A B/S Attitude and Heading Reference System

Changbin Chen
CQU-UC Joint Co-op Institute
Chongqing University
Chongqing, China
chencb@ucmail.uc.edu

Wenhao Du
CQU-UC Joint Co-op Institute
Chongqing University
Chongqing, China
duwo@ucmail.uc.edu

Cheng Huang
CQU-UC Joint Co-op Institute
Chongqing University
Chongqing, China
huangc5@ucmail.uc.edu

***Abstract*— This report is about the design and implement of a real-time Attitude and Heading Reference System (AHRS) based on GY953 and Arduino MEGA 2560. The system employs Browser/Server (B/S) architecture, which combines the expressive power of Python and highly-specialized-and-optimized front-end web technologies. The system provides a real-time Artificial Horizon (AH) with an accessory compass as well as historical data plots for all three x, y, z axes for not only Euler angles (roll, pitch, and yaw) but also all raw data from the gyroscope, the compass, and the accelerometer.**

Keywords*— *Attitude and Heading Reference System (AHRS), GY953, Arduino MEGA 2560

I. Attitude and Heading Reference System

A. Introduction to Navigation Systems

A navigation system is an electronic system that provides necessary information for navigating a ship or an aircraft. Modern navigation systems utilize the sensor fusion technology to combine various data collected from a wide range of sensors to construct an intuitional representation about the current attitude and position of the vehicle or aircraft. Furthermore, navigation systems can even provide context-sensitive information by locating the current position in maps embedded in the local computer or available on the Internet. There are several kinds of navigation systems available in the market nowadays, including automotive navigation systems (Baidu Map), satellite navigation systems (GPS), Inertial guidance systems, etc.

The orientation determined by a navigation system is commonly represented as Euler angles, which are three angles describing the orientation of a rigid body with respect to a fixed coordination system. In aerospace engineering, Euler angles is commonly referred to as roll, pitch, and yaw, as shown in Fig. 1.

B. Inertial Measurement Unit

Inertial Measurement Unit (IMU), which can be treated as a simplified form of Attitude and Heading Reference System (AHRS), is one of the most widespread navigation techniques currently in use. An IMU consists of motion sensors (accelerometers) and rotation sensors (gyroscopes) and sometimes an on-board computer for sensor fusion.

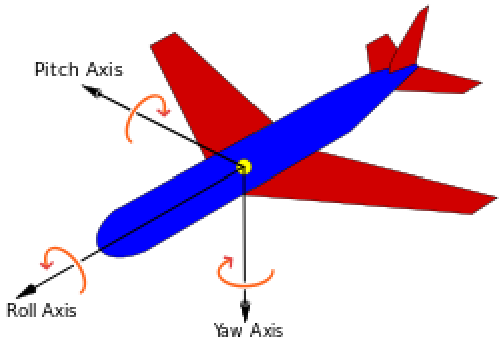

Fig. 1. Euler Angles

An IMU is a kind of self-contained navigation apparatus and is immune to jamming and deception. IMUs are typically equipped with a three-axis accelerometer and a three-axis gyroscope to measure linear accelerations and angular velocities in all three axes respectively. Gyroscopes measure the angular velocity of the object with respect to the inertial reference frame. If the initial orientation is known, by integrating the angular velocity, the object's current orientation is known at any given time. Accelerometers measure the linear acceleration of the object in the object's frame, and by integrating twice and using a known initial position, the object's current displacement, in the object frame, is also known at any given time. Then, by tracking the angular velocity and linear acceleration simultaneously and given an initial position and velocity, the object's current position relative to the inertial reference frame can be theoretically determined at any time.

Depending solely on IMUs, however, is susceptible to its innate inaccuracy. All inertial navigation systems suffer from the integration drift, which means tiny random error in the measurement from gyroscopes and accelerometers are accumulated in the ever-lasting integration process and finally results to considerably large total error in position, which invalids the whole IMU system after a certain period of time.

In order to correct errors in IMUs, various techniques have been developed. The most straight-forward method is utilizing hi-end sensors, which will effectively extend the performance period before an update or reset must be done. Sometimes, IMUs may use certain boundary conditions to correct themselves. For instance, in terrestrial use, the inertially tracked velocity is intermittently updated to zero by stopping, the position will remain precise for a much longer time, the so-called zero velocity update [1]. Then the most universal approach is supplementing IMUs with other kinds of sensors, including GPS, barometers, and so on.

C. Attitude and Heading Reference System

AHRSs can be treated as IMUs enhanced with magnetometers (compasses) and an on-board computing system. With the sensor fusion ability provided by the on-board computing system, the integration drift is compensated by measurements of gravity as well as the earth magnetic field, and attitude information, including roll, pitch, yaw, is directly available. When used in aerospace, AHRSs are often combined with air data computers to form an "air data, attitude and heading reference system" (ADAHRS), which provides additional information such as airspeed, altitude, and outside air temperature [2].

II. SYSTEM DESIGN OVERVIEW

A. Design Goals

Advanced microelectromechanical systems (MEMS) technology has opened the possibility that a whole AHRS can be packed into a single chip and controlled by commercially available microcontrollers. GY953 plus Arduino MEGA 2560 embody such a possibility, and with which we can implement a handhold AHRS and acquire real-time data in PC. This system can also find its niche in remote controlling, where for example, a tele-controlled aircraft equipped with such a system can not only send its attitude data back to the control panel in real-time, but also make possible a close-loop control system utilizing current attitude information to correct the flight course.

B. General Architecture

GY953 chip will provide Euler angles plus raw three-axis accelerations, three-axis angular speeds, and three-axis magnetic fields data from which Euler angles are computed to Arduino MEGA 2560 by SPI protocol. Arduino MEGA 2560 will transfer these data to PC using the serial communication protocol by in groups of 5 messages (RPY for Euler angles, ACC for accelerations, GYR for angular speeds, MAG for magnetic fields, and END) in which END is simply a string delimiter to ensure proper grouping. In PC, Python receives these data by using PYSERIAL and sets up a web server. Finally, React will fetch and present all these data on browsers with Bootstrap an Chart.js using HTTP and WebSocket. The whole procedure is shown in Fig. 2.

C. Hardware

GY953 is an AHRS chip whose basic specification is given in TABLE I. Arduino Mega 2560 is a micro controller board based on the ATmega2560 whose technical specifications is given in TABLE II. GY953 is connected to Arduino MEGA 2560 using SPI protocol, and the wiring is shown in TABLE III. Arduino is connected to PC using a USB cable.

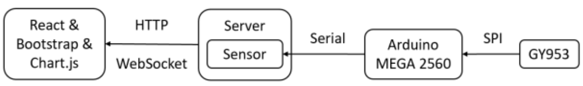

Fig. 2. General Architecture

TABLE I. Technical Specifications of GY953

Measurement Range	-180° ~ 180°
Resolution	0.1°
Measurement Accuracy	2°
Measurement Precision	2°
Default Refresh Rate	100Hz
Operating Voltage	3V ~ 5V
Operating Current	15mA
Operating Temperature	-20°C ~ 85°C
Storage Temperature	-40°C ~ 125°C
Size	15.5mm × 15.5mm

TABLE II. TECHNICAL SPECIFICATIONS OF ARDUINO MEGA 2560

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (Limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20mA
DC Current for 3.3V Pin	50mA
Flash Memory	256KB of which 8KB used by bootloader
SRAM	8KB
EEPROM	4KB
Clock Speed	16MHz
LED_BUILTIN	13
Length	101.52mm
Width	53.3mm
Weight	37g

TABLE III. WIRING

MOSI	51
MISO	50
SCK	52
CS	10
INT	2
VCC	5V
GND	GND

D. Software

Version information of all software used in this project is given in TABLE IV. Also note the whole project is developed and tested under newest version of Windows 10 and Chrome, but it should function properly under any operation systems where Python 3.5.3 or higher could run and any modern browser.

TABLE IV. SOFTWARE VERSION

ARDUINO Genuino	1.8.8
Python	3.7.3
PYSERIAL	3.4
SciPy	1.2.1
AIOHTTP	3.5.4
React	16.8.5
Bootstrap	4.3.1
Chart.js	2.7.3

III. WORKFLOW

A. GY953 and Arduino MEGA 2560

GY953 has built-in sensor fusion functionality, and all raw data and computed Euler anglers can be obtained using SPI protocol. Then in Arduino, we will use a library for GY953 written by sumotoy [3]. This library is able to set the refresh rate and read Euler angles, acceleration data, gyro data, compass data, measurement accuracy, and measurement ranges directly. Finally, we send data to PC using the serial protocol and intervein every set of RPY, ACC, GYR, MAG data with a string delimiter "END", which will used by the receiver to ensure proper group divisions.

B. Python Server

Python server consists of mainly 2 parts: one Sensor class providing an object-oriented encapsulation for Arduino MEGA 2560 and GY953 and a HTTP server.

The Sensor class uses a worker thread to automatically receive data from the serial port and perform necessary data conversion and processing and store them in a shared history dictionary which may be saved to the local file system later. The main thread keeps track of the next unread value in that history dictionary to allow users iterate over this object to fetch data.

The server part defines 4 routes, which is shown in TABLE V.

TABLE V. SERVER ROUTES

/display	real-time RPY data transfer using WebSocket
/analyse	for HTTP POST requests fetching raw data, either filtered or not, determined by specific POST parameters
/path	path to Sensor object's history JSON file
/	static files

C. *Front-End*

We use React for pages construction and data manipulation, bootstrap for styles and Chart.js for plotting. The Artificial Horizon (AH) is drawn using plain <canvas> tag and Vanilla JavaScript.

IV. RESULTS

The system functions as intended, the attitude and heading information is organized in an AH and an accessory compass as Fig. 3 shows. Also notice the maximum/minimum pitch angle is ±60°, and when the absolute value of the pitch angle is greater than 50°, the pitch reading on the top right corner will turn red as Fig. 4 shows. After a history JSON is recorded, the system is able to plot RPY, ACC, GYR, MAG data with an auto-determined data points number limit and applies filters on them if the user instructs it to do so as Fig. 5 and Fig. 6 shows.

Fig. 3. Artificial Horizon & Compass

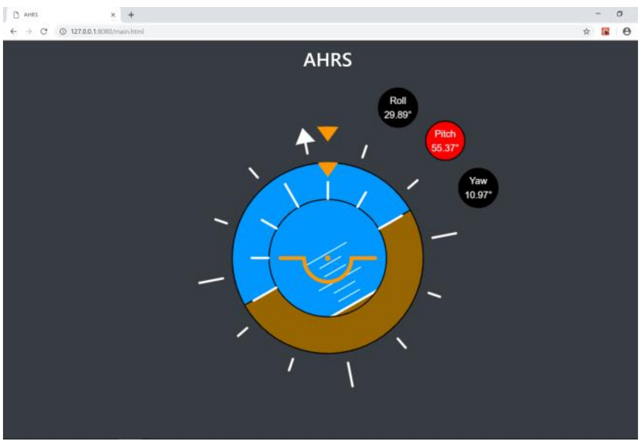

Fig. 4. Artificial Horizon & Compass (Pitch Warning)

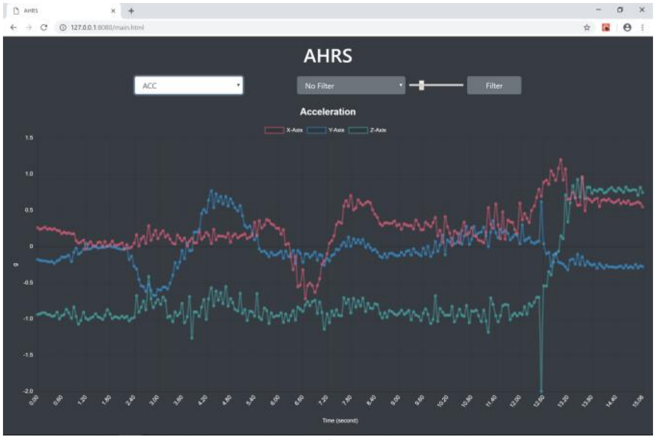

Fig. 5. ACC Plot (Raw)

Fig. 6. ACC Plot (Filter Applied)

V. FUTUREWORK

There are several points we can still make some improvements.

A. *More Compact Serial Protocol*

The transfer protocol between Arduino MEGA 2560 and PC is designed for clarity. A much more compact and robust protocol should be implemented for production usages.

B. *More Tests and Exception Handlers Needed*

For production usages, more unit tests and integration tests are required. Also, only most critical exceptions are handled in this version.

ACKNOWLEDGMENT

We are deeply indebted to friends who help us in this project. Specially, Prof. Wei Wei gave us detailed instructions for this project and insightful lectures. Maosheng Gao provided us a set of working hardware with which we finally determined our USB cable is broken. Sitong Wang provided us her GY953 with which we found we should connect CS to pin 10.

REFERENCES

[1] https://en.wikipedia.org/wiki/Inertial_navigation_system
[2] https://en.wikipedia.org/wiki/Attitude_and_heading_reference_system
[3] https://github.com/sumotoy/GY953