Lab 7: ArduIMU and GPS Testing





Elizabeth O’Neil

Instructor: Dr. Haiyang Chao and Trevor Schlieper

Department of Aerospace Engineering

April 7, 2014

# Acknowledgements

For the use of their facilities and resources, I would like to thank the University of Kansas Spahr Library staff for the use of their facilities. I would also like to thank Dr. Haiyang Chao and Trevor Schlieper for their assistance in the experiment.

Table of Contents

Page #

[Acknowledgements ii](#_Toc384570640)

[List of Figures ii](#_Toc384570641)

[List of Tables ii](#_Toc384570642)

[Abstract 4](#_Toc384570643)

[1 Introduction 4](#_Toc384570644)

[1.1 ArduIMU Components 4](#_Toc384570645)

[1.2 Ublox GPS 6](#_Toc384570646)

[2 Procedure 7](#_Toc384570647)

[2.1 Serial Connection 7](#_Toc384570648)

[2.2 GPS Data Analysis 7](#_Toc384570649)

[2.3 MATLAB Testing 7](#_Toc384570650)

[3 Results 7](#_Toc384570651)

[3.1 Parsed GPS Data 7](#_Toc384570652)

[3.2 Parsed IMU Data 7](#_Toc384570653)

[4 Conclusions 7](#_Toc384570654)

[4.1 Sources of Error 7](#_Toc384570655)

[4.2 Lab Questions 8](#_Toc384570656)

[5 References 8](#_Toc384570657)

# List of Figures

Page #

[Figure 1: ArduIMU v3 (Ref 1) 4](#_Toc384570658)

[Figure 2: Ublox GPS 6](#_Toc384570659)

[Figure 3: UBX Packet Structure 6](#_Toc384570660)

# List of Tables

Page #

Table 1: ArduIMU v3 Properties (Ref 1) 5

Table 2: MPU-6000 Characteristics 5

Table 3: LED Displays (Ref 1) 5

Table 4: UBX Packet 6

# Abstract

The purpose of this report is to discuss the testing of an ArduIMU sensor and the analysis of a u-blox GPS sensor’s data. The ArduIMU is a small sensor equipped with an accelerometer, magnetic sensors, and a three-axis gyroscope that allows it to provide the user with complete inertial measurements in real time through serial communication. In this experiment, the operation of the ArduIMU was demonstrated in the MATLAB programming environment, allowing the collection of data. The GPS used was a small U-Blox GPS commonly used on small UAV applications. GPS data was recorded before the lab time and analyzed with MATLAB.

# Introduction

This report focuses on using the ArduIMU to gather information about the linear and rotational acceleration of the sensor and the corresponding output sent through the serial connection. This sensor is an Arduino-compatible inertial measurement unit. It combines several sensors, including an accelerometer, a gyro, and a magnetic sensor to determine the board’s movements and outputs this information through a serial connection.

## ArduIMU Components

The ArduIMU version 3, produced by DIY Drone, allows easy inertial measurements. The sensor is shown in Figure 1.



Figure : ArduIMU v3 (Ref 1)

This sensor incorporates a small 3-axis gyro, a small 3-axis accelerometer, and a 3-axis magnetometer. The readings from these sensors is processed by an Atmega328 microprocessor, which outputs the data in serial. The board is set up to be compatible with an FTDI USB cable, allowing easy serial communication with a computer. Because the board uses an Atmega328 microprocessor like many common Arduino computers, it can be programmed and monitored through the Arduino IDE (Ref 1). However, it can also communicate serially with programs such as MATLAB.

Tables 1 summarizes some of the main characteristics of the ArduIMU.

Table : ArduIMU v3 Properties (Ref 1)

|  |  |
| --- | --- |
| ***Property*** | ***Value*** |
| **Power Voltage** | 5V |
| **Current Draw (Idle Mode)** | 5 µA |
| **Accelerometer Measurement Range** | ±2, ±4, ±8, ±16 |
| **Gyro Measurement Range** | ±250, ±500, ±1000 |
| **Interface** | FTDI |
| **Default Output** | Serial, ASCII Packets |

### Accelerometer and Gyroscope

The main sensor onboard the ArduIMU is an MPU-6000, which is both a 3-axis gyro and 3 axis accelerometer. The gyro was set to the +/- 1000 deg/s range, while the accelerometer was set to the +/- 4g range. The main characteristics of the MPU-6000 are summarized in Table 3.

Table : MPU-6000 Characteristics

|  |  |
| --- | --- |
| ***Property*** | ***Value*** |
| Gyro Sensitivity Scale Factor (+/- 1000 deg/s) | 32.8 |
| Gyro Output Data Rate | 8000 Hz |
| Gyro Low Pass Filter Range | 256 Hz |
| Normal Reading for 3-axis Gyro (static) | 0,0,0 |
| Accelerometer Sensitivity Scale Factor (+/- 4g) | 8192 |
| Accel Output Data Rate | 4-1000Hz |
| Accel Low Pass Filter Range | 260 Hz |
| Normal Reading for Accel (zero roll/pitch) | 0,0,8000 |

### LED Displays

Several colored LEDs onboard allow easy troubleshooting during the setup process. Table 2 summarizes the meanings of the LED patterns.

Table : LED Displays (Ref 1)

|  |  |
| --- | --- |
| ***Display*** | ***Meaning*** |
| **Green LED on/off** | Power on/off |
| **Red/Blue/Yellow LED Flash** | Calibrating |
| **Red LED On** | On after calibrating |
| **Yellow LED On** | Speed filter suppressing yaw correction – low speeds |

In order to test the sensor’s operation, both the Arduino IDE and MATLAB were used. An FTDI cable was connected to the proper pins on the ArduIMU to enable communication.

## Ublox GPS

The Ublox LEA 5h GPS is a small, low cost GPS with 4Hz performance. It is compatible with many small electronics and computers, featuring 5V tolerant UART communication. Figure 2 shows the GPS.

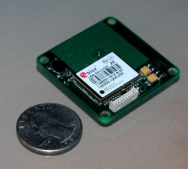


Figure : Ublox GPS

The UART serial communication sends packets to the receiver. Depending on the packet class and ID, these packets can feature various types of data, including latitude, velocity, and altitude. Figure 3 shows the basic “UBX” packet structure.

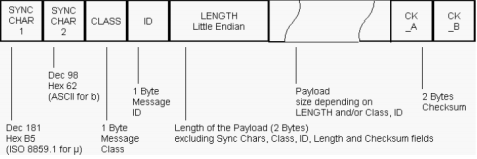


Figure : UBX Packet Structure

Table 4 shows the typical UBX packet structure both in hexadecimal and decimal.

Table : UBX Packet

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte No. | 1 | 2 | 3 | 4 | 5 | 6 | 7-34 | 35 | 36 |
| Description | Header1 | Header2 | Class | ID | Len1 | Len2 | Payload | CkA | CkB |
| 0x | B5 | 62 | 01 | 02 | 1C | 00 | Numbers |  |  |
| 0d | 181 | 98 | 01 | 02 | 28 | 00 | Numbers |  |  |

# Procedure

## Serial Connection

The FTDI Cable was connected to a USB port of the computer. The device manager in windows was used to ensure the cable’s connection and to determine the COM port used.

## GPS Data Analysis

Because the GPS data had been previously recorded, an M file was written to import the data, sort it into packets, and process the data according to the UBX packet structure discussed previously. The code performed several mathematical conversions in order to convert the bytes into the necessary data. The MATLAB code can be found in Appendix I.

## MATLAB Testing

In order to communicate with the ArduIMU through MATLAB, an M file was written to establish serial communication and monitoring. This code was written in order to configure the serial port and then process and store the data packets. The code set the proper COM port and the correct Baud Rate. The MATLAB code can be found in Appendix II.

The code collected three minutes of data with the IMU in a static situation.

# Results

## Parsed GPS Data

## Parsed IMU Data

The MATLAB code successfully allowed the collection of data packets from the ArduIMU through the serial port.

# Conclusions

This experiment demonstrated the functionality of the ArduIMU. It successfully communicated serially both with the Arduino IDE and with MATLAB. Additionally, the MATLAB code successfully parsed the GPS data collected before the lab time.

## Sources of Error

No large amounts of data were collected in this lab. The data that was collected was from a sensor that has many different calibration options. As we did not explore many settings or calibration options, it is possible that a sub-optimal setting was used. Additionally, human error in the various maneuvers used and in data collection could have added inaccuracy. Finally, errors in the processor, in the serial communication, and in the computer processing could have introduced errors. Inadequate GPS satellites available could have decreased the accuracy of the GPS data.

## Lab Questions

* See section 3.1, Figure
* See section 3.1, Figure
* See section 3.1, Table
* The packets are smaller, allowing for faster communication and easier data storage if the sensors were outputting data to a microcontroller such as an Arduino with limited data storage space.

# References

1. “ArduIMU\_GoogleProject.pdf”, Ardu-IMU Google Project, “https://code.google.com/p/ardu-imu/wiki/IntroductionPage”.
2. “ASCIItable.pdf”, Table of ASCII Characters.
3. “u-blox\_protocol\_Specifications”, ublox-lea 5h Protocol Documents.