SOF.13 - GPS Design Document

Erik Lindgren, Emma Jakobsson

I. Document overview

This document provides the design process and validation of the GPS data acquisition module.

II. Purpose of this document

The purpose of this document is to document information about the GPS and information that can be helpful for the students the coming years when working with the GPS. How to acquire data, some issues and knowledge learned during implementation is documented to not be repeated.

III. Application

The entire application is structured as a micro-ROS project, which means all self created libraries are called components. For this project, two components were created. One called protocol, which contains the I2C driver, which allows for I2C communication. The other component is called NMEA, which contains all the filtering and parsing for the NMEA messages that the GPS module is sending. The main part of the application is a micro-ROS publisher, that uses the above components to read and process the data, and then publish it to the ROS network.

The application was validated, to make sure it behaved as expected. Refer to I for details regarding the validation and testing.

A. GPS API

Most of the specifics, such as the structure and information about NMEA and the functionalities of the GPS module, is available in the documentation provided by the manufacturer, which explains it in detail. [1] [2] [3]

B. Issues

There have been many struggles with permanently configuring the GPS, changing the output rate and the filtering. uBlox have a software called u-center that can be used for changing the settings but the new version, u-center 2, did not work for permanently saving the configuration on the GPS, after a restart the default settings were back. The older version, u-center, does work with changing the configuration and having it permanently saved. Many different alternatives were tried, the library[4] used last year was not compatible with ROS2 and the Arduino library[5] from the hookup guide for the GPS did not permanently save the configuration. The GPS inbuilt filtering did not work as intended so a custom filter was written to remove the messages that does not include the position.

IV. Tests

Two tests were done, one static inside and one moving outside. The data from the two different tests can be seen in Table I. The test results were made using the gps_validation module, which allowed for validating the data gathered. A fault is defined as any value that is not an expected value. Since the office in which the tests were conducted has the coordinate degrees of 59 and 16 (Lat and Long), there was no way we could travel far enough that those values would change. So to validate the data, each Lat/Long value was checked if it was different from 59 and 16, and if it was, that meant a fault occurred.

A. Static

The static test was conducted by placing the GPS in a window sill. After seeing that data was being read from the sensor by watching an echo of the topic, the recording of these values began. Since the application publishes to a topic, a 'ros2 bag record' was used to save all the data. The sensor was left to collect data for 10 minutes.

B. Moving

The moving test had the same setup as the static test, with the key difference that all the equipment was placed inside a box that was carried during a walk around the nearby area. This is to test a more realistic use case. The box protected it against potential rain and allowed for easier carrying. This test lasted 15 minutes.

Table I
DATA FROM THE STATIC AND MOVING TEST.

Test	Samples	Duration	Messages/Second	Faults
Static	5946	608	9.78	2
Moving	10254	941	10.90	0

C. Evaluation and recommendations

The results show that the faults are very low, almost nonexistent. However, since a very small amount of faults were present, the recommendation is to apply a moving average filter when reading the data, to make sure these very few transient values do not mess any data up. This can either be done when reading the data, or when creating it. Right now the application sends data as fast as it can, one at a time, so it is not set up to save a bunch of messages and filter them, so it is probably easier to do it by creating a moving average filter when reading from the topic the application publishes to.

REFERENCES

[1] *NEO-M9N*, Ublox, 7 2021, rev. 6.

[2] *u-blox M9 SPG 4.04*, Ublox, 6 2021, rev. 1.

[3] *u-blox 8 / u-blox M8 Receiver description*, Ublox, 8 2021, rev. 25.

[4] KumarRobotics, "ublox," https://github.com/KumarRobotics/ublox.

[5] Sparkfun, "Sparkfun_u-blox_gnss_arduino_library," https://github.com/sparkfun/SparkFun_u-blox_GNSS_Arduino_Library.