Mobile Sports Applications and Data Mining

Lab 1.3 | Hl1033/CM2001 HT23

Programming & Sensors

Aim

For this last lab you are asked by your ergonomics department to create an application that can be used to monitor the arm elevation of a person throughout the day. With this data the ergonomists can determine the strain on the shoulder during a workday. Your department has asked you to make this application work both with the internal sensors of the phone, as well as with external Bluetooth sensors provided by polar.

General

This is a duo-assignment[1]. Both students must have successfully passed Lab 1.1. To ensure the quality of your work, please adhere to the following guidelines:

- The solution should be organized in a set of appropriate classes.
- Names of classes, methods and fields should describe their usage.
- Structure and document your code effectively. It is recommended to follow the MVVM (Model-View-ViewModel) architecture. While MVC (Model-View-Controller) is also acceptable, keep in mind that applications with "god" classes will be considered insufficient.
- **NB! It is mandatory to submit your source code at least a day prior to your presentation.**

The app

Figure 1: ab/adduction of the shoulder

For a proof of concept, you are **not** asked to perform the measurement throughout the whole day; instead, you are asked to create an application that can measure both abduction and adduction of the shoulder during a small period (e.g., 10-30s). To perform this measurement the user will either strap their phone or the polar sensor to their arm, after which they will perform a slow upwards motion from rest (arm along body = 0 degrees elevation) towards a position in which the arm is parallel to the ground (elevation = 90 degrees). This motion is shown in figure 1.
For both measurement methods (internal and external sensors), you are asked to compare two algorithms. For the first algorithm, you compute the angle of elevation using only the linear acceleration sensors. For the second algorithm, you compute the angle using both the linear acceleration and the gyroscope. These algorithms are explained further in section "Algorithms".
Your application should use both algorithms and have the ability to start a recording during which the values returned by the algorithms are saved. After the recording is done, the user should be able to export the data (elevation and corresponding timestamps) in a csv format.

[1] $2 \neq 3$

Basic requirements (1 point)

Create the simplest version of the application; the user should have a home screen on which they can connect with a Bluetooth sensor, choose whether they want to do the measurement with the Bluetooth sensor or the internal sensors, and start the measurement. During the measurement the user should have the ability to stop the measurement, after which the user should have the possibility to export the data. After 10-30s (you choose the value) the recording should be automatically stopped.

During the presentation you should be able to show a plot of exported data with algorithm 1 and algorithm 2 data of both your Bluetooth sensor and the internal sensor of your phone. *This plot can be made in excel.*

For these measurements we want you to collect data when a person abducts their arm from 0 to 90 degrees, and back to 0 degrees again. Firstly, make a slow motion (10s). Secondly, perform the same motion but quick (1s).

Higher level requirements (2 points)

To receive two points for this application you are required to at least do two of the following:

- During the measurement show a continuously updated graph of the elevation angle. After the measurement this graph should be shown.
- Your measurements should be saved to a local database so that you can find them back later. A different view will have to be made to display these results. You should also be able to export old results.
- Your application should be able to record in the background, so that you can perform “all day” measurements. This recording should continue when the user exits the application, but the user should be notified that the recording is going on.

Algorithms

Use your skills in basic physics and vectors to determine the angle. Figure 2 shows the orientation of the axes relative to the device.

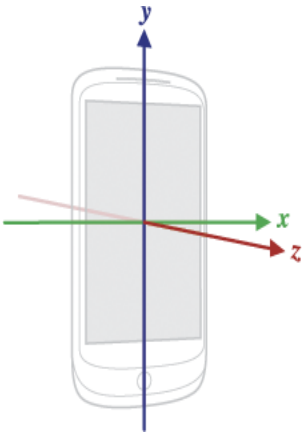

Figure 2: Reference axes

Algorithm 1 – linear acceleration

For the first algorithm, only apply a simple EWMA filter to remove "noise" from the raw data (the angle computed). The equation of an EWMA filter is the following:

$$y(n) = \alpha x(n) + (1 - \alpha) * y(n - 1)$$

In this equation, $y(n)$ is the current output, $y(n-1)$ is the previous output, and $x(n)$ is the current input; α is a number between 0 and 1. If α=1, the output is just equal to the input, and no filtering takes place.

Algorithm 2 – sensor fusion

For this second algorithm, you will combine the measurements of two sensors (linear acceleration and gyroscope) to get a more accurate value. To do this you will apply a complimentary filter, of which the equation is the following:

$$y(n) = \alpha * x_1(n) + (1 - \alpha) * x_2(n)$$

In this equation, $y(n)$ is the current output, $x_1(n)$ is the input from the linear acceleration, $x_2(n)$ is the angular input from the gyroscope, and α is a filter factor between 0 and 1. You must adjust the filter factor so that you reduce the drift from the gyroscope but still get a response when the device orientation is altered.

Detailed requirements

The below are mandatory to pass the assignment.

- Your application should be able to initiate a scan for Bluetooth devices, but only Polar devices should display.
- The user should be able to connect to a sensor from the list of scanned devices.
- If there is a problem with the Polar connection, an alert should display to the user.
- The user should be able to start both a recording with internal sensor and external sensor from the home screen.

Hints

- Create one class that processes the data, and two separate classes for acquiring the data from the different sensors (this way you can reuse your data processor and you separate your concerns)
- You can find more information about the Polar Verity Sense sensor through their GitHub page.

Android

- To perform measurements in the background, consider using a foreground activity.
- For saving results, consider the various forms of saving results (database, serializing to a file, DataStore<Proto>) and choose what fits best.
- Consider how you want to export your data; do you want to email it directly to the user? Do you want to save it to the local shared storage?

iOS

There are two ways to connect to Polars Verity sense. Other through standers BLE sdk and directly connected to the sensor for this use my example on https://gits-15.sys.kth.se/jwi/iOS_PolarH10. Or using the Polar SDK. Install Coco (follow guide on https://guides.cocoapods.org/using/getting-started.html#toc_3) then download https://github.com/polarofficial/polar-ble-sdk and under examples/examples-io/iosBleSdkTestApp/ run > install pod. After that you can open the workspace and run the app.