believe that this gross error came from the testing and not positional inaccuracies. I quickly designed a simple test to ensure that it was the testing procedures that caused such sporadic results. The following procedure was used to re-measure the positional accuracy:

- Place a static camera with the lens pointed in the same direction as the reflected laser.
- Run the calibration grid against a dark background.
- Image the grid with the camera on an extended shutter speed so that the grid runs in the entirety three times before the shutter closes
- Use the resulting picture to analyze positional accuracy by locating position (in pixels) of each intensity peak and converting displacement.

The images and new grid are in Figures 11 and 12:

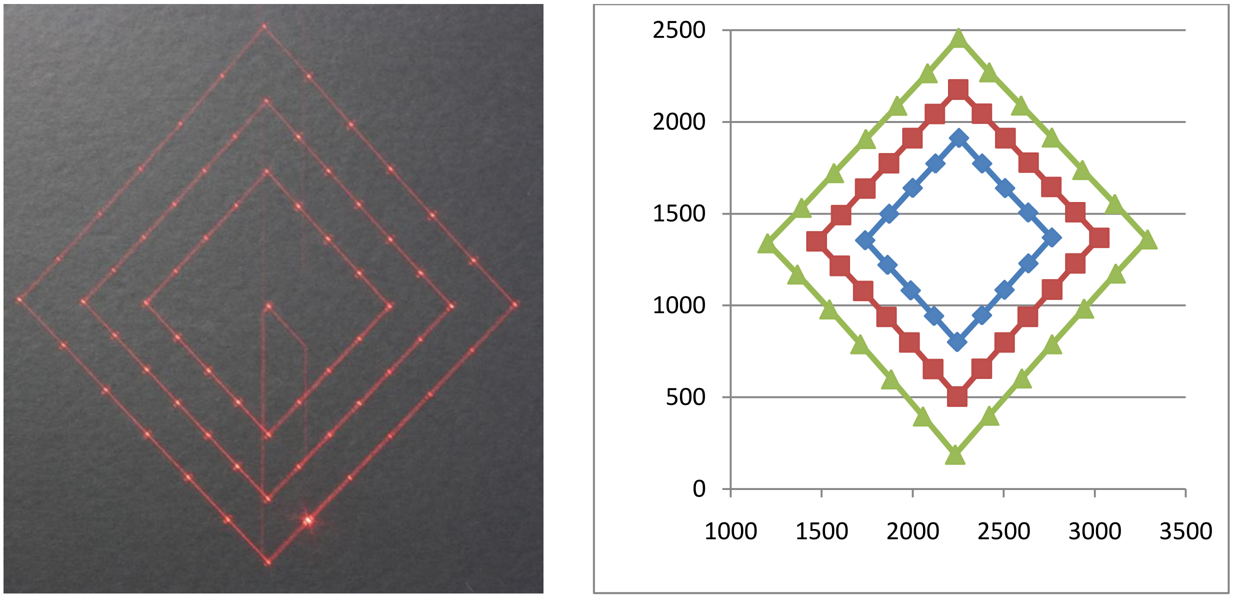
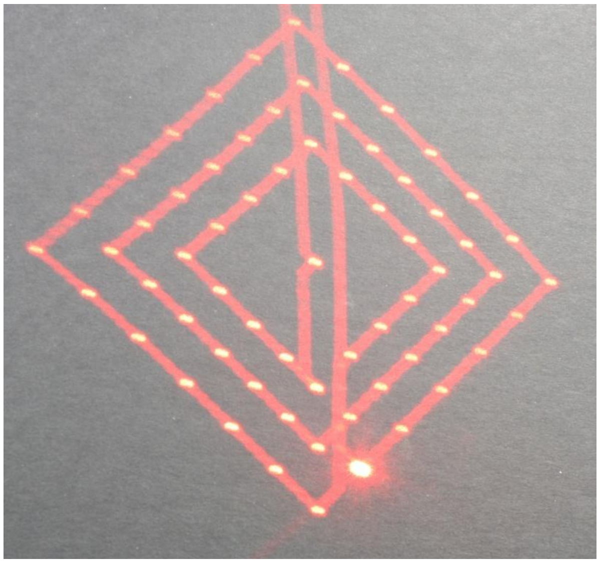


Figure 12 – (Left) Image of compliant design running position grid three times at 2, 3, and 4 degrees of tilt. (Right) Analytical grid of each intensity peak with pixel count.



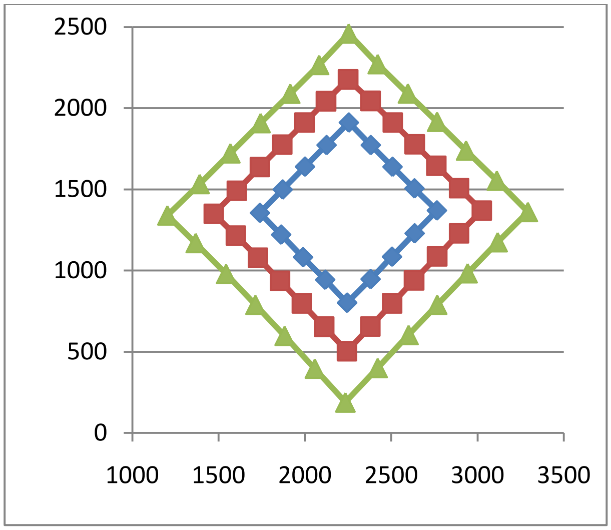


Figure 13 - (Left) Image of gimbal design running position grid three times at 2, 3, and 4 degrees of tilt. (Right) Analytical grid of each intensity peak with pixel count.

These results make it obvious that there were errors in the testing for positional accuracy. The reasons for the errors are:

- Misalignment of the mirrors. The mirrors were aligned using the human eye because optical grade equipment was not available and out of budget for purchasing.
- Low lighting. The Matlab script follows the change in light intensity, and the image on the high speed camera was extremely dark with the frame rate so high. LED lights were used to light up the edges of the reflective surface, which caused areas of light saturation when the devices reached a certain angle of tilt. One of the images taken by the high-speed camera can be seen in Figure 13. You can see the light saturation heavy on the left side.
- On the compliant device the center of rotation is behind the mirror rather than at the front.

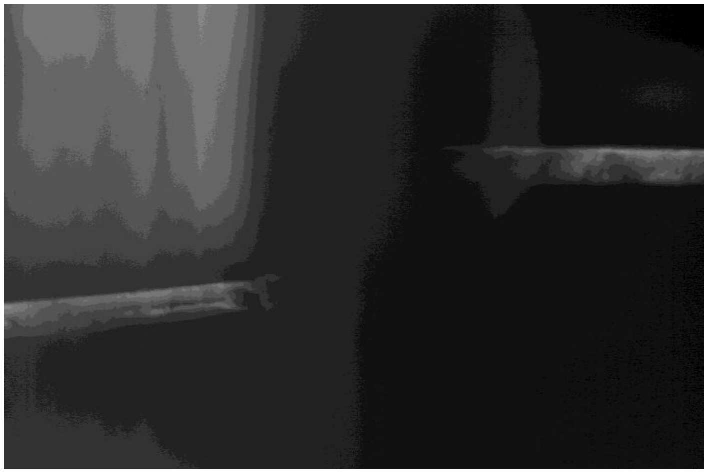


Figure 14 - Image taken by the high-speed camera. Heavy light saturation can be seen on the left side.

The images taken with an extended shutter are not precise enough to measure the positional accuracy of the devices within the desired .025 degrees; however, they can measure the accuracy to within about .1 degrees, which is not far above the desired precision. Analysis of the new grids shows that neither device positions to the exact angle programmed. The repeatability of each move, however, is extremely high. So any inaccuracies can easily be corrected with a simple calculation to put it in the desired position.

Conclusions

The goal and purpose of this capstone project was to create a proof of concept for a fast and accurate sub-reflector for radio antennae. L-3 Communications outlined very tight specifications for measuring the success of the project. These measures are tight because the end product that L-3 designs and builds has to overcome a 30 Hz vibration with very small amplitude. A sub-reflector that is too slow or inaccurate may not only fail in minimizing the signal loss, but can also amplify the loss.

We concluded the project to be a success for two reasons:

1. Aside from positional accuracy we were able to measure all of the specifications to within spec, and found that they do meet spec. Although we were not able to measure the positional accuracy to within the spec desired, we feel confident that both designs meet the spec.
2. Further development of the project has already been approved by the management of L-3 Communications. Our contacts at L-3 sent our reports and testing results to their management as the project progressed, and the management felt confident enough in the project that they a approved the development of the concept into a full functional product.

Acknowledgements

L-3 Communications Systems-West – Project Sponsor

Neil Harker – L-3 Contact

Dr. Scott Lyon – L-3 Contact

Dr. David Long – Coach/Mentor

Mechanical Engineering Capstone

Team Members

David Biladoe – Mechanical Engineering

Peter Ashworth – Mechanical Engineering

Darrell Skousen – Mechanical Engineering

Kevin Alstrom – Electrical Engineering

Caleb Perry – Electrical Engineering

Logan Bird – Manufacturing Engineering

Appendix A

Specification (click for definition)	Value	Alternate Unit
Microstep Size (Finest Resolution)	0.0238125 µm	
Microstep Size (Default Resolution)	0.047625 µm	
Travel Range	16 mm	0.630 "
Accuracy	+/- 8 µm	+/- 0.000315 "
Repeatability	< 0.4 µm	< 0.000016 "
Backlash	< 2 µm	< 0.000079 "
Minimum Speed	0.0004465 mm/s	0.00002 "/s
Speed Resolution	0.0004465 mm/s	0.00002 "/s
Maximum Speed	14 mm/s	0.551 "/s
Maximum Continuous Thrust	19 N	4.3 lb
Motor Temperature Rise	75 degrees C	
Linear Motion Per Motor Rev	0.6096 mm	0.024 "
Motor Steps Per Rev	200	
Motor Type	2 phase nema 8	
Motor Rated Current	400 mA/phase	
Motor Winding Resistance	5.1 ohms/phase	
Inductance	1.5 mH	
Motor Connection	Minidin 8 male	
Integrated Controller	No	
Recommended Controller	T-MCA	
Motor Frame Size	8	
Mechanical Drive System	Precision leadscrew	
Limit or Home Sensing	Magnetic home sensor	
Axes of Motion	1	
Mounting Interface	4 M2 tapped holes 15.4 mm apart or 9mm dia smooth shank	
Vacuum Compatible	No	
CE Compliant	Compliant	
RoHS Status	Compliant Version Available	
Weight	0.06 kg	

Appendix B

```
% capstone22 video analysis code
% written by DG Long at BYU 3 Nov 2010

clear; close all;

%DEBUG=1;
DEBUG=0;  % turn off debugging (runs faster)

% uncomment only one directory to analyze frames from
%dpath='dis_vel__0_02_C001H001S0001';
%dpath='dis_vel__0_04_C001H001S0001';
%dpath='dis_vel__0_06_C001H001S0001';
%dpath='dis_vel__0_08_C001H001S0001';
%dpath='dis_vel_0_1_C001H001S0001';
%dpath='dis_vel_neg_0_02_C001H001S0001';
%dpath='dis_vel_neg_0_04_C001H001S0001';
%dpath='dis_vel_neg_0_06_C001H001S0001';
%dpath='dis_vel_neg_0_08_C001H001S0001';
%dpath='dis_vel_neg_0_1_C001H001S0001';
dpath='repeatability_C001H001S0001';
linenum='2';
resultline=['k' linenum]; %this allows us to save the displacement in inches from the starting postion to a file.  This sets the column


% set column to look at
col=399;
% set actuator end detection threshold
thres=50;

if DEBUG
  % create debugging figures
  figure(1);colormap('gray');
  figure(2);
end

% get file listing
D=dir([dpath filesep '*.jpg']);

Nfile=size(D,1);
% initialize plotting
travel=zeros(1,Nfile);

for count=1:Nfile

  % select a single file
  fname=D(count).name;
  file=[dpath filesep fname];
  % read image from file
  a=imread(file);

  % extract image pixel values along column
  pcol=a(:,col);

  % find end of actuator
  ind=find(pcol>thres);
  ind=ind(end)+1;

  % record vertical position (in pixels)
  travel(count)=ind;

  if DEBUG
    % display data along column
    set(0,'CurrentFigure',2);
    plot(pcol); % show image pixel value along extraction line
    hold on; plot([0 length(pcol)],[thres thres],'r:'); hold off % show threshold
    xlabel('vertical displacement (pixels)')
    ylabel('pixel value')
    axis([0 length(pcol) 0 256])

    %display full iamge
    set(0,'CurrentFigure',1);
    imagesc(a);% show image
    hold on;plot([col col],[1 size(a,1)],'r'); hold off; % show extractionline
    title(sprintf('%s  %d',escape_underbar(fname),ind));
    drawnow
  end

end

% note: scale travel by inches/pixel to convert to distance
travel=travel-travel(1); %this makes it he aboslute position relative to where is started.
travel=travel*0.025/17.037;

% create time axis
time=(1:length(travel))*0.001;

% plot displacement versus time
figure(3)
plot(time,travel)
ylabel('vertical displacement (pixels)')
xlabel('time (s)');
title(sprintf('Actuator end travel vs frame %s',escape_underbar(dpath)));

% print main plot to a file (if desired)
print([dpath '.png'],'-dpng');
%xlswrite('results.xls', travel', 'results', resultline);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% source code for escape_underbar
%function out=escape_underbar(in)
```

```
%%
%% function out=escape_underbar(in)
%%
%% generates "escaped underbar" string out from input string in with
%% underbars so that underbars are not interpretted as subscripts
%%
%% e.g. out="this\_is\_a\_test" from "this_is_a_test"
%%
%% written 11/17/2007 by DGL at BYU
%
%out=sprintf('%s',in);
%ind=find(out=='_');
%for k=length(ind):-1:1
%  if ind(k)==1
%    out=sprintf('\\%s',out(ind(k):end));
%  else
%    out=sprintf('%s\\%s',out(1:ind(k)-1),out(ind(k):end));
%  end
%end
```

Appendix C

```
clear; close all;

load [datafile].txt

x=0:1/[frame rate of camera]:[# of time steps]/[frame rate of camera];

y=[datafile](:,1)';

cftool(x,y)
```

Appendix D

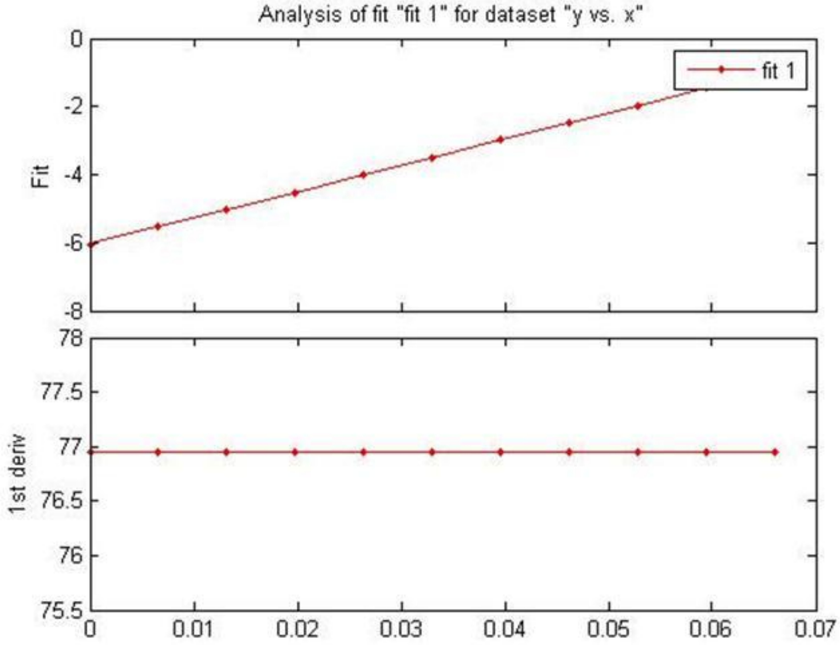


Figure 1 – Velocity graph for the compliant design.

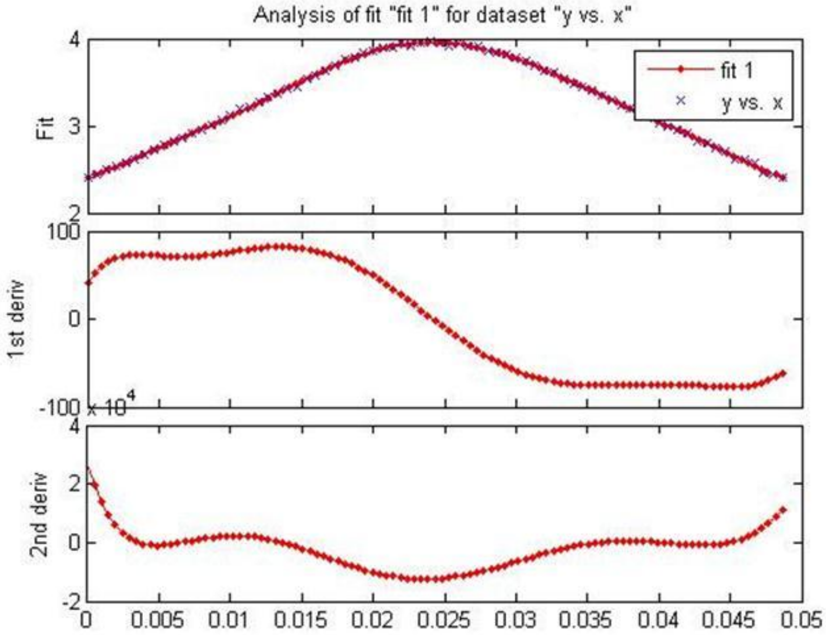


Figure 15 – Acceleration graph for the compliant design.

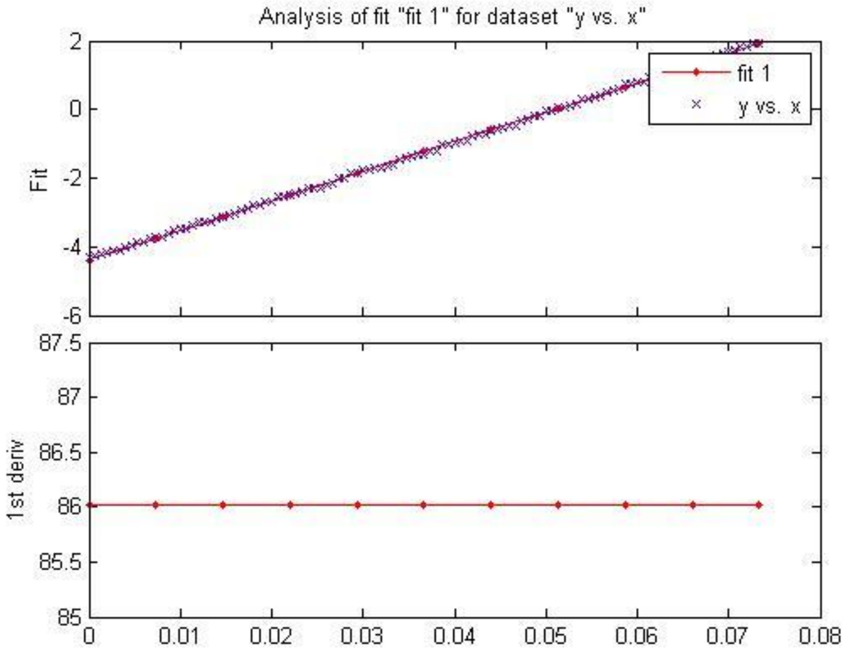


Figure 16 – Velocity graph for the gimbal design.

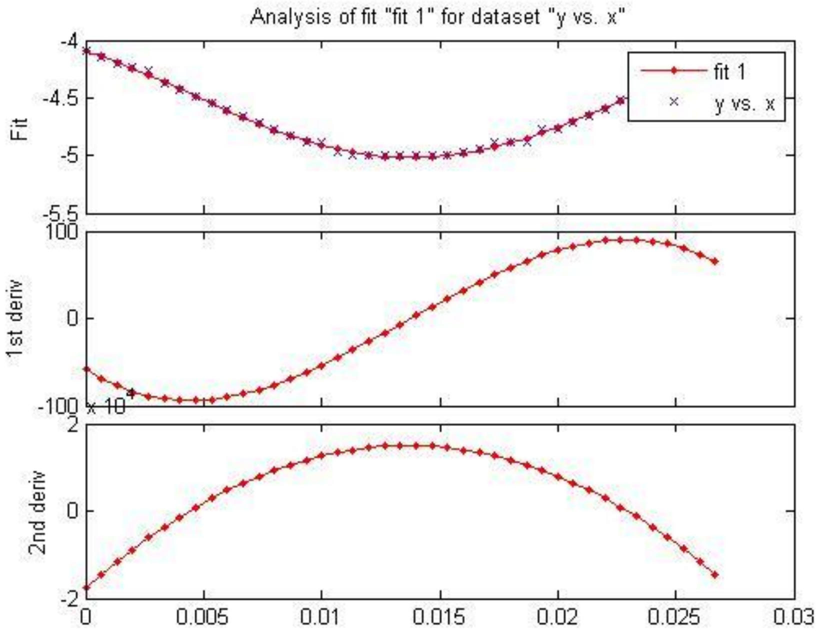


Figure 17 – Acceleration graph for the gimbal design.