Speed & Direction App Documentation

Tulbă-Lecu Theodor-Gabriel

May 2021

Abstract

The following paper is the documentation of my final project for the Data Acquisition course from the Polytechnic University of Bucharest. This paper aims to describe how data samples from different Android device sensors were used to create human readable information about device velocity and orientation relative to Earth.

Contents

1 Introduction

For the project, an Android application was developed using Android Studio.

Android was used because it facilitated lots of sensors, easy development and testing.

2 User Interface & Features

The Program's User Interface features four main elements:

- Current speed indicator, in both m/s and km/h.
- Total distance traveled, measured in meters.
- An arrow representing the direction in which the device is traveling relative to it's orientation
- A compass to show the device's orientation relative to Earth.

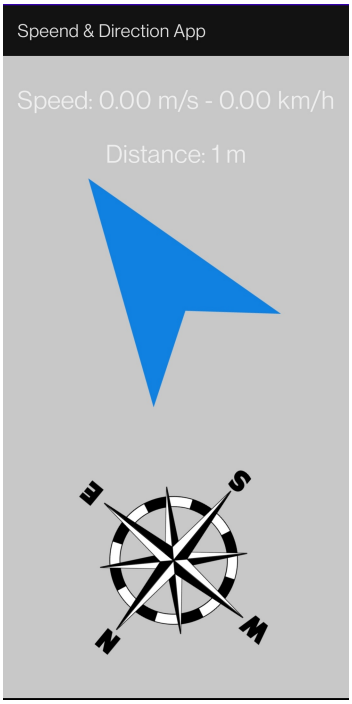

Figure 1: The user interface of the app, when the device is facing South-East and is at rest

3 Sensor Data Collection & Interpretation

3.1 GPS data, speed and total distance

The `android.location` package allows for easy data acquisition of GPS coordinates via the **Location**[2] class.

The Location class features a `.getSpeed()` method, which provides the device's speed, this is how the speed element of the UI is calculated.

It also provides a `.getElapsedRealtimeNanos()` method for determining the timestamp of the location package, thus the time between two packages can be calculated, and implicitly, knowing the time interval Δt, and the mean velocity s, we can calculate the distance as $d = \Delta t \cdot s$, this is how the distance element of the UI is calculated.

3.2 Gravity, Magnetic Field and Compass

The `android.hardware` package provides two sensors[3]
`Sensor.TYPE_ACCELEROMETER` and `Sensor.TYPE_MAGNETIC_FIELD`, which together with the method `SensorManager.getRotationMatrix()`, can be used to compute the device's orientation as a 3D vector composed of **Azimuth**, **Pitch** and **Roll**.

The Azimuth component is our angle the represents the compass' orientation.

3.3 Velocity vector

In order to compute the velocity vector (relative to the device's orientation, we first need to calculate the velocity vector relative to Earth this can be done by calculating the distances traveled longitudinally, respectively latitudinally between two GPS locations.

After we normalize this vector and the result represents the travel direction relative to Earth. To make it relative to the device's orientation we need to add the Azimuth computed in the previous section.

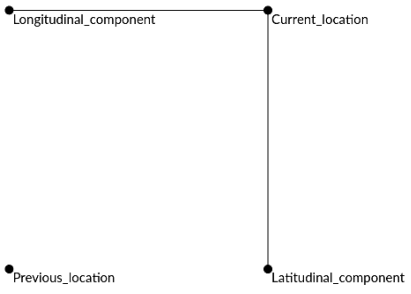

Figure 2: Illustration of how the velocity vector is calculated

4 Program Workflow

The code for the entire project can be found here: [1]. The following section is a short description of that code.

4.1 Sensor listeners classes

Two custom classes have been implemented, `GPSListener`, `SensorListener`, which have the purpose to Listen to signals from the GPS and hardware sensors and send them to the `MainActivity` class.

The two classes are sending signals to the main class using the Observer design pattern[4].

4.2 Main activity data interpretation

In the main class, whenever one of the listeners gets updated the data is processed and the UI is updated accordingly.

References

[1] Github project: https://github.com/GabiTulba/Speed_and_Direction_App

[2] Android location API:
https://developer.android.com/reference/android/location/Location

[3] Android hardware sensors API:
https://developer.android.com/guide/topics/sensors/sensors_overview

[4] Wikipedia page for observer design pattern:
https://en.wikipedia.org/wiki/Observer_pattern