5-2012

Robust and Economical Autonomous Navigation via Advanced Bayesian Methods

Ali H. Chaudry
University at Albany, State University of New York

Robust and Economical Autonomous Navigation via Advanced Bayesian Methods

Ali H. Chaudry[abc] (*Adviser*: Dr. Kevin H. Knuth[abd])

[a]*Departments of Physics, CAS, University at Albany, SUNY*
[b]*Knuth Cyberphysics Laboratory, CAS, University at Albany, SUNY*
[c]*Advanced Lithography Group, CNSE, University at Albany, SUNY*
[d]*Autonomous Exploration, Inc., 385 High Plain Road, Andover, MA*

Abstract. A 2009 NASA SBIR grant awarded to Autonomous Exploration, Inc., was partially used to fund students at the University at Albany to help research whether autonomous navigation systems can be prepared at a much lower cost than is currently being done. Expensive and bulky sensors (the basis of current-generation navigation systems) can be avoided by using commercially available low-cost sensors; the disadvantages of these low-cost sensors can be eluded by using novel data-processing techniques such as Bayesian analysis. Integration of these methods can lead to low-cost navigation systems, and hence, rovers. Potential applications include extra-planetary surface exploration as well as military deployment in situations where, at present, compulsory human supervision wastes expensive resources.

INTRODUCTION

A portion of funds from a 2009 NASA SBIR grant awarded to Autonomous Exploration, Inc. (to research and develop a ‘Lunar Surface Navigation’ system) were allocated to the University at Albany’s Physics Department to help with the research phase. At the University at Albany, this project was headed by Prof. Kevin Knuth, who also serves as the president of Autonomous Exploration, Inc. After much deliberation, the Knuth Cyberphysics Group decided to work on developing a self-governing navigation system, one that could be adapted to any sort of parent vehicle. For this reason, it was decided that a test-bed parent vehicle should be made out of readily-available commercial Lego® construction toy parts. The decision to use Lego parts was implemented to lower costs and to allow for the flexibility of altering/upgrading the parent vehicle at will, without having to design and manufacture a new vehicle for testing every time a new feature for the lunar navigation system needed to be tested.

The basis of any navigation system is the *sensor*. In fact, the basis of any interaction between the physical world and the digital world, where the input comes from the physical world and the output is needed in a digital format, is the sensor. A *sensor* is "a device that responds to a physical stimulus (as heat, light, sound, pressure, magnetism, or a particular motion) and transmits a resulting impulse"[1] – in the modern era, this is primarily as electrical output. Due to Knuth Cyberphysics Lab's prior experience with robotics, microcontrollers and sensors, a plan was formulated to integrate a diverse set of sensors as part of the navigation system. The uniqueness of this navigation system, however, lay in the *method* that the code used to process data from the sensors. The method used is Bayesian inference, an approach to probability and statistics which differs from the standard approach to probability and statistics as taught in the classroom (which is known as *frequentist inference*).

The prime feature of Bayesian inference is its use of feedback to update the probability estimate for a hypothesis as new information (evidence) is learned. In the ideal world, where events (from as small as on the atomic scale to as large as on the galactic scale) are predictable, frequentist analysis would be able to predict scenarios with a decent degree of accuracy. In a robotic system which is navigating the real world, however, Bayesian analysis has a significant advantage over frequentist analysis because in the practical world there is a certain degree of uncertainty which cannot be predicted ahead of time and leads to greater errors in frequentist analysis. Bayesian analysis is better equipped to handle these scenarios.

The navigation system was equipped with webcams (for visual odometry, a technique to identify stationary features from a source video), an inertial measurement unit (to measure acceleration in the *x*, *y*, *z* axes as well as rotational motion and acceleration), a magnetometer

[1] http://www.merriam-webster.com/dictionary/sensor

module (to measure magnetic flux density in the three axes), GPS receivers (to provide location and time information via satellite to help with timing the data – time-syncing – on all sensors), and a netbook (to govern the sensors and act as the permanent storage device for navigation runs). The use of Bayesian algorithms promises to lead to a significantly more accurate navigation system than would be possible with less costly equipment; in fact, it should give results similar to, if not better than, those provided by much more expensive hardware working in unison.

SENSORS

The computer in control of all the sensors is an *Asus Eee PC 1005HA* subnotebook/netbook with an Intel Atom ultra-low-power central processing unit (CPU) which delivers the full capabilities of an x86 desktop CPU. The CPU clock frequency is 1.66GHz.[2] This is a good compromise between high clock frequency (high power consumption requirements) and low clock frequency (excessive lead times during sensor management). The thermal design power is around 2W (which is around 3% that of an everyday CPU such as the Intel Core 2 Duo). The idle power draw is around 30mW.[3]

The on-board graphics processing unit (GPU) is also an Intel-manufactured chip, the Mobile Intel GMA 950; although unable to process 1920x1080 resolution progressive scan (1080p) high-definition (HD) video at full frame-rate, it is still competent enough to process 320x240 resolution (QVGA) standard definition (SD) video from up to four different cameras. The CPU/GPU combo can also actively get raw data in 640x480 resolution (VGA) from four different cameras and store it on the hard-disk drive (HDD), thanks to the hyper-threading

[2] http://reviews.cnet.com/Asus_Eee_PC_1005HA_white/4505-3121_7-33698891.html
[3] http://www.pcpro.co.uk/reviews/processors/202845/intel-atom

capability of the processor. The group has also discussed replacing the HDD with a solid-state drive (SSD), which, by virtue of the fact that it has no moving parts, will reduce power consumption, help with heat management, and increase read/write capability of the netbook by up to ten-fold. This speed far bypasses the capability of the processor to process data for larger raw data sets. The 1005HA is as seen in Figure 1.

FIGURE 1. Asus Eee PC 1005HA.

The sensors, as they from the core of the navigation system, need some more explanation. The webcams are *Logitech C905* cameras. They form the basis of the visual odometry subsystem, which can track motion in areas where no radio beacons or navigation satellites are available. The C905's are 2-megapixel webcams capable of shooting video up to 30 frames per second.[4] Two webcams are used to simulate stereo vision, so as to receive depth data (which can

[4] http://www.logitech.com/en-us/webcam-communications/webcams/devices/6600

be used to calculate the distance from the source to features on the images). The C905 can be seen in Figure 2.

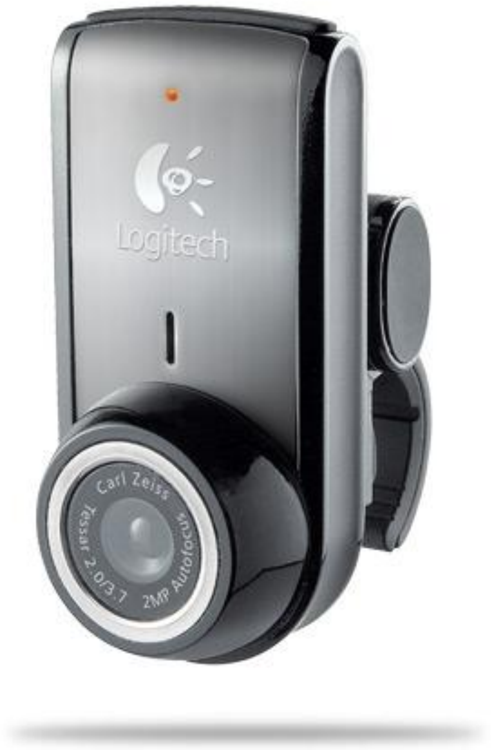

FIGURE 2. Logitech C905.

The inertial measurement unit (IMU) is a *CH Robotics CHR-6d IMU/Inclinometer*, which combines three accelerometer axes (measuring linear acceleration) and three rate gyros (measuring rate of change of angles with time) in a small package measuring just over half a square inch in size.[5] This IMU was manufactured and procured by the group specifically for its low cost, small footprint, and ease of use. The linear acceleration can be measured up to *+/- 3 g*. In comparison, one of the fastest production cars, the Bugatti Veyron Super Sport, accelerates to *100 km/h* in *2.46 s*.[6] This leads to an average acceleration (to *100 km/h*) of *11.29 m/s/s*, or *1.15 g*. Also, the IMU can measure rotation up to *+/- 400 deg/s*. The upper limits of the IMU

[5] http://www.chrobotics.com/index.php?main_page=product_info&cPath=1&products_id=1
[6] http://www.topspeed.com/cars/bugatti/2011-bugatti-veyron-164-super-sport-ar93002.html

sensitivities are clearly well above expected physical requirements. The IMU can communicate with microcontrollers via the universal asynchronous receiver/transmitter (UART) interface. The CHR-6d can be seen in Figure 3.

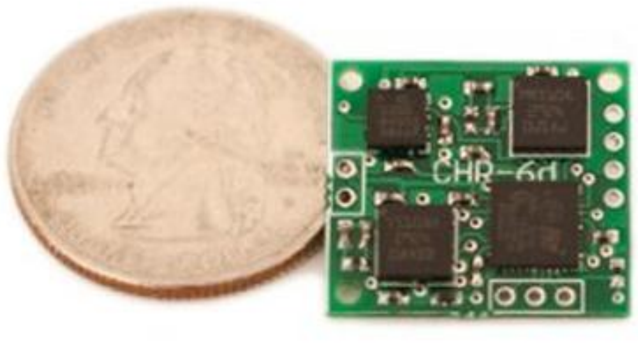

FIGURE 3. CH Robotics CHR-6d IMU/Inclinometer.

Our next sensor is a *Holux M1200 32-Channel Bluetooth-capable GPS receiver*. Like the name suggests, the GPS receiver communicates via the bluetooth interface. The included (built-in) Lithium-ion rechargeable battery can power the device for up to 15 hours without a power source, thanks to the low-power profile of the receiver. The receiver is also extremely sensitive to satellite reception (up to -159 dBm).[7] The M1200 can be seen in Figure 4.

FIGURE 4. Holux M1200 GPS receiver.

[7] http://www.holux.com/JCore/en/products/products_content.jsp?pno=227

The last sensor, the magnetometer, is a *Honeywell HMC6343 compass module with tilt compensation*. As my research was focused primarily on the compass, I will introduce it and go into details in the next section. The HMC6343 can be seen in Figure 5.

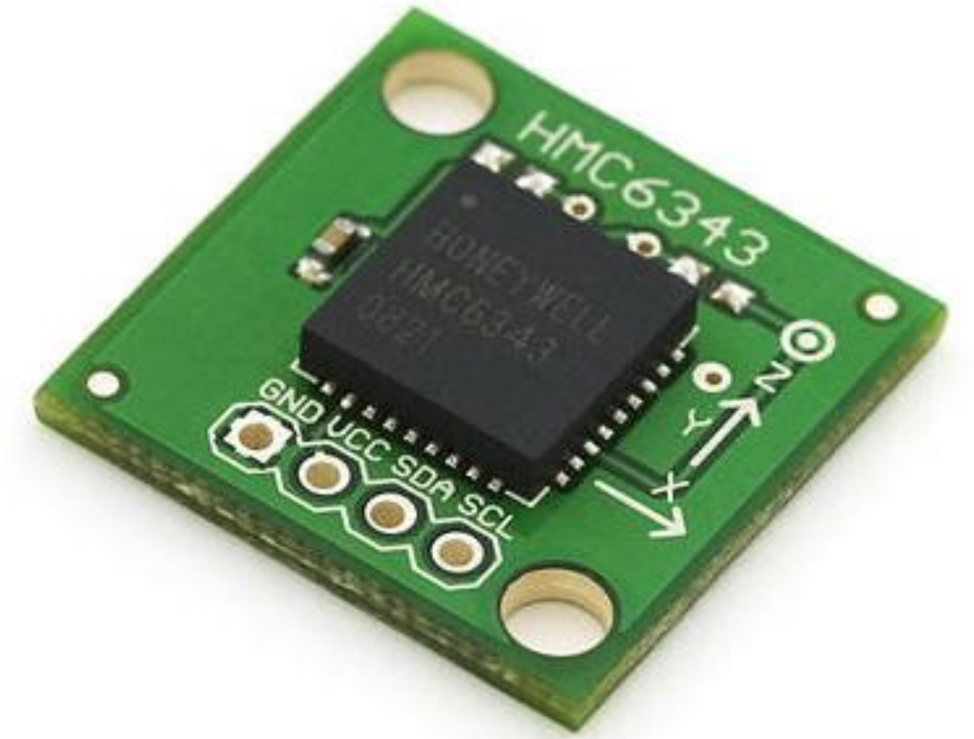

FIGURE 5. Honeywell HMC6343 (our version had a red board instead of the green shown here).

THE COMPASS MODULE

The compass module (magnetometer), the HMC6343, interfaces with the computer via the Inter-Integrated Circuit (I^2C or I2C) circuitry[8]; having worked with I2C hardware before, I chose to work on the magnetometer aspect of the autonomous navigation unit project. The magnetometer is a relatively inexpensive one: manufactured by Honeywell, the *HMC6343 compass module with Tilt Compensation* weighs 0.32 grams (in comparison, an average metal nail weighs around 0.45 grams), and costs $143 (an industrial-grade compass module can cost

[8] http://www.honeywell.com/sites/servlet/com.merx.npoint.servlets.DocumentServlet?docid=DB8BAEA9F-DB1F-45FF-843E-2F15B0A19786

anything between $300 and $2000, depending on required accuracy). This keeps the magnetometer in line with the original goal of minimal cost, weight, and power requirements. As the HMC6343 contains 3-axis magnetic field sensors, it is versatile enough to be mounted in any orientation, and then configured through the internal microprocessor accordingly. The HMC6343 also contains accelerometers; even though it is the job of the IMU to provide accurate linear acceleration readings, the acceleration output from the HMC6343 can be used as a "reality check" of sorts for the output from the IMU, and any inconsistencies can be handled by Bayesian methods.

To control the HMC6343, I designed circuitry utilized by an Arduino/ATmega168 microprocessor to monitor output from the magnetometer, and to signal changes in orientation, enter sleep mode, or perform sensor calibration. The software and instructions that I developed for the ATmega168 were written in the Arduino coding platform, which is based on C/C++. The code can be flashed on to the HMC6343 microprocessor and run independently. As the communication between the ATmega168 and the HMC6343 depends on the I2C protocol, I had to implement the Wire library into the Arduino platform to get access to I2C communication instructions.

This setup was successful in its purpose: the test-bed parent vehicle, the Lego robotic rover, was able to use its computer to control the magnetometer system effectively. An image of the magnetometer during the testing phase can be seen below (Figure 6). Here, the compass code had been completed and the breadboard was being used to run final tests on the compass before it was implemented in the main navigation unit.

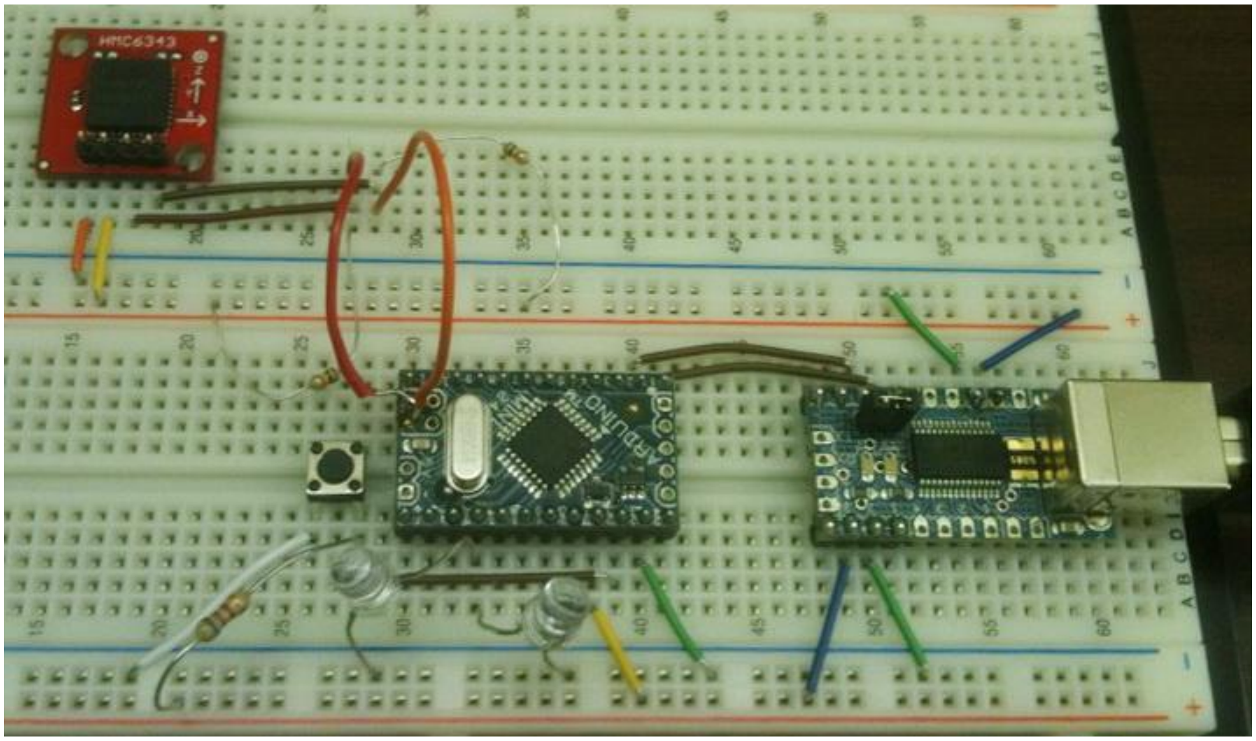

FIGURE 6. HMC6343 (upper left) with Arduino Mini (lower left) and USB-to-Arduino connector (lower right).

The HMC6343 is the printed circuit board assembly (PCBA) in the upper left hand corner. The two cables below it going downwards are the power lines. The brown cables going horizontally are the receive/transmit (RX/TX) cables going to the Arduino Mini, which is the left PCBA on the lower end of the breadboard. There is a reset switch next to it and two light-emitting diodes to signal status details during testing. The PCBA on the right of the Arduino Mini is the USB-to-Arduino connector, which, when connected to a powered USB port (which are most USB ports), provides a 5V voltage and also acts as a serial TX/RX interface to the computer.

The code for the HMC6343 was written in the Arduino coding platform. The code instructs the Arduino microcontroller (ATmega 168) to poll the HMC6343 on its current reading as necessary. The ATmega 168 can also put the HMC6343 into extended standby mode if no further readings are necessary for at least 500 milliseconds (the time it takes the HMC6343 integrated processor to reset or start-up). The HMC6343 can also be instructed to change its sensors orientations – if the magnetometer is not mounted in the default orientation (as in the picture), the *x*, *y* and *z* axes can be defined arbitrarily. The code used to run the Arduino, and hence the HMC6343, is as follows (Table 1):

```
/******************************************************************************
 *  Ali Chaudry

 *  Arduino analog pin 5 - I2C SCL
 *  Arduino analog pin 4 - I2C SDA
 ******************************************************************************/

#include <Wire.h>
#define HMC6343 0x32

int slaveAddress = HMC6343 >> 1;

byte ali[6];
int i;

int Ax; int Ay; int Az;
int Mx; int My; int Mz;
int heading; int pitch; int roll;
int temp;

void setup() {

  Serial.begin(9600);
  Wire.begin();

  delay(500);

}

void accelData() {

  // make buffer empty
  Wire.beginTransmission(slaveAddress);
  Wire.send(0x40);
  Wire.endTransmission();

  delay(1);

  Wire.requestFrom(slaveAddress, 6);
  i = 0;
  while (Wire.available() && i < 6) {
    ali[i] = Wire.receive();
    i++;
  }
  Ax = (ali[0]*256 + ali[1]);
  Ay = (ali[2]*256 + ali[3]);
  Az = (ali[4]*256 + ali[5]);

 Serial.print(Ax); Serial.print("; "); Serial.print(Ay); Serial.print("; "); Serial.print(Az);
 Serial.println();

}

void magData() {

  Wire.beginTransmission(slaveAddress);
  Wire.send(0x45);
  Wire.endTransmission();

  delay(1);

  Wire.requestFrom(slaveAddress, 6);
  i = 0;
  while (Wire.available() && i < 6) {
    ali[i] = Wire.receive();
    i++;
  }

  Mx = (ali[0]*256 + ali[1]);
  My = (ali[2]*256 + ali[3]);
  Mz = (ali[4]*256 + ali[5]);

 Serial.print(Mx); Serial.print("; "); Serial.print(My); Serial.print("; "); Serial.print(Mz);
 Serial.println();

}

void headingData() {

  Wire.beginTransmission(slaveAddress);
  Wire.send(0x50);
  Wire.endTransmission();

  delay(1);

  Wire.requestFrom(slaveAddress, 6);
  i = 0;
  while (Wire.available() && i < 6) {
    ali[i] = Wire.receive();
    i++;
  }

  heading = (ali[0]*256 + ali[1]);
  pitch = (ali[2]*256 + ali[3]);
  roll = (ali[4]*256 + ali[5]);

 Serial.print(heading); Serial.print("; "); Serial.print(pitch); Serial.print("; "); Serial.print(roll);
 Serial.println();

}
```

```
void tiltData() {

  Wire.beginTransmission(slaveAddress);
  Wire.send(0x55);
  Wire.endTransmission();

  delay(1);

  Wire.requestFrom(slaveAddress, 6);
  i = 0;
  while (Wire.available() && i < 6) {
    ali[i] = Wire.receive();
    i++;
  }

  pitch = (ali[0]*256 + ali[1]);
  roll = (ali[2]*256 + ali[3]);
  temp = (ali[4]*256 + ali[5]);

 Serial.print(pitch); Serial.print("; "); Serial.print(roll); Serial.print("; "); Serial.print(temp);
 Serial.println();

}

void loop() {

 Serial.print("Acceleration data: ");
 accelData();

 Serial.print("Magnetic data: ");
 magData();

 Serial.print("Heading data: ");
 headingData();

 Serial.print("Tilt data: ");
 tiltData();

 Serial.println(); delay(996);
}
```

TABLE 1. Code in C/C++ to run the Arduino/HMC6343 combo. This code needs to be flashed on to the ATmega 168 microcontroller in the Arduino Mini to control the HMC6343.

RESULTS / THE BIGGER PICTURE

Since the navigation unit was designed so that the IMU and magnetometer would both be directed via the Arduino/ATmega 168, it was decided that the IMU, magnetometer, Arduino Mini and USB-to-Arduino connecter should be packaged as a discrete unit. After the completion of the design of the IMU and the magnetometer, the IMU/compass package was completed by soldering all electrical connections and enclosing the PCB and sensors in a black box. The result can be seen in Figure 7 below.

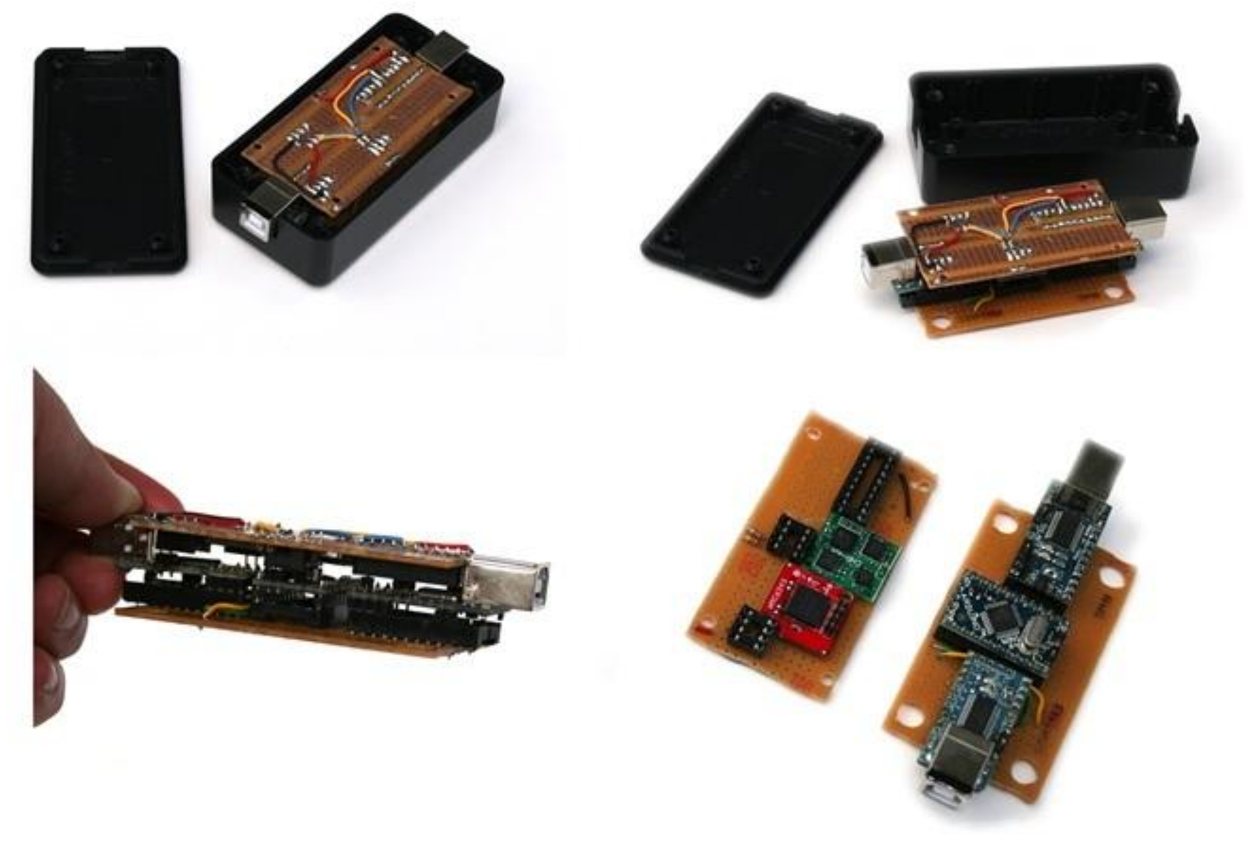

FIGURE 7. HMC6343 packaged with the Arduino Mini and the USB-to-Arduino connector.

The latest test-bed parent vehicle design can be seen in Figure 8. This is built to house the main computer and the different sensors. Multiple test runs using this setup were done. A typical test run was done in a course such as in Figure 9. In this particular case, the test location was the terrace outside the Science Library at the University at Albany. The course is marked in red (the rectangle near the center of the image).

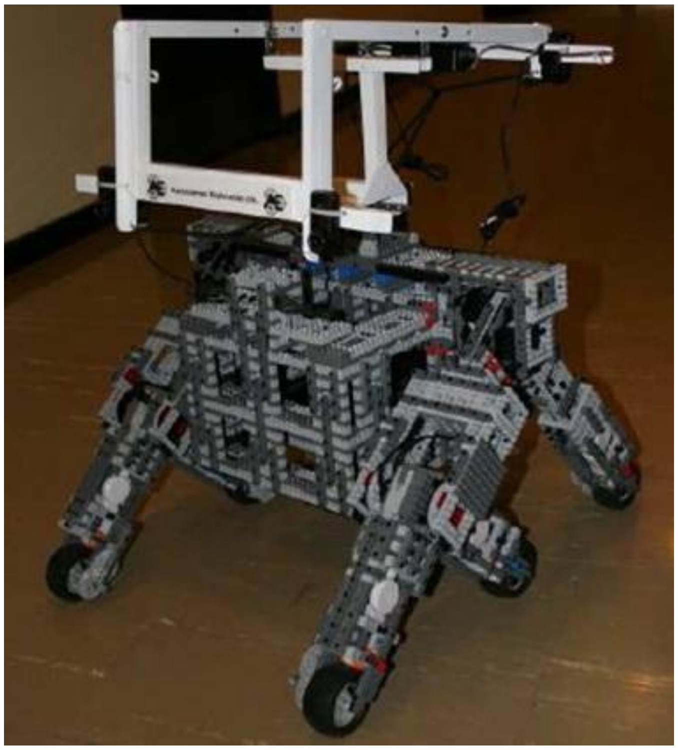
FIGURE 8. The Lego rover (test bed for the navigation system).

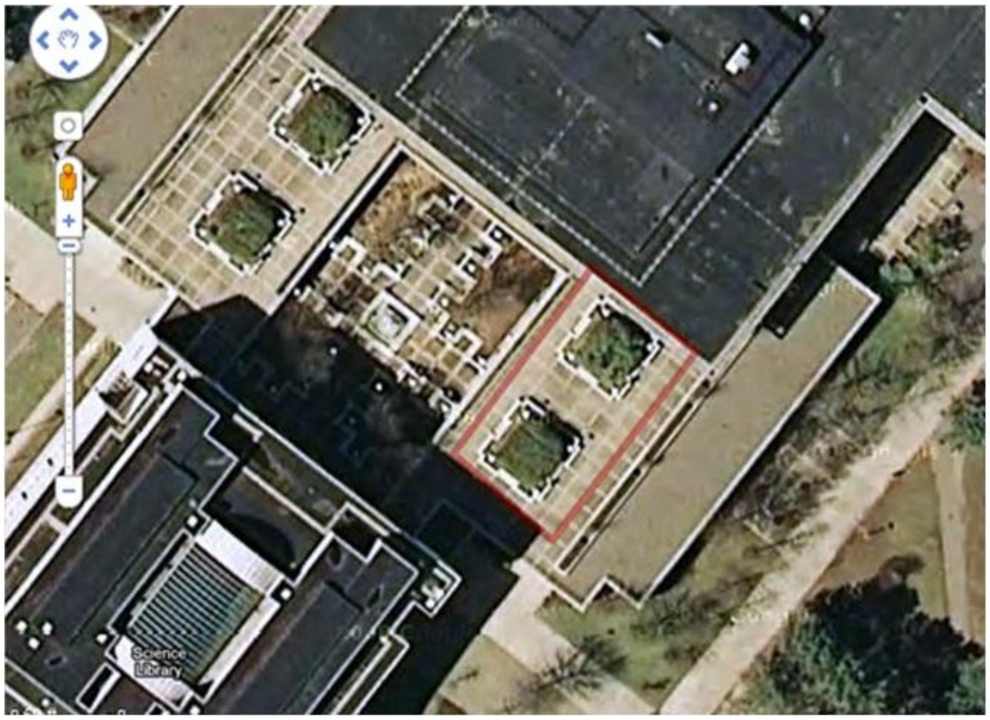

FIGURE 9. The test course – terrace outside Science Library at the University at Albany.

The test run as mapped by the navigation unit in this case is shown in Figure 10. During this run, the magnetometer data for the magnetic flux density was mapped, and is shown in Figure 11.

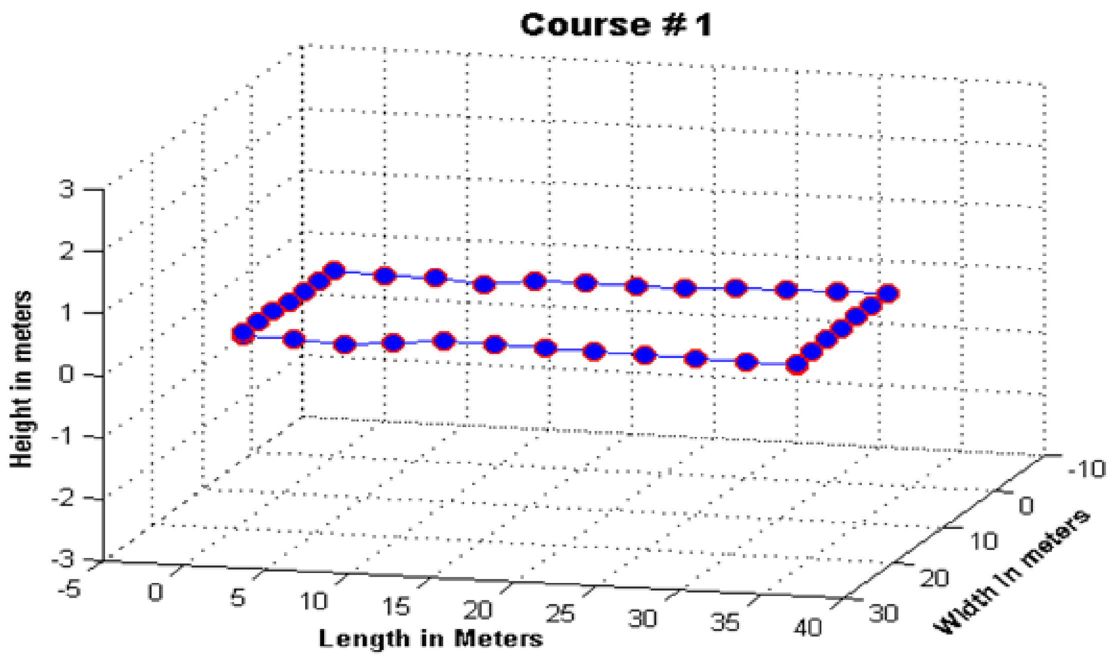

FIGURE 10. The test course as mapped by the rover.

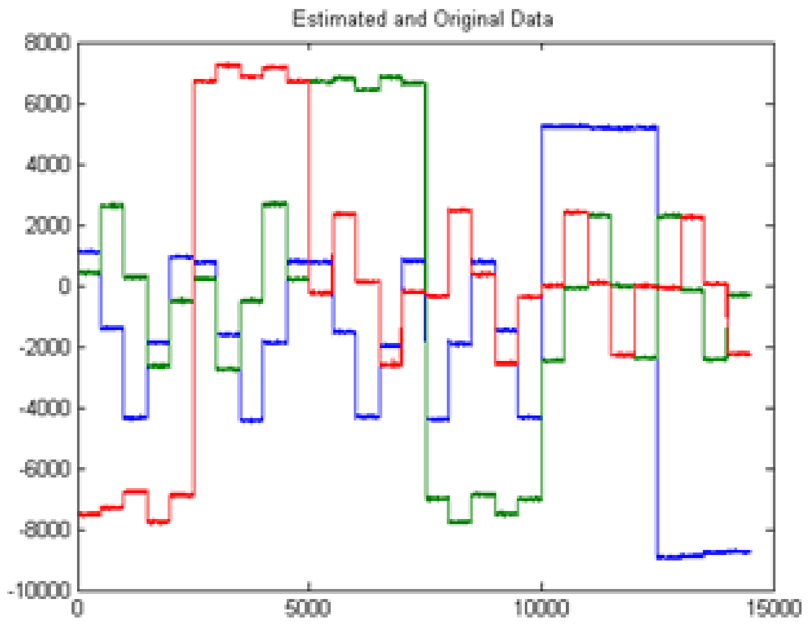

FIGURE 11. The magnetic flux density mapped in the y-axis versus the passage of time in the x-axis.

CONCLUSION

The autonomous navigation system project was partially funded by NASA for potential lunar surface navigation uses. As such, the rig needs to be tested in locations that resemble the topography of the moon – a quarry would be the ideal location to test it out. However, the parent vehicle which would act as the test bed for the navigation unit in such a location is still under construction. The current parent vehicle, although built for decently uneven surfaces, cannot traverse as rugged a location as a quarry.

Other sensors are also being developed which would aid the current array of sensors, including a laser-guided tracking system which would mesh well with the visual odometry system.

The finished navigation system will facilitate the design of lower-cost and lightweight rovers which could potentially make it feasible to launch a set of rovers to be used for wide-area exploration. These sorts of rovers or autonomous and semi-autonomous vehicles have potential applications for extra-planetary surface exploration or for the military in situations where it is either not safe or viable to send humans, and where minimum supervision of the exploration units is important and/or necessary.