ECE 155 Lab Report

Hung Yui Lo | 20518882

Question 1: Provide a detailed description of how you tested your labs 1 through 4, and reflect on the effectiveness of testing you did. [10 marks]

Smoke testing is used whenever new classes, new methods as well as bug fixes were implemented. Whenever the program is unable to compile I will look at the messages logged in LogCat. LogCat is the first place that I go to check for errors because it records the type of the error as well as which line it is occurring. Compiling errors are fixed by observing the affected line. Smoke testing is used again after modifying the code.

Unit testing is used when smoke testing is passed. Unit testing is for testing individual classes as well as small units in the project. For instance TextView instances are created for displaying information such as output from sensors as well as results from operations. Any errors or bugs will be found when conducting various test cases specified for a particular class. Unnecessary classes that were not being tested were commented while using unit testing. Looking at Lab 1, Different types of sensors are split into AccelerometerListener, LightSensorListener, MagneticListener and RotationListener classes for better organization and clarity. After compiling successfully, all the data from various sensors are displayed on the screen passing them to corresponding TextViews. Test cases for all classes are conducted. For accelerometer, I shake the phone in all directions to make sure that acceleration readings from all the directions are recorded and displayed accordingly. For light sensor, I cover up the light sensor on the device periodically and verify the data displayed to ensure that I have the right sensor used. For magnetic sensor, I placed the device close to the speakers of my laptop because the coils in the speakers creates strong magnetic field that differs from surrounding environment. Lastly for rotation sensor, I rotate the device about all 3 axes listed on android development's page and make sure they corresponded to the TextView displayed on the device.

For future labs, integration testing is used to test whether the implementation of multiple sensors and algorithm works properly. For example in Lab 2, TextView instances were created for displaying current state as well as step counter to test whether providing sufficient conditions will trigger the system to record a step.

I think that the types of testing that I used are effective to complete the lab. The reason for that is because my tests covered all the levels in the code by using unit testing and integration testing. I also think that my method of testing is not robust enough if the scale of the lab getting larger. There are better solutions such as JUnit testing compared to using TextView to debug.

Question 2: The current lab descriptions do not require JUnit tests. Do you think it would be beneficial to add these as a requirement of the labs? Explain briefly, why or why not? [5 marks]

I believe that having JUnit testing as a requirement will strengthen our knowledge in proper testing which should be beneficial to people enrolled in this course. But before adding JUnit testing as a requirement of the lab, I think the lab should provide more clarification on android development, since there exist there exist fragment implementation and the lab manual did not clarify enough on how to actually implement it in real life. I know that some of my friends are having trouble using the fragment method when they proceed to later labs. I would strongly suggest dealing with that issue before introducing JUnit test.

Question 3: Describe a major design decision you made in Lab2. Explain the problem you faced, describe the alternatives you considered, justify your design decision, and evaluate the outcome of the decisions (did you make the right decisions?). [10 marks]

One of the major design decisions that I made in Lab 2 is to use 7 states in my state machine. The seven states are WAIT, RISING, PEAK, FALLING, NEGFALLING, TROUGH and NEGRISING. The reason for using so many states is because I want to increase the accuracy of my step counter. WAIT is entered when a step is incomplete or a step is completed. RISING is entered when more than 10% of positive acceleration threshold is detected and acceleration is increasing. PEAK is entered when more than 55% of positive acceleration threshold is detected and acceleration is increasing. FALLING is entered when less than 55% of positive acceleration threshold is detected and acceleration is decreasing. NEGFALLING is entered when acceleration is less than zero and acceleration is decreasing. TROUGH is entered when 50% of negative acceleration threshold is detected. NEGRISING is entered when acceleration is negative and acceleration is increasing.

One of the problems that I faced when implementing my decision is the handling of filters. I need to find the optimal value when using low-pass filter on the data received from the accelerometer. Filtering too much noise would result in a flat graph which is not desirable when finding patterns of a step. Filtering less noise would result in the presence of noise, thus affecting the state transition of the step counter.

There are alternative solutions to completing the lab. One of alternatives is abandoning the use of fragments and implements all the code without the use of fragment methods. I found that alternative rather time consuming because all my codes are implemented using fragment. Abandoning fragment at that stage is not an effective way of using resources. That is why this alternative is not considered. Another alternative is to use a different algorithm for detecting a step using the acceleration in a different way. I thought of using horizontal acceleration to calculate horizontal displacement and then use that data to identify a step. The reason for not using this alternative is that the noise generated from accelerometer significantly affects the outcome of calculations, regardless of the use of filters.

I believe that I have made the right decision when implementing Lab 2 as I have received a positive outcome from Lab 2. I have managed to shrink the error to +/- 1.25 % when walking for 80 steps. This accuracy is due to the efficient algorithm that I used as well as the state machine that was implemented. My design decisions are justified by the accuracy of the step counter and I believed that I have made a right choice.

Question 4: Explain one improvement you could make to your Lab 4 solution, if you had more time to work on it. [5 marks]

The one improvement that I would make to my lab 4 solution should I had more time to work on it is to improve the algorithm to be applicable to more complex maps. The reason for making this improvement is that I want my algorithm to be more readable and more adaptive to different maps. Right now my algorithm consists of a lot of nested IF statements and it works only for peninsula map and some bugs exist when implementing on the island map. I would like to eliminate the use of repetitive nested IF statements and use recursion method to create a more general algorithm that works on all types of maps. By using recursion I am able to reduce excessive codes and achieving a more adaptive solution.