First, we want to test our new Euler angle initialization program with the data already collected from last week. When we first ran it, we noticed that the graphs didn't look right. So we double checked the new initialization code. We noticed that we forgot to change latitude into radians. So we added: lat = lat*pi/180 at the beginning of the code. We also noticed, in the attitude computer, that when we call the gravity computer, we were passing the variables incorrectly. The order in which we were passing the variables was wrong. So we changed it to: gravity(lat, h, earth).
Which brings up another change to our attitude computer that we decided on. Notice that we passed earth into the gravity computer. We decided that since several functions call the earth_param function throughout the loop, we would call the earth_param function once at the beginning of the loop and pass the variables to the functions that needed it. This will save time because there are several calculations being made in the earth_param function.
Another change that we made was updating the gravity computer each time the loop is run. We didn't do this before because the latitude wasn't updated each time. It isn't really necessary for our purpose because we will never travel far enough to make a difference in gravity, but if our code was ever used in a car or plane, then it would make a difference.
Since we added an updated gravity computer, we decided that we would have to update the latitude position. So we decided to implement the update latitude, longitude, and height function. It was already written over winter break, we just had to call it. Here is the code:

%LATITUDE, LONGITUDE, HEIGHT UPDATE

```
function [pos] = position2(v, lat, h, delt, earth)
% [position] = position2(velocity, lat, height, earth)
%
%                pos.latitude(i)
%                   .longitude(i)
%                   .height
%
% Written By: Brian  Bleeker
%           Rob MacMillan

%earth = earth_param

pos.lat = v.n/(earth.ro + h);
pos.lon = v.e*sec(lat*pi/180)/(earth.ro + h);
pos.h = -v.d;

pos.lat = pos.lat*delt;
pos.lon = pos.lon*delt;
pos.h   = pos.h*delt;
```

One last change that we made before we did anymore testing. We went through every function that we wrote and double checked the code. We wanted to make sure that we pass the variables correctly and that we used the correct earth parameters for the function. So after we made all those changes, we ran the code using the 360 degree yaw change. Here is the Euler angles graph:

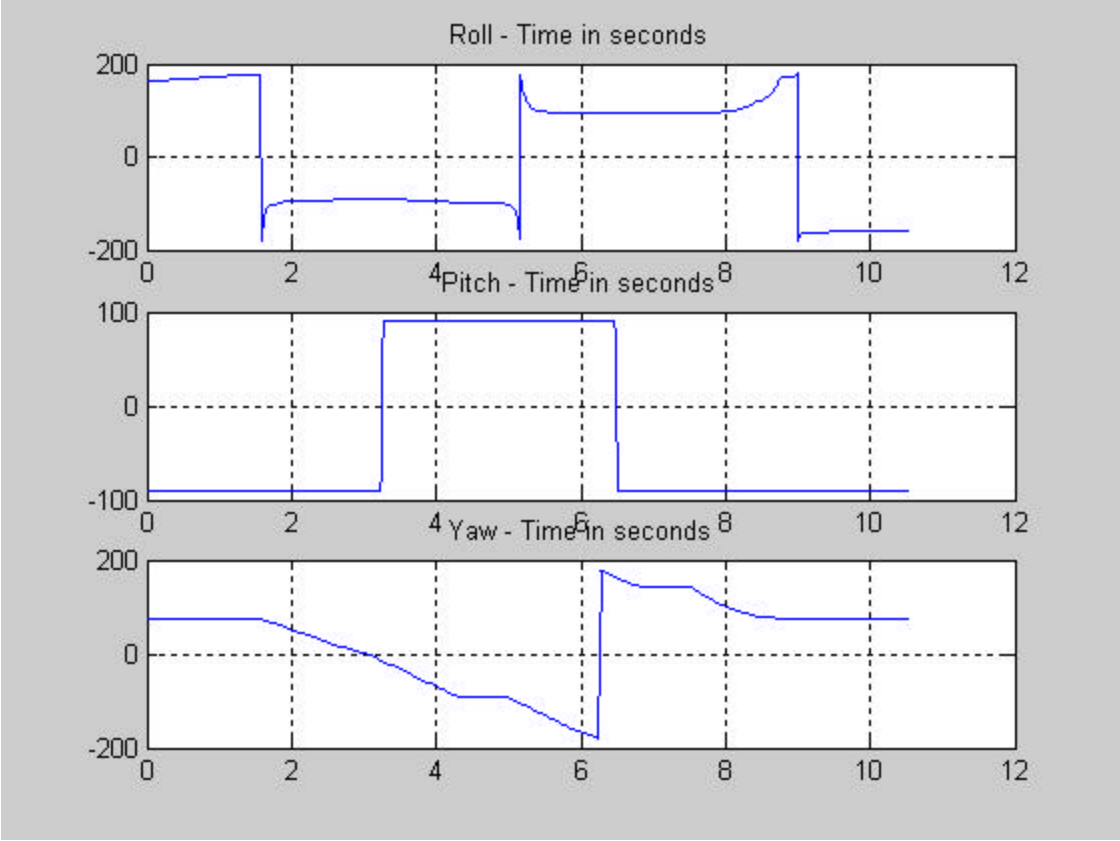

Fig - (Roll, Pitch, and Yaw for 360 yaw rotation. New angle adjustment function.)

As can be seen, the yaw did move approximately 360 degrees, but the roll and pitch also changed. This is a step backwards as compared to last weeks results. First of all, the pitch starts at -90 degrees, which means it's facing down. The roll started at 180 degrees, which would mean the IMU was upside down. Also, the roll moved quite a bit when it did not move during testing. So we believe there is something wrong in the angle adjustment function. We commented it out and ran the old Euler_init function to see if we made any changes that made a difference. The program ran as expected from last week. In fact, the height was improved because we fixed the gravity computer. So this verifies that there is something wrong in the angle_adjust function. We checked the code again and it was all correct.

Dr. Ahn came down and looked at our data. He looked at Cnb and it is way off:

39.0635895488246	128.381004065431	45.6837279742062
127.857652386099	-39.0902847125402	-1.13439625565782
-2.03264438684778	-4.87507997116882	-2.78127248842018

Then he looked at the norm(Cnb). It should be 1.

141.885459090124

This proves that the angle_adjust program is not working properly. We figured out that this is due to the noise and bias of the IMU. The angle_adjust function works by comparing the angular rates of the IMU with the rotation of the earth. The rotation of the earth is on the magnitude of 10^-5, but just noise alone is on the magnitude of 10^-1 or 10^-2. So there is no way we would be able to detect the rotation of the earth with our IMU. So we decided we will have to go back to our original euler_init function.

The second half of the lab period we worked on the testing platform. We got the entire base and uprights to hold the axis together. We had to measure the IMU, laptop, and the battery pack first, so that we could plan out how much of the wood we had to cut. All we have to do now is wait for Dave Miller to grind the metal rod down and attach it to the metal plate that will hold the IMU. Also, he will have to cut the metal rod that we will need to run though the gears.