Accelerometer-Based Inertial Navigation

Matthew Beckler

Advisor: Charles Rennolet

Senior Honors Design Project 2007-2008
University of Minnesota
Electrical and Computer Engineering Department

May 9, 2008

Abstract

While continually improving, micro-electromechanical (MEMS) gyroscopes remain relatively expensive and somewhat inaccurate. To create a reliable, inexpensive inertial navigation system, we will be developing and constructing a system that uses only MEMS accelerometers. Multiple 3-axis accelerometer units will be oriented in an intelligent pre-defined arrangement, and their raw data values interpreted to provide estimates for the position and orientation of the overall device.

Contents

List of Figures

1 Introduction

Inertial-based navigation is a system designed to provide navigation information using only direct measurements of acceleration, without any external measurements. Before the advent of global positioning systems, such as the United State's GPS or the EU's Galileo system, inertial navigation was relied upon to provide accurate position data for a number of vehicles, including guided missiles, aircraft, submarines, and spacecraft.

To conceptualize the ideas of inertial navigation, it is illustrative to consider a blindfolded passenger in a personal vehicle. The passenger cannot see any reference points outside the vehicle. As the vehicle accelerates forward, the passenger can feel himself being pressed back into the seat. If the driver goes around a corner or over a hill, the passenger can also feel these accelerations. The passenger can also keep track of which way the vehicle is turning (measuring the angular acceleration), by feeling the car turn left and right around corners, or tilt up and down a hill. If the passenger had some way to quantitatively measure these accelerations, and was able to do so rapidly and accurately, he could make a reasonable estimate of the vehicle's position at a later point in time. This is the idea behind inertial navigation.

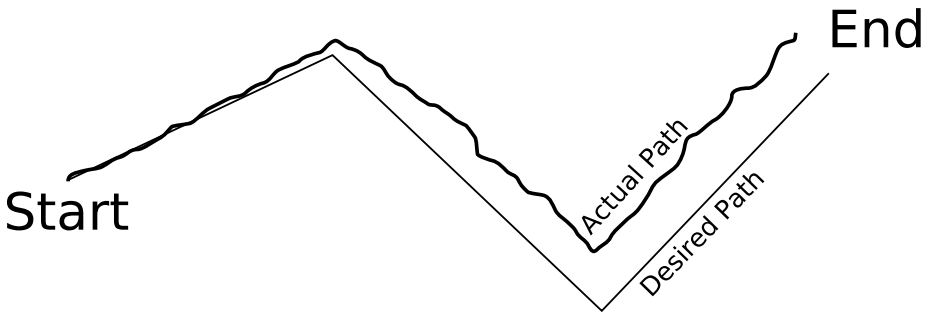

Figure 1: Integration Drift

The accuracy of an inertial navigation scheme is a function of the accuracy of your sensor inputs and the frequency of data capture. To calculate the position, the acceleration samples must be integrated twice to obtain position. These integrations can introduce errors in the data, known as *integration drift*. The problem stems from the fact that small errors in the acceleration measurements are integrated into larger errors as the time progresses and distance increases.

There are two major measurement types that need to be made in a successful inertial navigation system: Linear acceleration with respect to the sensor package's inertial reference frame, and the angular acceleration of the inertial

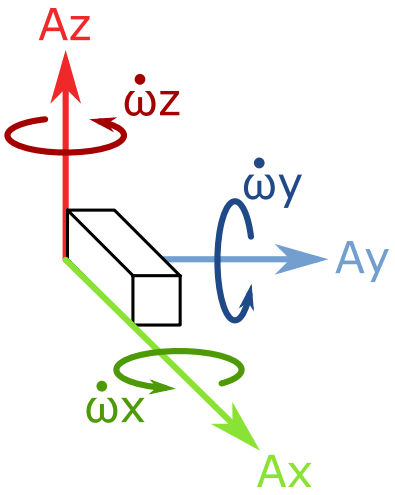

Figure 2: The Six Cannonical Accelerations

reference frame with respect to the rest frame. These six cannonical accelerations are displayed on page 4. Accelerometers are easier to produce, and high quality microelectromechanical system (MEMS) accelerometers are available for purchase from a number of manufacturers. To measure the angular acceleration, there are a couple of solutions. Traditionally, a gyrostabilized gimbal platform was used to measure the orientation of the vehicle with respect to the rest frame. This platform is a pivoted support that is allowed to rotate freely in three dimensions.

A simplified 3-axis gimbal is shown on page 5. The orange platform is stabilized by two gyroscopes of identical angular momentum and angular speed. To cancel the gyroscopic precession, the gyroscopes are mounted at right angles to each other. Three linear accelerometers are also mounted on the same stabilized platform, and measure the vehicle's acceleration with respect to the original (reference) coordinate system. These linear accelerations can be transformed into the inertial frame by reading the angles from the gimbal bearings, and calculating the orientation of the inertial frame.

A major problem with a gimbal system is a phenomenon called *gimbal lock*, which is occurs when the vehicle's rotation causes two of the three gimbal rings to align their pivot points in the same plane. This reduces the system's degrees of freedom, and the gimbal will no longer be able to rotate to maintain

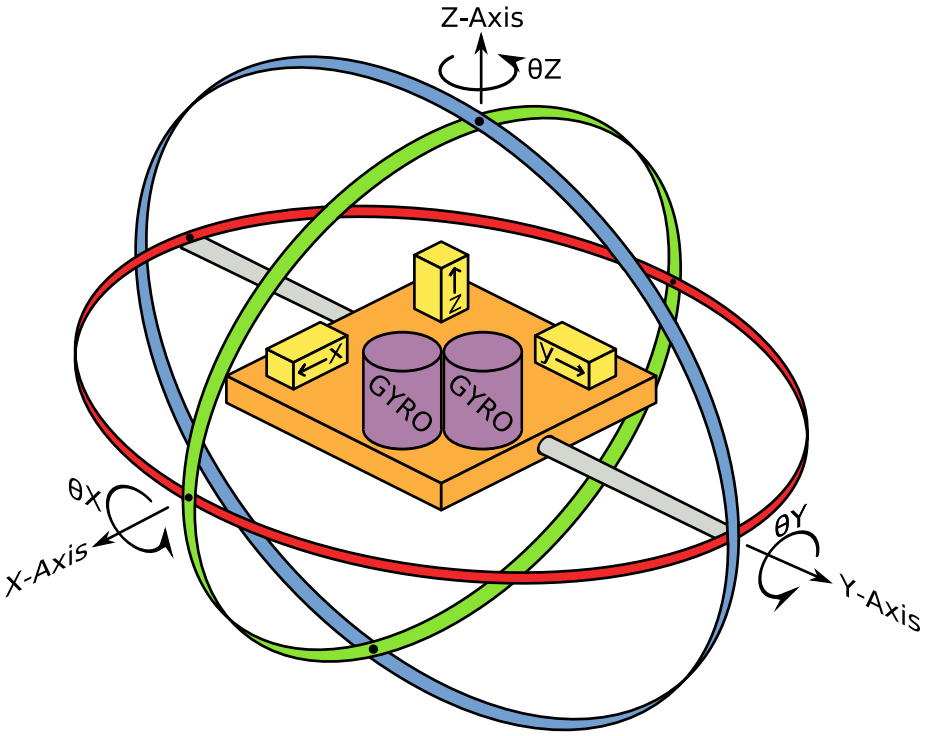

Figure 3: 3-Axis Gyrostabilized Gimbal Platform

the platform's orientation. For example, using the image above, if the platform were to rotate about the X-axis so that the green and red rings were together in the same plane, there would be no way to rotate about the original y-axis. Gimbal lock can be avoided by adding a fourth ring around the outside, and adding driver motors that continually adjust the gimbals to maintain a large angle between ring pivots.

The major disadvantages of using a gimbal platform are that mechanical systems are much more susceptible to failure, as well as being very precise machinery, and therefore expensive. A newer system is called a "strapdown" inertial navigation system, where the sensor package is simply strapped to the vehicle. Instead of a gimbal platform, small sensors called "rate gyroscopes" are used to measure the angular acceleration. These sensors work in a similar fashion to the linear accelerometers, in that they provide an instantaneous reading of the angular acceleration. This value must be recorded and integrated over time to calculate the vehicle's orientation, and is subject to the same accuracy and sample frequency problems as the linear acceleration sensors. Unfortunately, these gyros are much more expensive than linear accelerometers, and won't be used

for this experiment. Our challenge is to develop a working inertial navigation system using only linear accelerometers.

2 Prior Work

There is a small body of related work in this area. United States Patent #6308134, "Vehicle navigation system and method using multiple axes accelerometer", granted to Croyle, Spencer, and Sittaro, have proposed a system of finding longitudinal (front-to-back) and lateral (side-to-size) acceleration information for a standard passenger vehicle. This system relies on the greatly limited degrees of freedom of a passenger vehicle, namely being a primarily two-dimensional model, and only needing to measure at a minimum two values, since a vehicle cannot rotate about it's center axis without any translation. The longitudinal acceleration is used in combination with the lateral acceleration to calculate a vehicle's changing position. The problem we propose to solve will be more general than this, and will handle the whole problem in three-dimensional space. [1]

Peng and Golnaraghi of the Dept. of Mechanical Engineering, University of Waterloo, have a paper, *A Vector-Based Gyro-Free Inertial Navigation System by Integrating Existing Accelerometer Network in a Passenger Vehicle*, where the propose utilizing the already-existing network of MEMS accelerometers in a passenger vehicle as a network for inertial measurements. Traditionally, each sub-system of a vehicle, such as anti-lock braking, anti-theft system, rollover control, etc, provided its own accelerometer suited to the specific problem, with many more sensors than are technically required. By utilizing the spread-out nature of this network of inexpensive accelerometers, the authors provide a way to calculate the vehicle's angular accelerations without the use of the more expensive gyrometers. [2]

Gebre-Egziabher, et. al, from the Department of Aeronautics and Astronautics at Stanford University published a paper on *A Gyro-Free Quaternion-Based Attitude Determination System Suitable for Implementation Using Low Cost Sensors*. In it they describe how to use a quaternion formulation of Wahba's problem, along with a time-varying Kalman filter implementation, to determine a vehicle's attitude using inexpensive sensors. They also test the presented algorithms on real and simulated data from a maneuvering aircraft. Earth's magnetic field is used in the initialization of the accelerometers' alignment, and is measured by a series of magnetometers. Hybridization with GPS correction was also addressed, to great success. [3]

Luethi and Moser, at the Electronics Laboratory of the Swiss Federal Institute of Technology, Zurich, Switzerland, designed and built a "Low Cost Inertial Navigation System," using a laptop PC, a PCMCIA DAC, and a variety of accelerometers and gyrometers. They chose to use a PC instead of an embedded solution to enable them to concentrate on the inertial and sensor challenges of the project, instead of struggling with limited storage and processing power.

They too utilized a GPS as part of their research into a hybrid GPS/inertial project. An interesting feature of this project was a temperature sensor that was used to investigate the sensor's sensitivity to changes in temperature. [4]

3 Progress

Our approach towards the development of this system is based on the following idea. If we are avoiding the use of gimbals or MEMS gyrometers, then we must add extra accelerometers to try and capture the three additional degrees of freedom beyoned the the three linear accelerations easily measured by a single three-axis accelerometer. We have developed a hardware system consisting of three, 3-axis accelerometer units, a PIC18F4520 microcontroller, and a secure digital (SD) flash memory card. Each axis of the accelerometers will output an analog voltage. The microcontroller will sample the analog voltages at a regular interval, and store their sampled values in memory until enough have accumulated to write them to the SD memory card. PC software is used to post-process and analyze the recorded accelerations.

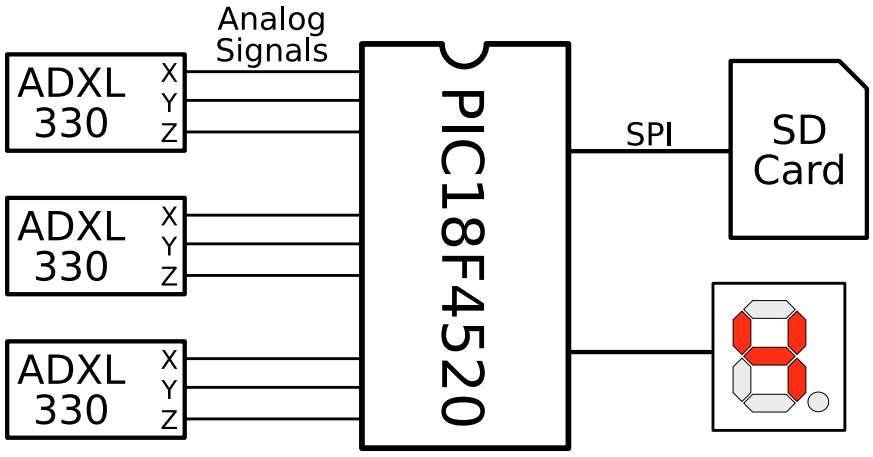

Figure 4: Block Level Circuit

To simplify the calculations and conversions from the individual accelerometer's coordinate system frame to the global coordinate frame, we have chosen to arrange the accelerometers as shown in the figure on page 8. The accelerometers are all in the plane of the circuit board, with units 2 and 3 sharing the x and y axes respectively with unit 1.

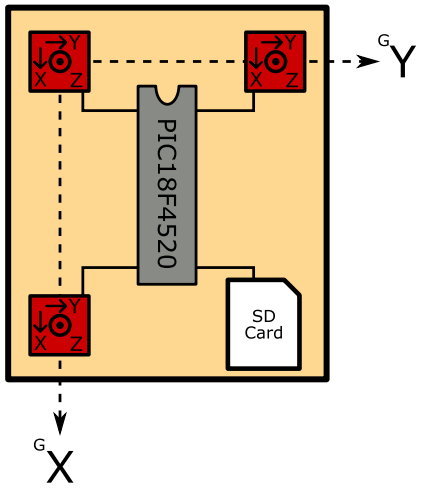

Figure 5: Layout of Accelerometers

A major part of the project has involved the development of the embedded software to run on the microcontroller. We require high precision measurements, both in terms of sampling accuracy, but also time-step accuracy. When we perform the integrations to calculate velocities and positions from these measured accelerations, we need to have a very regular time interval between successive samples. A sample rate of 50Hz has been chosen for each sensor. Since there are nine analog values to sample, on for each of the measured accelerations, we need to sample these nine values 50 times each per second, meaning that we need to perform and analog to digital conversion (ADC) 450 times per second. This corresponds to a period of 2.222 ms. Using the CCP2 module with Timer 1, we have created an interrupt schedule that does just this. Using a 32Mhz clock as our F_{OSC}, with a 1:8 prescaller, we have a native Timer 1 period of $1\mu s$. We set the CCP2 period registers to the value 2222, which gives us a time of 2.222 ms. After this interrupt occurs, the next AD conversion is automatically started, using the channel previously selected. Using $T_{AD} = 64 \cdot T_{OSC} = 2\mu s$, we know that our AD conversion will be well finished by the time the next interrupt fires, because each AD conversion requires $14T_{AD}$ cycles for each conversion, which takes $28\mu s$. The remaining time marked as "Aquisition Time" in the diagram is used to periodically save the data to the SD memory card.

SD flash memory cards can be accessed using the standard four-wire SPI bus, which is used for this project. The microcontroller has a built-in SPI module, saving us much time and complexity, as we can avoid "bit-banging" an SPI interface. Thanks to the numerous versions and revisions of the SD card specification, along with manufacturers not always following the standard to the letter, initialization of an SD card can be tricky. We have developed a series of

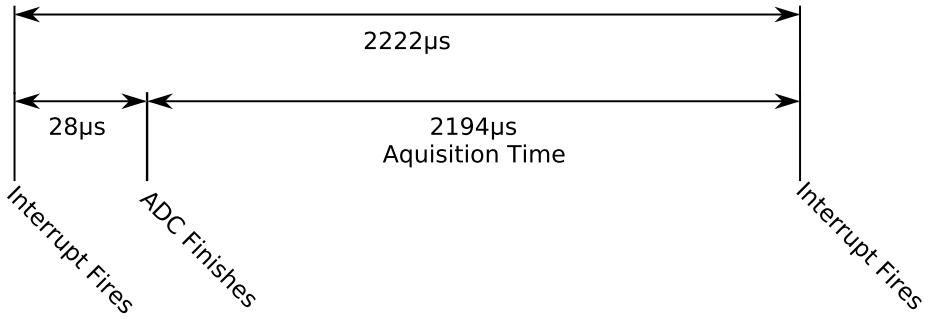

Figure 6: Timing of Interrupts and AD Conversion

accessor functions to help interface with the SD card through the SPI interface:

- `SPIGetSend(unsigned char sendData)` - Sends and receives a byte of data using the SPI interface.
- `SD_GetStatus(uint timeout` - Gets the status response from the SD card. Returns the first non `0xFF` value received, or `0xFF` if a timeout occurs.
- `SD_SendCommand(char command, long param, uint timeout)` - Sends a command and optional parameter to the SD card. Calculates the checksum as appropriate. Returns the card's status response.
- `SD_GetData(char* buffer, uint numBytes)` - Retrieves a specified number of bytes of data from the SPI into the provided buffer.
- `SD_ReadBlock(uchar* data, uint blockAddress)` - Reads a 512-byte block from `blockAddress` into the `data` buffer.
- `SD_WriteBlock(uint blockAddress, uchar* data)` - Writes a 512-byte block from the `data` buffer to `blockAddress`.
- `SD_CardInDataMode(void)` - Uses CMD13 to check if the card has already been initialized into data mode.
- `SD_InitCard(void)` - Brings the card out of init mode and into regular data mode.
- `SD_ConfigCard(void)` - From data mode, this function reads the card's properties, such as hardware sector size, number of hardware sectors, total size, etc.

These funtions allow us to easily initialize an SD card, query the card to find it's size and features, and write blocks of data to the card. The minimum block size for an SD card is 512 bytes, and this is the block size we have chosen

to use.

One side-effect of this 512-byte block size is that we need to save up 512 bytes before writing a block to the SD card. This can be easily implemented with a circular buffer, which is a memory structure that consists of two pointers, called *start* and *end*.

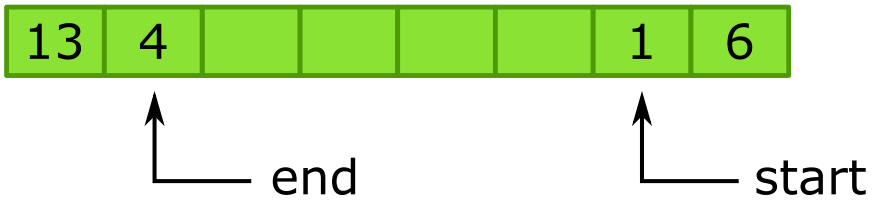

Figure 7: Circular Buffer

The pointer *start* points to the start of valid data, and data removed from the buffer is taken from *start*. Data that is added to the buffer is added one position past *end*, which points to the most recently added piece of data. Some implimentations call these pointers *tail* and *head*, respectively.

To aid the user, we have added a very simple user interface to the data logging hardware. There is a main power switch, which enables the voltage regulator, and starts up the main processors. The other switch is called the "logging enable" switch, and will start the sampling clock and SD interface when switched on. This allows the user to temporarily pause recording by switching it off, and then resuming logging at a later time, with recording resuming at the SD block number if left off at.

One concern with a portable data-logging project such as this would be the amount of storage required for long-term recordings. For this project, we are sampling with a 10-bit ADC, so our results are stored in a 16-bit word. We generate 9 of these samples 50 times per second, so this is a total of 900 bytes per second. For an hour of recording, we require 3.09 miB, for a day's recording, 74.16 miB, and a whole year of recording would require 26.43 giB of storage space. The 1 giB card used for development and testing could store data for more than 13 days. In reality, the noise inherent in the system would make such long-term recordings impractical, but the small space requirements are certainly an advantage.

As noted above, the accelerometers output analog values, which much be

sampled by an analog-to-digital converter. The analog outputs are called "ratiometric", meaning that they output a voltage proportional to the acceleration seen. For example, when using a 10-bit ADC, where the output values range between 0 and 1024, a $0g$ reading should correspond to a sampled value of 512, exactly in the middle of the range. In reality, this is not the case, as process variation and other factors making the $0g$ position to be different for all sensors. Also, the amount of voltage change for a given change in sensed acceleration is not constant for all sensors. In order to fully characterize our sensors, we constructed a rotating assembly to apply a controlled and calculable acceleration to the sensors.

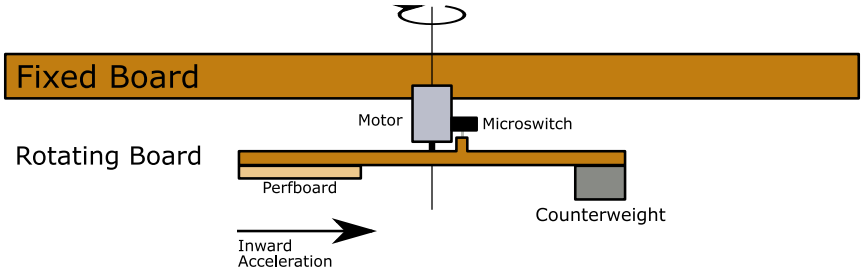

Figure 8: Rotating Assembly

The assembly, as seen in the figure on page 11, consists of a motor attached to a fixed cross-beam, with a freely swinging beam with the project's sensors attached to it, with a counterweight to reduce the cross-torque on the motor shaft. The rotating part is spun to a constant angular velocity, as measured by the microswitch. We used two angular velocities, corresponding to running the motor from 5v and 12v. The motor exhibited very good steady-state speed control, and no external speed control was required to maintain the angular velocity within 1%.

Using the newly-constructed rotating assembly, we used it to apply a constant and calculable acceleration to the accelerometers. In the figure on page 12, we have plotted the raw accelerometer data from each of the nine sensors. The combined plot is in the lower-right corner, with the three accelerometers split up in the other three plots.

One thing to observe is that for the three individual plots, the same axes are displayed with the same color, meaning that the x-axes are all blue, the y-axes are green, and the z-axes are red. For this data, we see that the y-axis values are much smaller comared to either x or z. This is because the force of gravity was acting in the negative y-axis, and it i expeiencing the standard $1g$ of acceleration. Based on the arrangement of the sensors we used, we know that

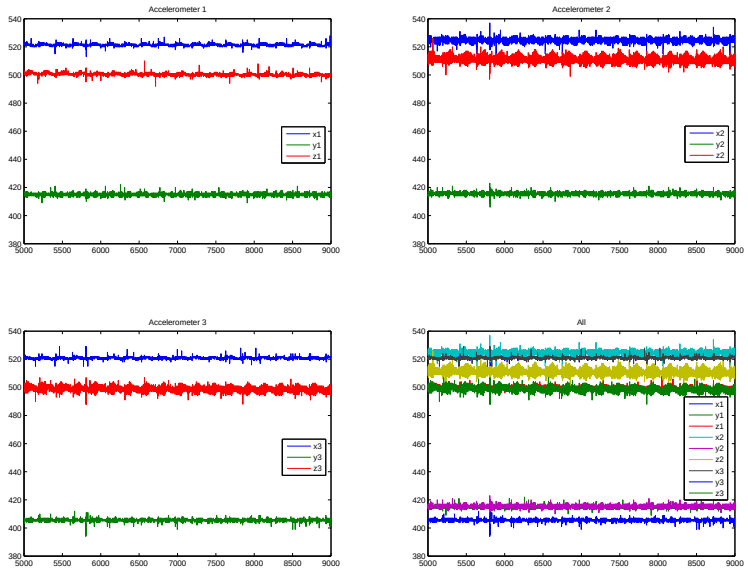

Figure 9: Raw Accelerometer Data

the x-axis, sitting on top of the plot, was perpindicular to gravity, and is seeing $0g$ worth of acceleration. The acceleration measured by third axis, the z axis, has been reduced from the $0g$ point by the inward centripetal acceleration. This is the acceleration we can control, by running the motor at a different angular velocity, or by moving the sensors radially inward or outward. By measuring the angular velocity and the radius of the sensor, we can calculate the velocity that should be seen. By comparing this to the difference between the measured value and the $0g$ value, we can find the ratio of ADC value to actual, real-world acceleration. We perform this task for all of the sensors involved.

Now, we will discuss the exact procees used to find these conversion values, using the example of the Y-axis sensors. During this test, the accelerometers were arranged with $Y1$ and $Y3$ located closer to the center at a radius of 0.1323 meters. Accelerometer $Y2$ was located further from the center at 0.1663 meters. Previously to this test, we aligned the y-axes to be orthogonal to the ground, and recorded the $0g$ readings. They are recorded in the data table below. We also performed another set of readings with the y-axis sensors aligned with gravity, to capture the readings for $1g$.

$0g$ and $1g$ Values			
	Y1	**Y2**	**Y3**
$0g$	521.8185	523.5429	510.8543
$1g$	415.4131	416.0541	405.9174

Two angular velocities were tested, one at $1.62\frac{rad}{s}$ and one at $2.92\frac{rad}{s}$. For each of these, we performed five independent trials, starting at a slow velocity and allowing the motor to accelerate the sensors up to full speed. When steady state was reached, we recorded the data for 10,000 samples. This steady-state data was averaged over time, and left us with the ADC sample value that corresponds to the acceleration applied. We will demonstrate the process with a single trial at the slower angular velocity, $1.62\frac{rad}{s}$. These values are shown below:

Raw Values		
Y1	**Y2**	**Y3**
508.0780	506.1506	497.1521

From these three raw values, we need to subtract the individual $0g$ offsets:

Raw Offsets		
Y1	**Y2**	**Y3**
13.7405	17.3923	13.7022

At this point, we need to divide these values by the calculated acceleration, to find the ratio of the sampled offset value to the actual accelerations. For $Y1$ and $Y3$, we have $\alpha_{1,3} = 1.2969\frac{m}{s^2}$, and for $Y2$, we have $\alpha_2 = 1.6294\frac{m}{s^2}$:

Conversion Factors		
Y1	**Y2**	**Y3**
10.5953	10.6742	10.5657

Using a similar process for the $1g$ sampled values, we obtain another source of conversion factors:

Conversion Factors ($1g$)		
Y1	**Y2**	**Y3**
10.8577	10.9682	10.7078

We performed five trials for each of two angular velocities, in addition to using data from the steady-state $1g$ measurements to produce the following data.

Calibration of y-axis sensors			
	Y1 (522)	**Y2 (523)**	**Y3 (511)**
$\omega = 1.62\frac{rad}{s}$	10.5953	10.6742	10.5657
$\alpha_{in} = 0.13g$	10.5600	10.6180	10.4809
$\alpha_{out} = 0.16g$	10.6112	10.6856	10.5772
	10.6388	10.7126	10.5902
	10.6094	10.6795	10.5273
$\omega = 2.92\frac{rad}{s}$	10.6818	10.7425	10.6412
$\alpha_{in} = 0.43g$	10.6818	10.7761	10.7137
$\alpha_{out} = 0.53g$	10.7569	10.8094	10.7287
	10.7003	10.7333	10.6531
	10.7396	10.7952	10.6967
$\alpha = 1g$	10.8577	10.9682	10.7078
Mean	10.6757	10.7450	10.6257
StdDev	0.0862	0.0936	0.0831

Something interesting to note about the data values is that as the applied acceleration increases, the converson factor also increases slightly. This is cause for concern, as we would have prefered to have the conversion factors stay constant over the entire range of applied accelerations. When considering the effect of this non-linearity in the response of the accelerometers, we need to take a look at what exactly the conversion factors tell us. Here, with a conversion factor of, say, 10.6, this means that a change of 10.6 in the sampled output indicates a change in measured acceleration of $1\frac{m}{s^2}$. This means that we only have 10 bit-values available to represent each change of $1\frac{m}{s^2}$, or that we have a resolution of $0.1\frac{m}{s^2}$. This resolution may be much too large, but when considering the effect of the 0.1 non-linearity seen in the table, the non-linearity appears to small enough in magnitude to be unimportant.

After characterizing all the sensors, we then moved on to recording experimental data of real-world activities. As an example of this, the accelerometer's perfboard was mounted to a carrier board, and the author walked around his abode while carrying the sensor board at waist level. The raw data from this exercise is shown on page 15. The data is presented in the same form as the sensor characterization data above. Notice the large range of the red axis, which is the z-axis data. This acceleration is caused by the author's individual footsteps as he walked through the yard. You can also notice a decrease in scale of the accelerations around step 500 and 950, which were caused by the author's brief pause at the first two corners of the house.

An issue with the sensors is the presence of noise, which has the characteristics of white Gaussian noise, equal in contribution at all frequencies. We can perform a simple moving average filter with width 10, to eliminate much of this noise without removing the data in the signal. The figure on page 16 shows both the pre- and post-filter data.

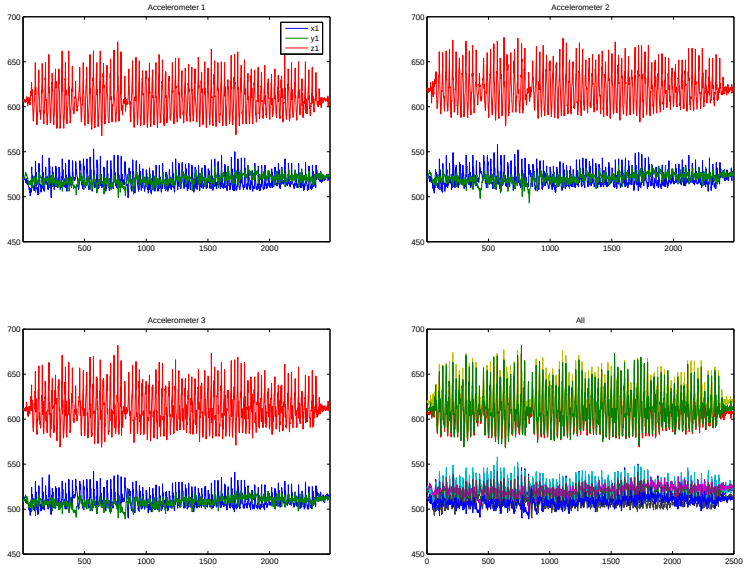

Figure 10: Experimental Raw Data - Walking Raw Data

4 Summary of Changes

Initially we had provisions for attaching extra external RAM to the processor to store the acceleration samples before writing it to the SD card. After further analysis and development of the project, it was decided that given the generation rate of samples (900 bytes / second), and the time required to write 512 bytes to the sd card (under 10 ms), there was plenty of time and internal RAM to not require the external ram. We have yet to find a cool project to make use of the extra RAM chips that were sampled.

5 Changes to Budget

Many of the parts for this project were already in hand or readily availabe for free or very cheap. The largest outlay of funding was for the three, 3-axis accelerometers and their breakout boards, a total of about $100. Processors can be sampled from the manufacturer for university student projects, and the other supplies are common to any lab kit. The prototype board and mounting hardware was purchased at the local RadioShack in an effort to save time and avoid the week delay associated with using the department parts shop. Due to the rel-

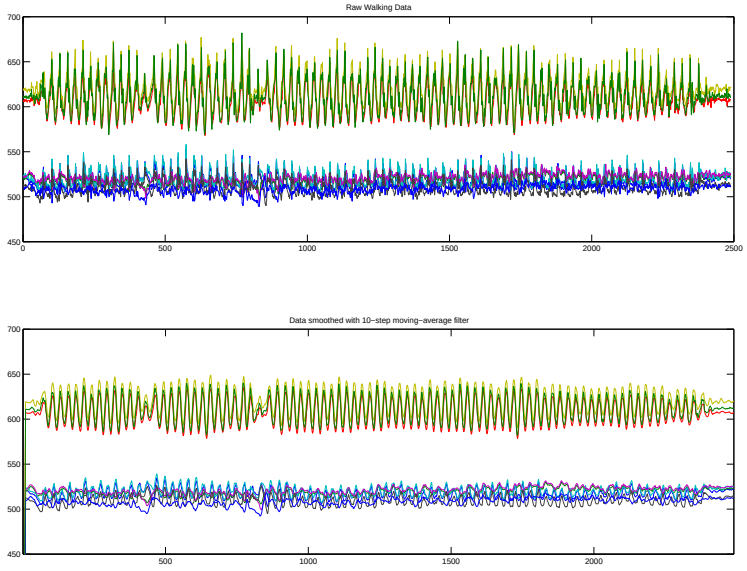

Figure 11: Experimental Data - Walking Filtered Data

ative simplicity of the project circuit, we decided to go with a "perfboard" type solution, both to save time and work associated with the schematic capture and layout creation, but also to save hastle messing with the boardshop. After bad experiences with the boardshop during the second junior year lab, we decided that the advantages of a PCB, such as reduced noise and general appearance, were outweighed by the advantages of simply using perfboard, which include cost, time delay, and local availability.

The construction of the rotating apparatus required mostly small lumber from the local home improvement store. It actually turned out to be more difficult to construct than anticipated. The author nearly forgot that he doesn't have access to a garage full of tools like at his parents' house, and ended up having to make due with a limited selection of tools and fabrication methods. In the end, things worked out just fine. Hopefully the landlord doesn't mind some extra sawdust in the corners.

6 Conclusion

A data-logging hardware system consisting of three, 3-axis accelerometer IC's has been developed. An industry standard SD flash memory card is used for

long-term, non-volatile store of recorded data. Through the construction of a controlled rotating assembly, we were able to apply arbitrary accelerations to the sensors, and fully characterize their numeric converstion factor.

7 Further Work

Due to time restrictions, the final software for analysis of the acceleration data was not completed. Given that we have found the conversion factors for all acceleration sensors in the project, we can easily convert the data from ADC sample values into actual accelerations expressed in $\frac{m}{s^2}$. We also know the δt value, as that is set to $\frac{1}{50Hz} = 0.02s$ by the ADC timer. Having these two items, it should be relatively straightforward to develop software to perform the numerical integrations to bring the acceleration values to velocity values, and integrate once more to calculate the change in position.

8 References

[1] Croyle, et. al, "Vehicle navigation system and method using multiple axes accelerometer", U.S. Patent 6,308,134, October 23, 2001.

[2] Peng, Y.K., and Golnaraghi, M.F., "A Vector-Based Gyro-Free Inertial Navigation System by Integrating Existing Accelerometer Network in a Passenger Vehicle," Proceedings of the IEEE Position Location and Navigation Symposium (PLANS), pp. 234-242, Apr. 2004.

[3] Gebre-Egziabher, D., Elkaim, H K., Powell, J.D., and Parkinson, B.W., "A Gyro-Free Quaternion-Based Attitude Determination System Suitable for Implementation Using Low Cost Sensors," IEEE PLANS, pp. 185-192, 2000.

[4] Luethi, Peter, and Moser, Thomas, "Low Cost Inertial Navigation System", Website Address: http://www.electronic-engineering.ch/study/ins/ins.html, Accessed Oct. 7, 2007.