**Increment 4**

Sai Praneeth Paruchuri

Giresh Babu Yemparala

Mounika Karampudi

Divya Battu

**Project Goal and Objectives**

**Motivation and Significance:**

Touch based games are drastically being replaced by motion sensing games worldwide. An android game, MSnake, which is a touch based is being implemented by enabling motion sensing in this project.

**Objectives:**

(1) The Main objective is to control the various movements in the game via gestures using the sensor tag. (2) Plot various graphs and maps showing the actions recognised and the places at which the user played the game. (3) Calculate the accuracy (Precision, Recall, F-Measure). And these objectives have sub tasks to be done like motion recognition and more.

**Online Application:Game by Motion**

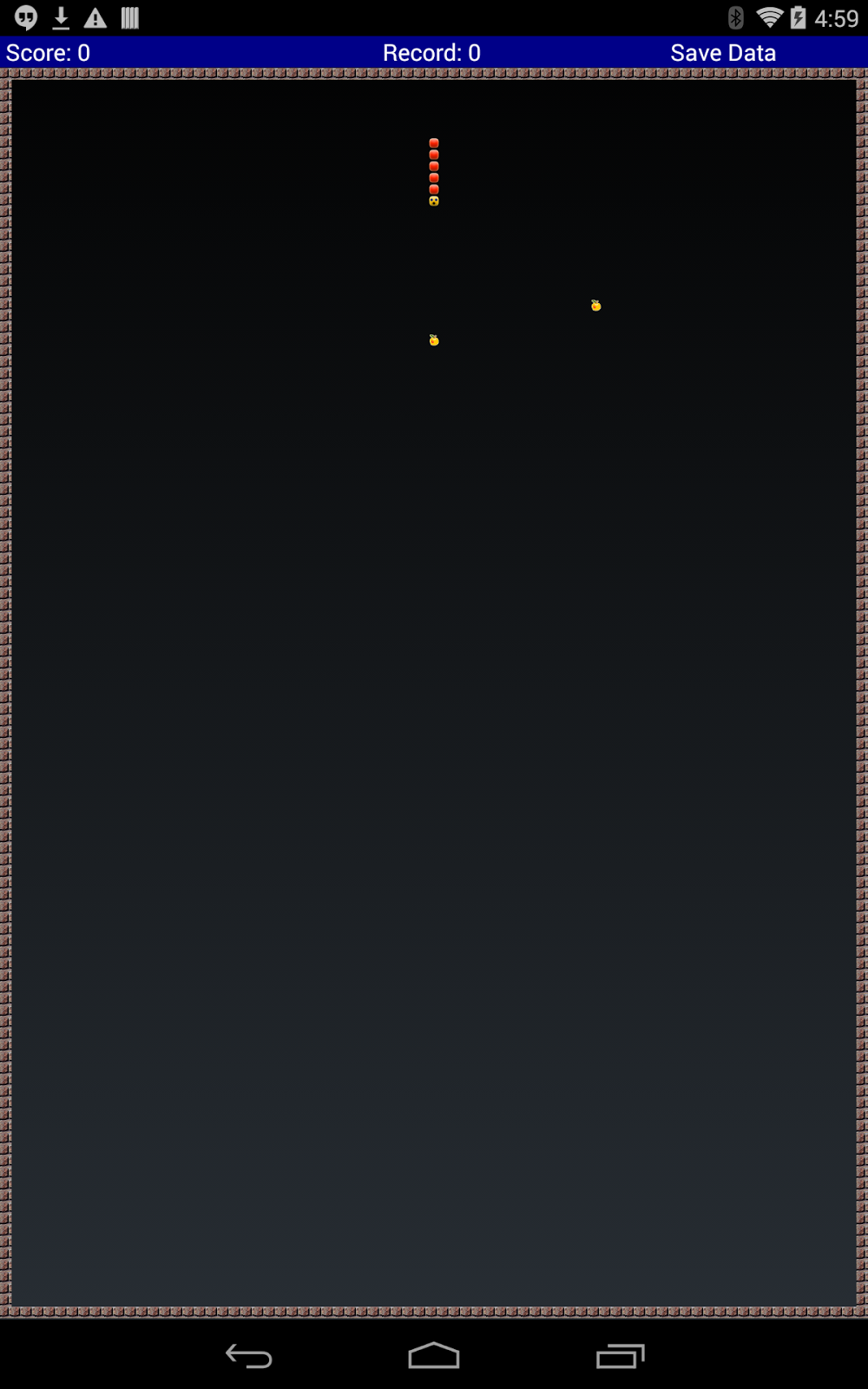
**Devices and Sensors:**

TI Sensor Tag - (Accelerometer, Gyroscope)

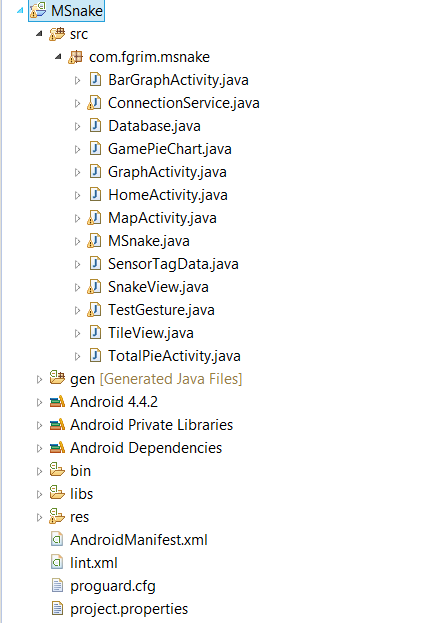
Nexus 7 Tablet/ Nexus 5 smartphone - (Bluetooth, GPS)

**Game Logic and Features:**

The Snake game consists of four movements, Left, Right, Up and Down. The Snake has to eat the apples on the screen to grow in size and gain points, and shouldn’t touch the walls of the screen or itself. The more the apples the snake eats, more the score.

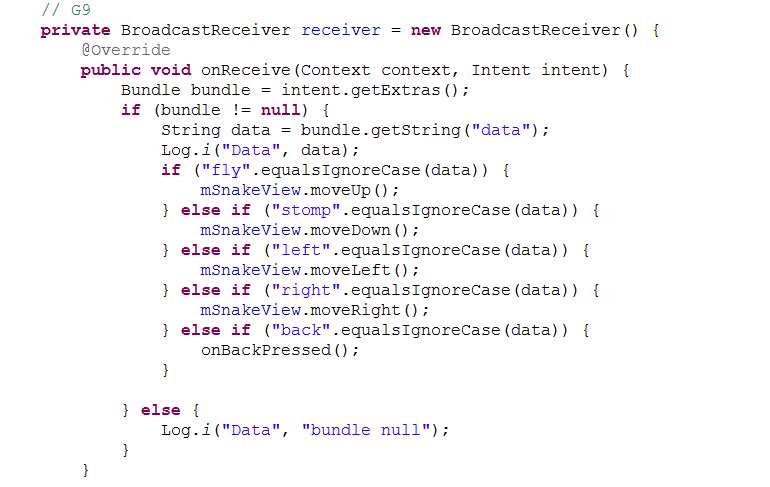


The basic android snake game consists of only three classes(MSnake, SnakeView, TileView). And here is the modified project.



The following methods will be called when the correct gesture is returned.

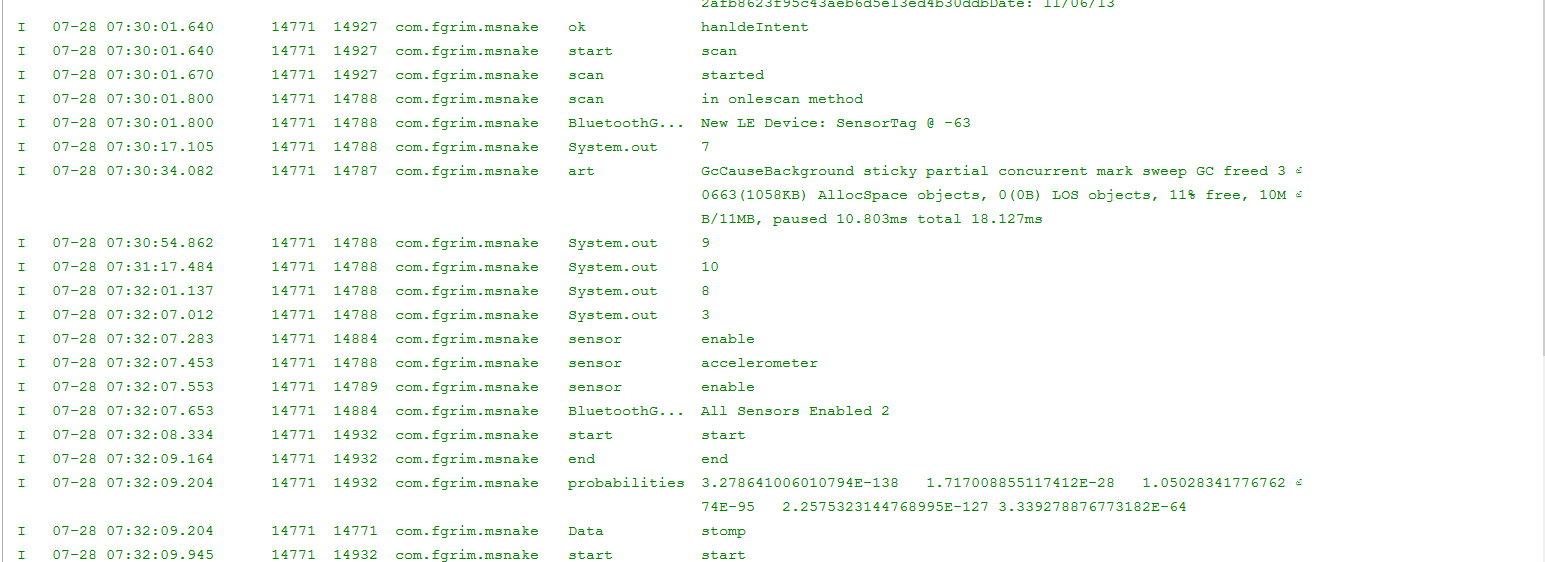




We’ve taken **five** gestures totally, one for every movement of the snake and the fifth is the back button, used to pause the game. Fly indicates upward movement, Stomp for downward movement, left for left movement, right for right movement and back to pause the game.

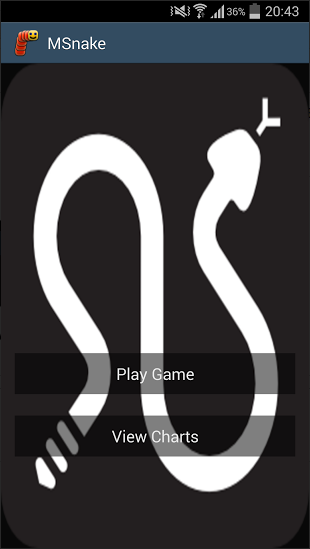
**Activity Recognition:**

HMM model was used to detect the different gestures.K means clustering is used to detect the activity. We start with k as 10 and if there is no error, k is 10, but if there’s an error the value of k is decreased and so on. We collected the data for five gestures using hand movements, trained the data for atleast 16 times. We deployed the HMMWS war file into our local glassfish and converted the txt files into sequence files with the recognition of the start and end of every gesture. These sequence files are trained at the start of the game, and once they are trained the gesture are recognized and the corresponding function is called, providing the movement of the snake.

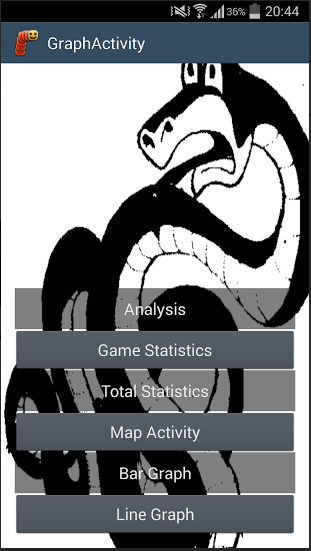


The k value for the five gestures, fly, stomp, left, right and back are 7, 9, 10, 8 and 3 respectively.

This is the main activity of the game.



Play Game button directs you to the actual snake game and the View Charts directs you to another activity with different pie charts, maps and other graphs.



**Offline Application:**

**Activity Models, Features and WorkFlow:**

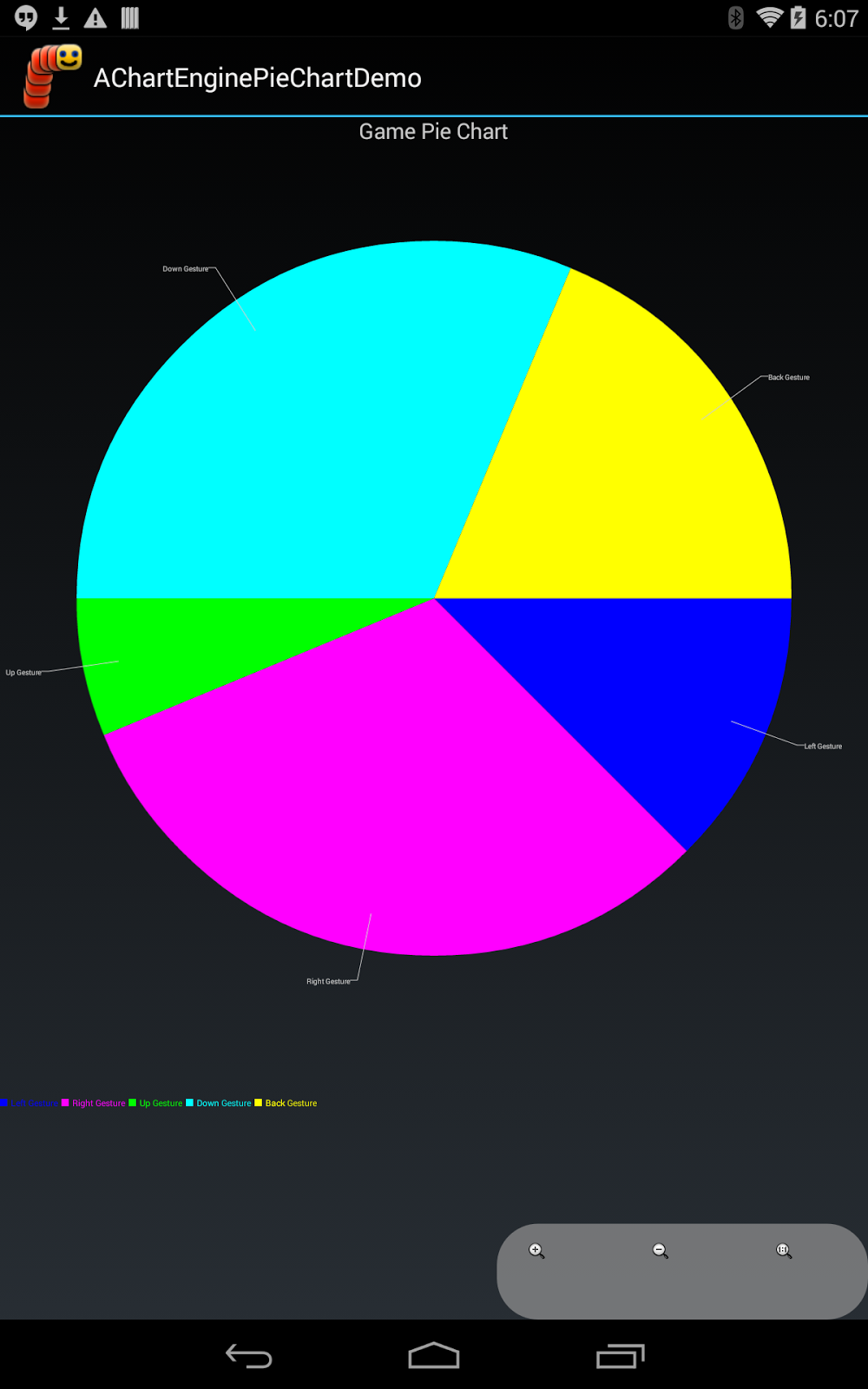
The Data is stored in Hbase after the game and retrieved to use it in the various activity models. If the server is down or if it’s too slow, we implemented an alternative way by using the Android’s SQLite database.

We used the AchartEngine API to draw the different acitivity models.

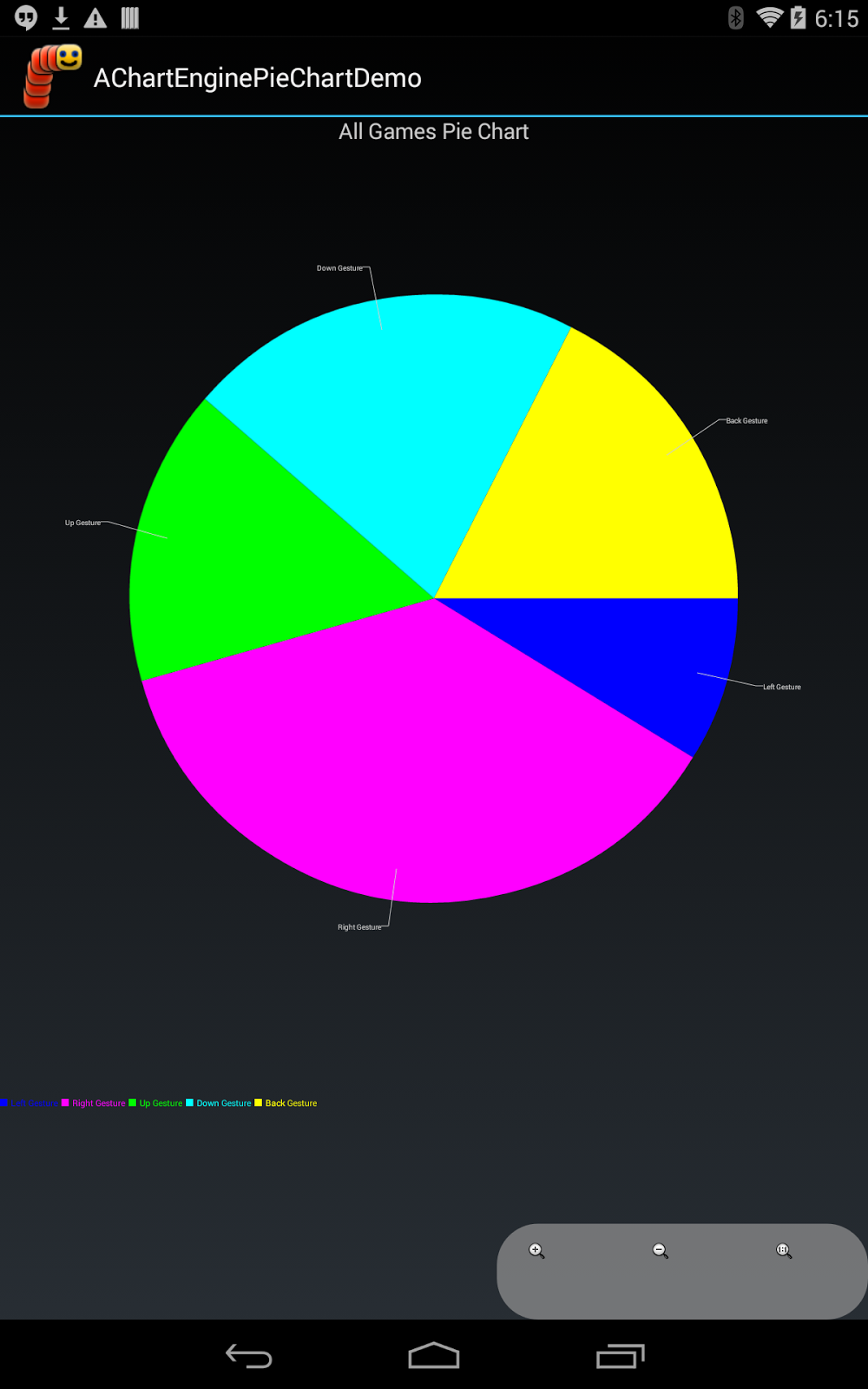
We used 4 different activity models, a pie chart, bar graph, google maps, Line Graph.

We showed two different pie charts. One pie chart shows the number of times a particular gesture is done in the last played game, and the other shows the total number of different gestures done since the app is installed on the device.

The first pie chart :

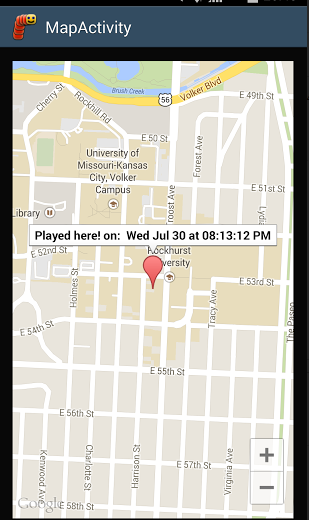


The second pie chart:

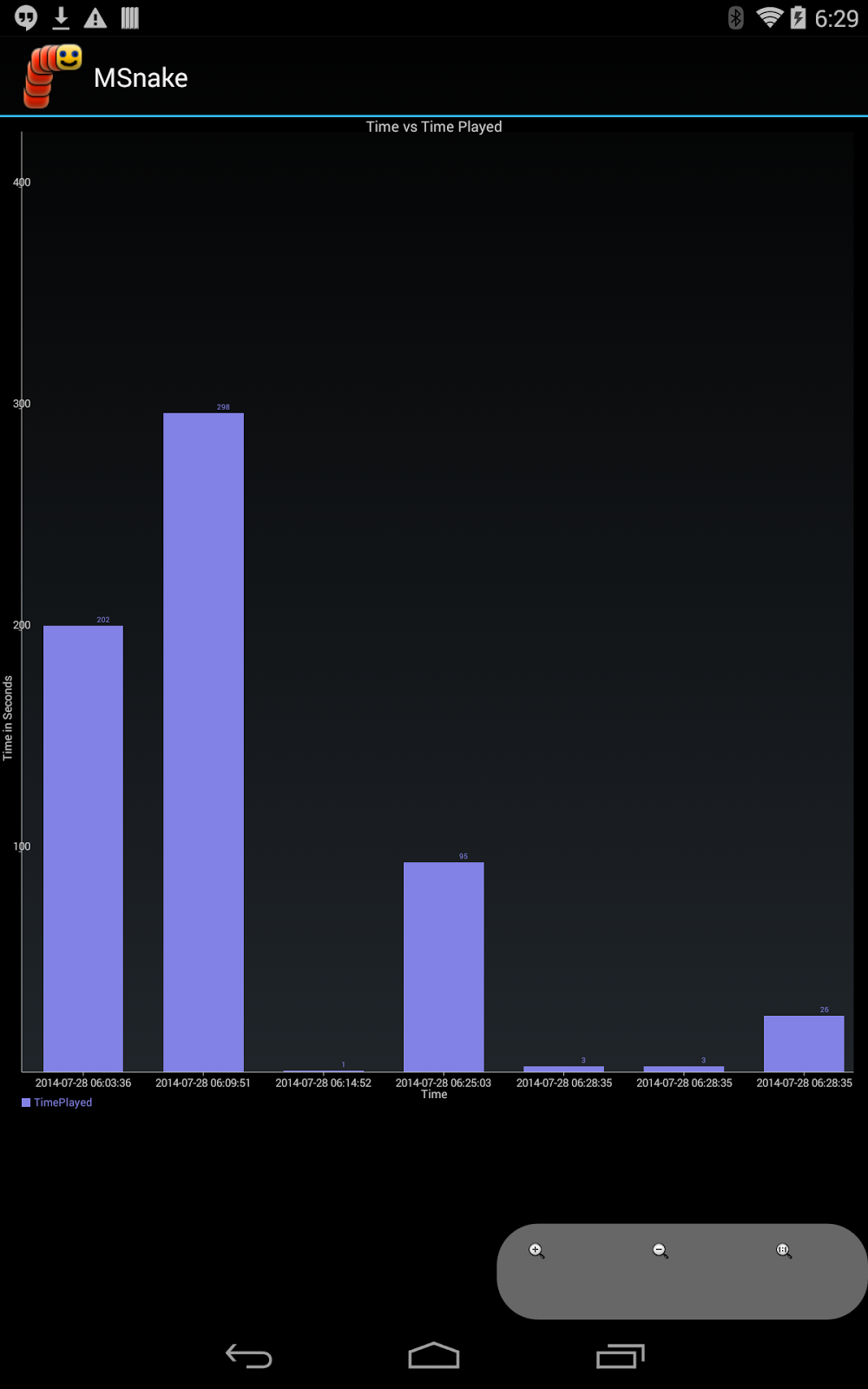


The maps button shows the different places at which the user played the game. A marker is shown on the map at these locations.

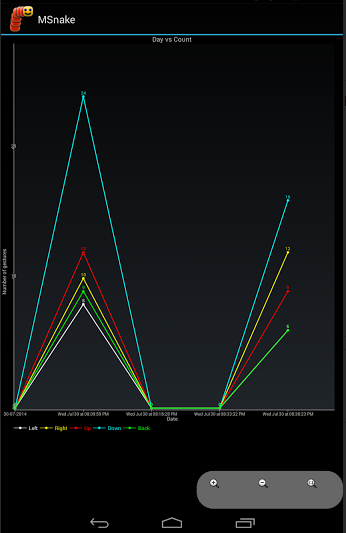
The latitude and longitude is saved into the database after the game, then retrieved and the markers are plotted on the map.



The bar graph shows how much time the user played at a particular time.



The line graph gives the comparision between the numbr of gestures used for all the games played.



**Evaluation: Motion/Activity Recognition**

**Types of Motions/ Activities:**

We employed five different gestures, four for the motion of the snake in the game and the other to pause the game. The Activity recognition is done by HMM model and K means clustering. The movement of the snake is controlled by the gestures.

We’ve trained the five gestures at least 18 times, because this increased the accuracy than when we trained for 8 times.

**Accuracy(Precision/Recall/F-Measure):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Left** | **Right** | **Up** | **Down** | **Back** |
| Left | **49** | **1** | **0** | **0** | **0** |
| Right | **0** | **50** | **0** | **0** | **0** |
| Up | **2** | **0** | **48** | **0** | **0** |
| Down | **0** | **0** | **0** | **50** | **0** |
| Back | **0** | **1** | **2** | **0** | **47** |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F-Measure** |
| **Left** | **0.96** | **0.98** | **0.9698** |
| **Right** | **0.96** | **1** | **0.9795** |
| **Up** | **0.96** | **0.96** | **0.96** |
| **Down** | **1** | **1** | **1** |
| **Back** | **1** | **0.94** | **0.969** |

**Limitations:**

(1) Since the gestures have to be processed to determine what type of gesture we have done, there is a delay in the actual movement of the snake on the screen.

(2) The gestures are not 100% perfect and so it doesn’t always reflect in the movement of the snake on the screen.

(3) Since, we’ve trained the sequence files 18 times per gesture, the training for these takes about 2 - 3 minutes when we start the game.

**Bibliography**

**https://gitorious.org/f-droid-mirrors/msnake/source/f030e855c2c32af11f2a685c9e8f1232b8dc831c:**

[**https://code.google.com/p/achartengine/**](https://code.google.com/p/achartengine/)

[**http://processors.wiki.ti.com/index.php/SensorTag\_User\_Guide**](http://processors.wiki.ti.com/index.php/SensorTag_User_Guide)

[**https://developers.google.com/maps/documentation/android/**](https://developers.google.com/maps/documentation/android/)

**YouTube:**

**https://www.youtube.com/watch?v=GYG2jwU8mmM**