

# Inertial measurement and realistic post-flight visualization

David Fifield  
Metropolitan State College of Denver  
Keith Norwood, faculty advisor

June 28, 2007

## Abstract

Determining the position and orientation of a moving object is one of the principal challenges in aerial and space flight. A related challenge is interpreting recorded position and orientation data after a flight; that is, transforming a sequence of numbers into something easy to understand.

An investigation into these challenges and methods of overcoming them, as part of the DemoSat IV program, is presented. A DemoSat payload containing an inertial measurement unit and a recording device was flown and its measurements analyzed. These data were filtered and processed to make a three-dimensional, real-time, realistic visualization of the motion experienced by the payload during its flight.

A design for an inexpensive and effective inertial measurement unit is presented. The raw acceleration and rate of rotation measurements from an actual DemoSat flight are given, along with a description of the algorithms used to filter them. Still frames from the post-flight visualization are shown and information on how to obtain the complete video is given. References are provided to the source code of the software created for the project, all of which is available to the public.

Three-dimensional, real-time, realistic visualization is an intuitive way to understand the qualitative behavior of an object during flight. Inertial navigation on its own is not sufficient for determining an object's absolute position and direction over a long time. However, inertial measurements allow the recreation of an object's orientation and rate of rotation, which is not as easily done using other methods.

## 1 Inertial navigation

In order to understand the information in the rest of this report, it will be necessary to have some understanding of inertial navigation. Inertial navigation is a way of determining an object's position and orientation by measuring its acceleration and rotation.

Inertial navigation is made possible by a group of sensors called the inertial measurement unit. While several designs for an inertial measurement unit are possible, the most common implementation consists of six sensors: three accelerometers, which measure linear acceleration; and three gyroscopes, which measure angular velocity (rate of rotation). The sensors are arranged in pairs along three mutually perpendicular axes. This design allows for easy interpretation of measurements into acceleration and rotation in the three dimensions of space.

## 1.1 Integration and noise

The sensors in the inertial measurement are sampled periodically. Each of the many measurements taken represents a rate of change of one of the physical properties associated with the object being measured. These rates of change can be added together over time to derive the total change experienced by the object.

This process is exactly numerical integration; acceleration is integrated twice to give position, and angular velocity is integrated once to give orientation. The object's initial position, velocity, and orientation serve as the constants of integration. In theory, the initial conditions and a continuous description of small changes is all that is needed to recreate an object's motion exactly.

The theoretical ideal is not achieved in practice because the readings from the inertial measurement unit are inherently noisy and discontinuous. Small errors accumulate; the approximation given by integration becomes worse as time goes by. How quickly the approximation diverges from the true values depends on the quality of the sensors, the sample rate, and other factors. The inaccuracy increases faster the more the samples are integrated. The error grows linearly when integrating angular velocity to get orientation but quadratically when integrating acceleration twice to get position.

## 2 Method

We, the students of the Metropolitan State College of Denver DemoSat team, conducted a practical investigation into the possibilities of inertial navigation as part of the DemoSat IV program. DemoSat [2] is a program in which students build a payload designed to perform some scientific mission. The payloads are flown to about 100,000 feet by a balloon, then dropped back to earth on a parachute.

We chose to complete a mission called FieldSat, whose objective is this: "On-board sensors record orientation and vibration information during flight." We accomplished this mission by building a payload containing an inertial measurement system. We flew the payload and recovered its data. These were filtered and integrated to recreate a realistic depiction of the flight.

This is an overview of only the parts of our project that are relevant to this paper. Complete details can be found in our project final report [7].

## 2.1 Payload design

The heart of the FieldSat is a microcontroller. We chose the ATmega32 microcontroller from Atmel, primarily because of its easy programmability using standard tools. It has built-in analog-to-digital converters that are suitable for reading the signals from the sensors in the inertial measurement unit.

The FieldSat's inertial measurement unit consists of three angular rate gyroscopes and three linear accelerometers. We used the ADXRS300 angular rate sensor and the ADXL320 two-axis accelerometer, both from Analog Devices. These sensors produce an output voltage that is proportional to the rate of rotation of linear acceleration. They are very small (less than 5 mm on all sides) and lightweight.

The inertial sensors are arranged in pairs consisting of one accelerometer and one gyroscope. The pairs are mounted on independent auxiliary circuit boards that are connected by wires to a main circuit board. These boards are mounted on the interior walls of the package on mutually perpendicular axes.

The microcontroller records its data on two Secure Digital (SD) cards. (This type of memory card socket is commonly found in digital cameras.) Having two cards rather than one provides some degree of redundancy.

## 2.2 Programming

The microcontroller runs a custom program created for this project. It is written in C using the AVR Libc [1] C library. The program samples the inertial sensors 100 times per second. Each analog measurement is converted to a 10-bit digital value and written to the SD cards. The source code [3] for the payload firmware is available to the public.

## 2.3 Stability and filtering

Once collected, the inertial measurements need to be integrated to recreate the total change in position and orientation experienced by the payload. We started with a simple first-order Euler integrator; however, our experiments using this method showed it to be highly unstable. For example, a model of an object that had been perfectly still would start to rotate faster and faster until it was spinning wildly. Our attempts to derive position by integrating acceleration twice had the object zooming off to infinity in one direction or another after a brief period of stability.

The primary problem is noise. The inertial sensors have small fluctuations in their output even when perfectly still. Using the readings without filtering out these fluctuations results in a jittery and unstable simulation.

We were eventually able to partially solve these stability problems through filtering. We used essentially the same filter for each of the data streams: an exponential moving average. A reading  $x_t$  is taken to be a weighted average of a sample  $s_t$  and the previous reading:

$$x_t = x_{t-1} + \alpha(s_t - x_{t-1})$$

This averaging happens independently for each sensor on each of the three axes. The smoothing parameter  $\alpha$  controls how quickly the filter reacts to changes in the input. Values near 0 produce a smooth but slow-reacting filter, which values near 1 produce a jerky but fast-reacting filter. Through experimentation with the simulation we determined appropriate values for the smoothing parameters for each kind of inertial measurement. For angular velocity we used  $\alpha = 0.2$  and for short- and long-term acceleration we used  $\alpha = 0.3$  and  $\alpha = 0.05$  respectively.

Why two averages for acceleration? We originally wished to recreate the flight in all aspects, including position. This was to include producing a three-dimensional map of the path taken by the payload during flight. Ultimately, twice-integrating the acceleration data proved to be far too unstable to give meaningful position information for even a minute, let alone the two hours of an entire flight. So instead of using acceleration to derive position, we used it to get some idea of vibration. We keep track of two averages for the current acceleration vector, one that responds quickly to changes ( $\alpha = 0.3$ ) and one that responds slowly ( $\alpha = 0.05$ ). The difference between the the first vector and the second is used as the vibration vector.

Filtering out noise is not the only way to improve stability. There is some redundancy in the measurements from the six sensors, allowing us to check one sensor against another and keep both more in line with reality. We improve stability by reconciling two differing descriptions of which way is “down”: the one reported by the accelerometers and the one reported by the gyroscopes.

The accelerometers of course measure linear acceleration, but they also have the side effect of measuring the gravitational field. The acceleration vector reported by a stationary three-axis accelerometer will point along the gravitational field; i.e., straight down. Even when in motion, the vector will on average point down as long as the motion is not too great. This provides a long-term stable measurement of one aspect of orientation, one which is not subject to accumulating integration error. At each time step the model is rotated toward the acceleration vector by 8% of the angle between them. This value was determined through trial and error. This correction made our simulation stable enough to last the duration of the flight.

## 2.4 Limitations

It may seem that with enough filtering and signal processing, integration of inertial measurements may be made as accurate as desired. Unfortunately, this is not the case. The information provided by three accelerometers and three gyroscopes in the inertial measurement unit is simply not enough to solve for all the unknowns in an object’s orientation. The simple reason for this is that there is no way to measure rotation about the gravity vector. The values reported by the sensors are identical whether the object is facing north, east, south, or west. This lack of absolute orientation information, along with integration errors caused by noise, is the main barrier to using inertial navigation as the sole means of position determination.

### 3 Results

Our DemoSat payload flew successfully, recording about two hours of inertial data. The results of the flight are the raw inertial measurements and the three-dimensional visualization created to help understand them.

#### 3.1 Inertial measurements

The most direct results of the flight are the raw inertial measurements [5]. We recovered a data file containing raw rate of rotation and acceleration information. The measurements span from before lift-off until after the lift balloon burst and the payload started to descend. The recording ended before the payload returned to earth, probably because of some mechanical malfunction.

There were two periods during which the accelerometers malfunctioned. The accelerometers measuring the Y and Z axes reported a maximum acceleration for about forty minutes. This, we believe, was caused by a loose wire. Despite these mechanical difficulties, we were able to recover enough good data to produce our visualization.

#### 3.2 Visualization

Perhaps the most exciting result of our work is the post-flight visualization we produced. We developed the filtered and integrated data into a three-dimensional, real-time, realistic recreation of the payload’s behavior during flight. The video shows graphically the rotation and vibration experienced by the payload during its two hours of operation.



The principle guiding the visualization we developed is that it should require no interpretation. It should replicate as exactly as possible the actual appearance of the payload in flight. It should proceed in real time. The modeled payload should look like the actual payload. Ideally, the visualization should appear as if the entire flight were video recorded at close range.

Why attempt to recreate the flight in this way? After all, this method of presentation loses much of the precision of the measurements. The reason is to gain an intuitive understanding of the motion of the payload. Almost anyone can understand a video of a moving object, even if the object shown is not familiar. It is one thing to see rapid oscillation of an acceleration graph, but quite another to see a three-dimensional model of an object shaking back and forth as it would in real life. Realistic visualization helps the viewer get a “gut feel” for complex information.

The graphical portion of the visualization program is written using OpenGL. A realistic three-dimensional model of the payload was created. It has the same proportions as the actual payload and it is texture-mapped using photographs. The overall picture is quite realistic; some viewers of an early version of the visualization program did not realize that the image was computer-generated.

Because of the aforementioned difficulty in integrating raw acceleration data, the visualization does not attempt to track the payload's flight path. Rather, the payload is translated on the display along the vibration vector, which is calculated according to the algorithm given above. The effect is a visible shaking during periods of vibration. So while long-term changes in position are lost, short-term changes are made visible. Especially strong vibration is evident during the launch and at the apex of the flight.

One of the DemoSat payload's other functions was to deploy a pair of solar panels. The moment when the deployment happened was recorded in the data log. The visualization simulates the deployment of the solar panels at the appropriate time. Unlike the rest of the simulation, the solar panel deployment is not necessarily realistic; there were no sensors to record whether the panels were actually deployed.

In an effort to further improve the visualization's accessibility, we put the video on a DVD so that it could be played without unusual hardware. The DVD image is available for download [8].

## 4 Conclusions

Inertial navigation on its own is not sufficient for determining an object's absolute position and orientation over a long time because of incomplete information and accumulating errors in integration. However, it is good in the short term and can reliably report the acceleration and rate of rotation over any time interval. These data give much insight into the nature of an object's flight.

Combining inertial navigation with an absolute positioning system such as GPS is an attractive prospect. GPS can provide the long-term absolute positioning that inertial navigation lacks. Short-term integration of inertial measurements can fill in the gaps between GPS readings, providing high-resolution position information as well as orientation. While GPS can be used to determine orientation [4], it is more straightforwardly done using acceleration measurements.

Communicating complex scientific information requires providing a variety of perspectives. Three-dimensional, real-time, realistic visualization is an intuitive way to understand the qualitative behavior of an object during flight.

## References

- [1] AVR Libc developers. AVR Libc home page. 2006.  
(<http://www.nongnu.org/avr-libc/>).
- [2] Colorado Space Grant Consortium. DemoSat home page. 2006.  
(<http://spacegrant.colorado.edu/demosat/>).
- [3] Fifield, David. Metro State DemoSat IV source code. 2006.  
(<http://www.bamssoftware.com/software/demosat/>).
- [4] Johnson, Andrea M. “Spacecraft Attitude Determination Using Global Positioning Satellite Signals.” *Proc. of the Colorado Space Grant Consortium Undergraduate Symposium*. 2005.
- [5] Metropolitan State College of Denver DemoSat team. Data file. 2006.  
(<http://www.bamssoftware.com/wiki/DemoSat/DataFile>).
- [6] Metropolitan State College of Denver DemoSat team. DemoSat IV home page. 2006. (<http://www.bamssoftware.com/wiki/DemoSat>).
- [7] Metropolitan State College of Denver DemoSat team. Final report. 2006.  
(<http://www.bamssoftware.com/wiki/DemoSat/FinalReport>).
- [8] Metropolitan State College of Denver DemoSat team. Flight video. 2006.  
(<http://www.bamssoftware.com/wiki/DemoSat/FlightVideo>).