

**Department of Computer Science and Electrical Engineering**

**CS590BD**

**Big Data Analytics and Apps**

**GLTron**



**Team** **Instructor**

**S**irisha **A**yyagari Prof Dr. Yugyung Lee

**P**avan Kumar **C**hinnakotla

**S**reekailash **D**epuru **TA**

**S**ree Vishnu **S**atarajupalli Feichen Shen

**Table of Contents**

**Introduction:**

For the CS590BD Course we are required to Interact with an open source Game on Android platform using the Texas Instruments Sensor Tag, which has got 9 axis accelerometer, gyroscope, magnetometer, using any or all of the on board sensors by applying Machine learning Techniques to make sense of raw data and use them as Input methods to the Game.

Gaming is an interesting domain and always growing. Interactive gaming is slowly gaining popularity with touch screen, mouse, keyboard, keypad, joystick, etc all becoming obsolete. There is a growing demand for close to reality gaming. Gaming input ranges from plain hand gestures, sensor gloves, sensor cords, and so on. We use Texas Instruments sensor tag (TI Sensor Tag) in our game to pass the input signals. So, after many thoughts and putting time on other ideas, finally we have chosen the GLTron game.

**About GLTron**

GLTron is an enthusiastic game which is about tactical racing. It is a six player game out of which 5 players are operated by the computer. A player has to ride his bike. The trail leaves a colored marking (wall). So, a player has to control his bike and keep it from running into others’ walls or obstacles. One must also not hit a wall created by oneself. The color of the wall made by each player is different. We can maneuver the bike using sensor tag.

**Approach:**

As we are given freedom to choose an open source game, it made our life easy, as Developing a Game entirely takes up a lot of time and skill. And again it is not that easy to tweak a built game. Either ways we need to have the entire percept the mechanics of game. We have chosen GLTron and skimmed through the code and found the skeleton of game which can roughly broke down into Game Logic, Input events, resources such as Images, Audio.

The bike in the game can take Left, right turns, which constantly keeps moving forward and when hit by a wall, the game ends. So, we tweaked the Input touch event classed to accept the Gesture detected by HMM model form the values of the sensor tag.

**System Features:**

The main system features include collecting the data from gestured provided by sensor tag, processing the input, and project the output appropriately on the screen. The movement of the bike is controlled by sensor tag. The bike can be turned left/ right by moving the sensor tag accordingly. When you move the sensor tag up, the game is paused. When you move it down, then the application would be closed.

**Analytics**

We can also view the statistics of the game played. When you click the ‘Show Stats’ button, you get a pie chart of all the movements of the sensor tag.

The accuracy of right/left turns can be calculated as follows:

Accuracy = number of recognized right turns/

number of actual right turns

The percentage of error in right/left turns can be calculated as follows:

Percentage error = number of actual right turns -

(number of recognized right turns/ X 100

number of actual right turns)

**First Increment**

**Introduction**

The primary goal of our project is to extend the existing GLTron which is a touch based game app and implement the same as a gesture based game app. The game uses the accelerometer sensors form the sensor tag to recognize the direction and move bike in that direction.

**Project goal and Objective:**

The main objective is to learn the Machine learning concepts and gesture recognition. For this to achieve we first have to train our system by taking different training samples by pulling accelerometer readings (x,y,z values)from the sensor tag via Bluetooth connection.

Now labeling is done to particular gesture (array of x,y,z values), and make considerable number of samples for same gesture, so the training data is robust enough to convey a particular gesture

**Project Background and related work:**

* GLTron (open source android game)
* Sensor tag (Open source Android application)

**Proposed System:**

* GL Tron App.
* The data obtained from training will be stored in the Hbase .
* Sensor tag is used as a motion sensor to play the game.

**Devices/Sensors:**

The devices we have planned to use are the sensor tag, android smart phone. Sensor tag is used

to store the gestures left, right, up and down. The smart phone is used as an interactive surface to

play the game.

**Data Collection and Preparation:**

A gesture is used to make a call a turn, so as to turn into that particular direction. First, the

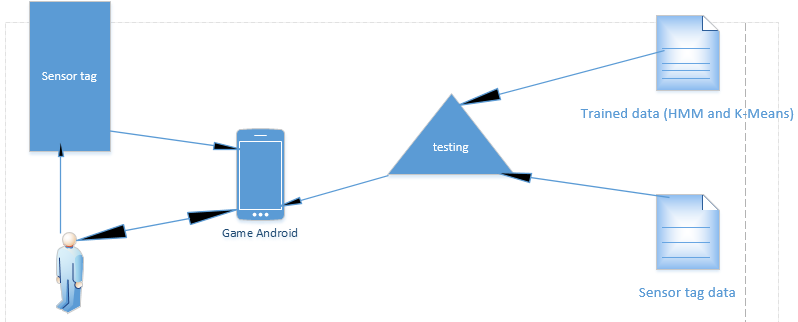
system is to be trained for gestures. Then, a gesture is assigned to that direction through code.

The system is trained using the data collection initially using the testing data.

**Second Increment**

In this Increment we skim through the Game code and identify the classes where the input events trigger.

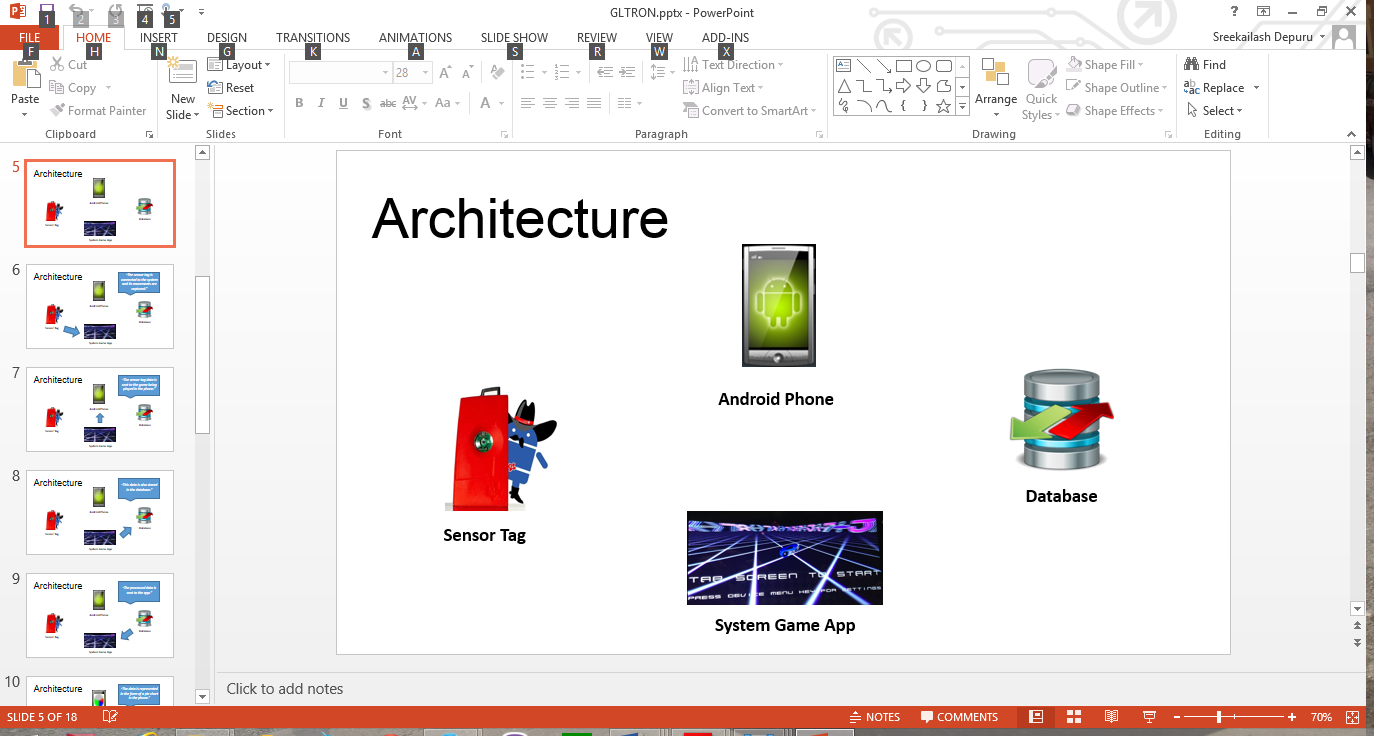
We setup our Hbase configuration on the UMKC cloud and upload out Sensor data file and generate Sequence files



**Game Logic:**

We have basically 4 things to look at when we implement this game. They are shown below:

We can explain the working of the project by means of the following figures:



**Third Increment**

Being sketched out the entire plan, we set out to train gestures required for our project

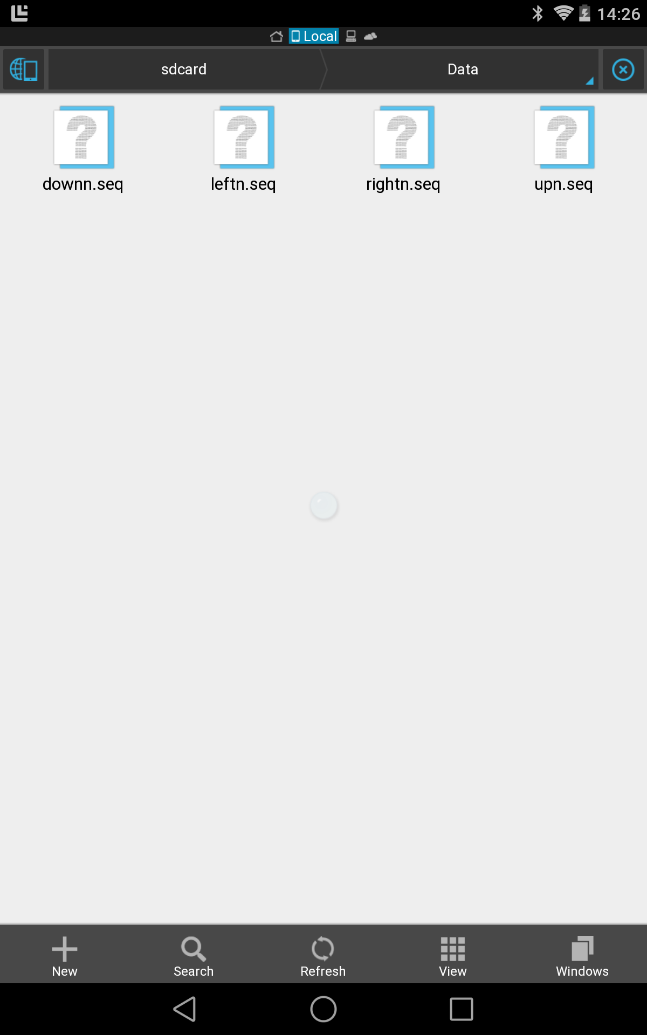
Left

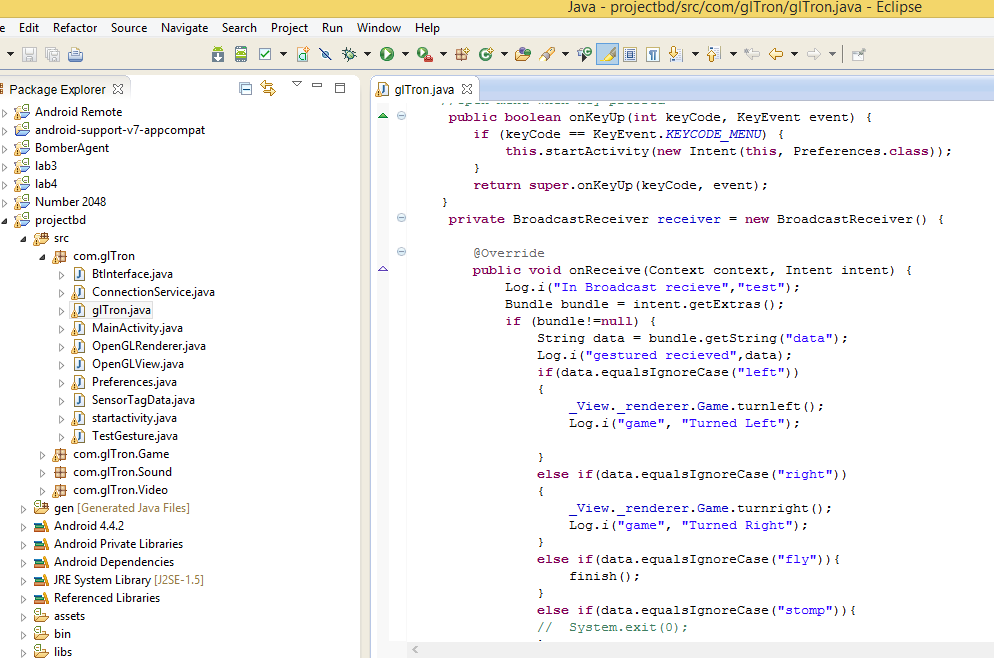
Right

Up

Down

The sequence files generated are placed in the Data folder





In main\_activity.java we will receive the broadcasted motion from Connectionservice.java and make the necessary movement in the game.

The Code snippet, where the gestures recognized invoke touch events (Left or Right) for the game. For the Final Increment, we want add Analytics part to Application

**Fourth Increment**

**Game by Motion/ Activity Report:**

**Devices/ Sensors:**

We use the Texas Instruments Sensor Tag (TI Sensor Tag) and HTC One Android mobile phone

**Motion Models:**

There are basically 4 gestures in our game. They are:

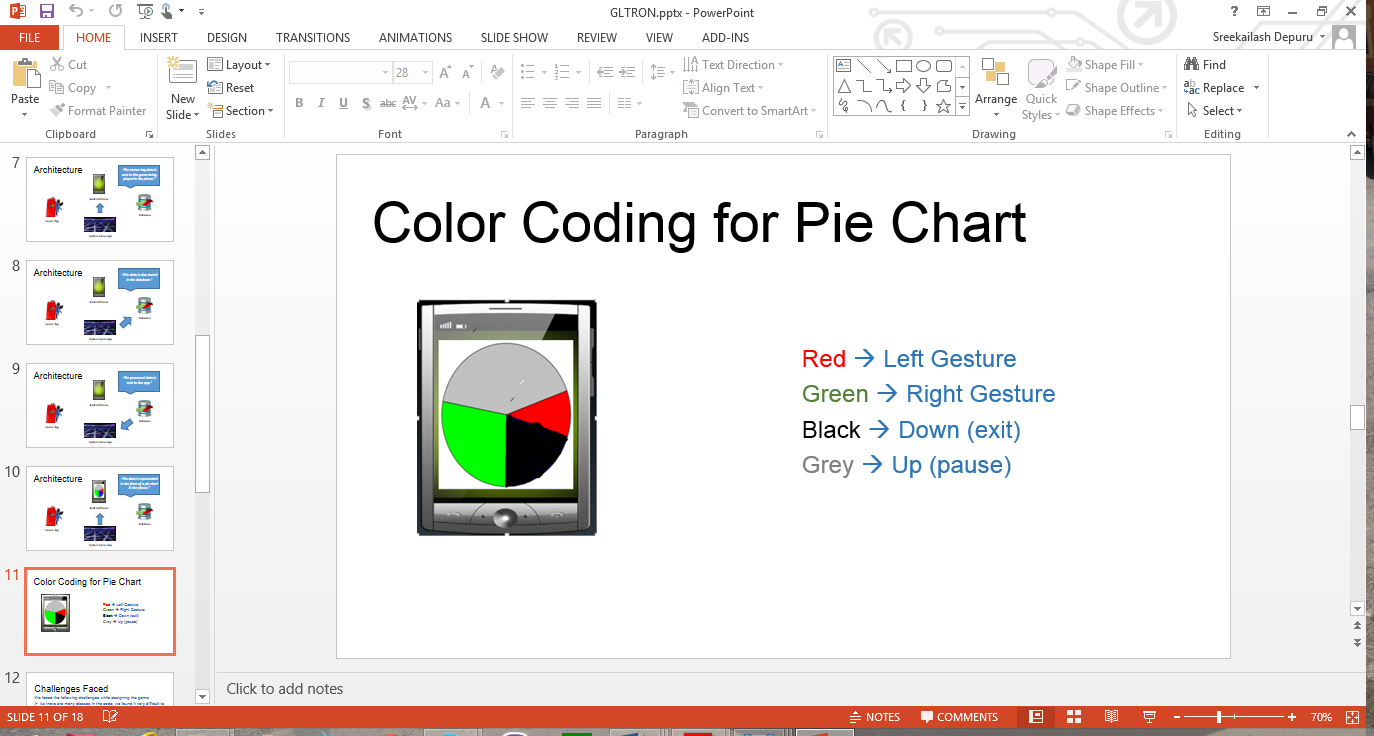
Left 🡪 Left turn

Right 🡪 Right turn

Up 🡪 Pause

Down 🡪 Exit

We already have sequence files of the trained data. The new gesture data is collected using the sensor tag and then a sequence file is generated for this data. Now, this sequence file will be compared with the trained sequence files and the appropriate gesture is detected.



**System Features and Android App GUI:**

The system requires the TI Sensor tag with the highest Bluetooth version compatibility. The android device used must also be of the highest version possible to see better results.

The picture below shows the Home Screen of the game. This screen allows user to either to start the game or view the statistics of all the games played.

**User Manual**

* Start the game
* Next the app takes a few moments to train form the seq files, after the data is trained the ‘Tap to Start’ screen appears



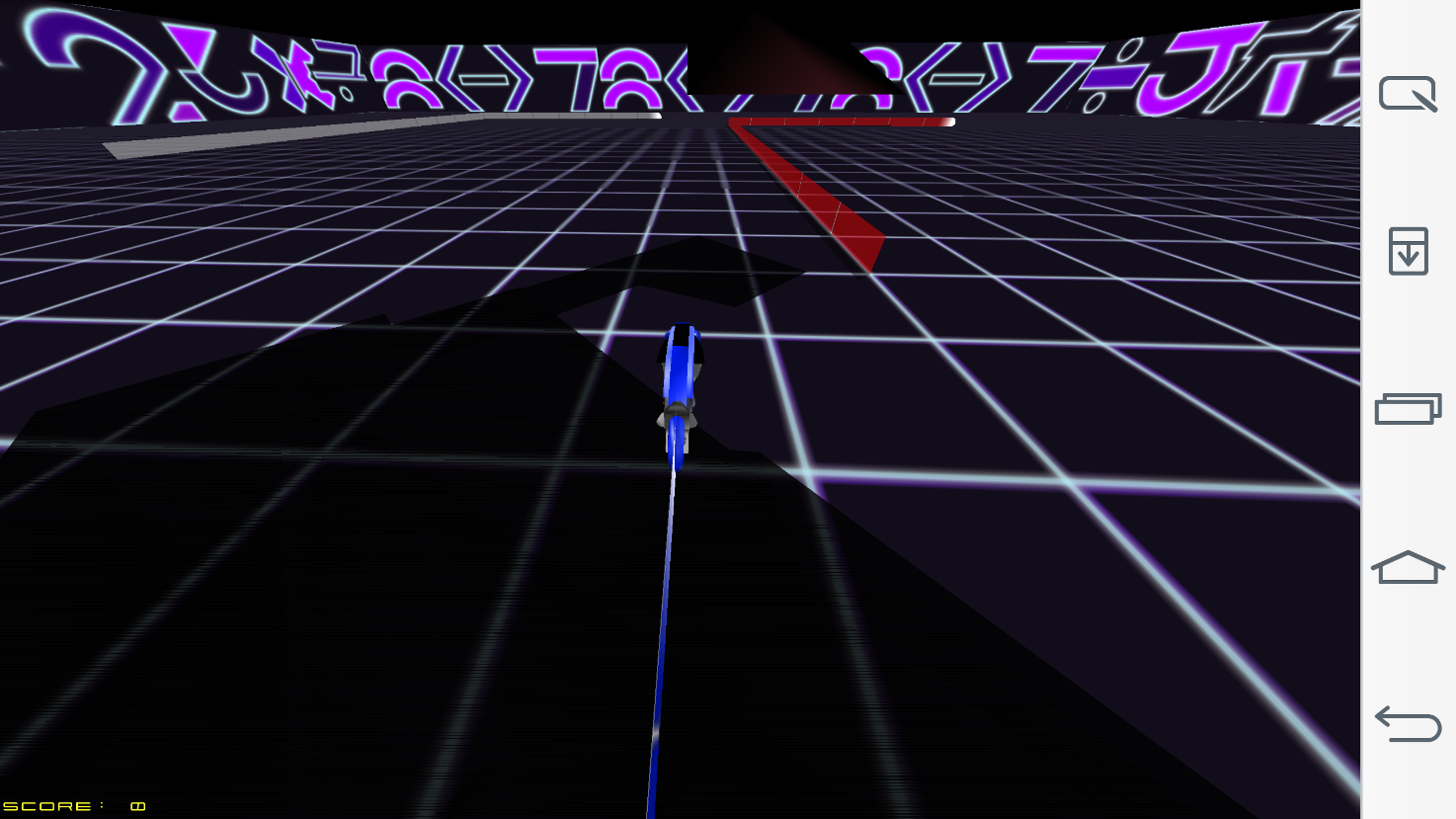
The following screen shows what happens after selecting Start Game option:



Game Play

Perform the Gesture using sensor tag

Left or Right



Game Over Screen



**Evaluation: Motion/Activity Recognition**

**Number of Users:**

There is only one user in the game. This is a 6 player game in which one player is the mobile app user and the other 5 are operated by the computer.

**Types of Motion/Activities:**

We can maneuver the bike using sensor tag. The various movements of the sensor tag are as follows:

Left 🡪 Left turn

Right 🡪 Right turn

Up 🡪 Pause

Down 🡪 Exit

**Size of Data, Number of Gestures:**

We use 4 distinct gestures and the size of data for each trained gesture is found to be around 20-25 KB. But each training file has a huge number of values which are found be good enough for detecting the activities.

**Data Preparation and Accuracy:**

The data has been collected using the sensor tag. We use this data for machine learning to generate the sequence files after capturing each gesture.

We had a problem with the accuracy initially with the value being 60%. But, we reduced the speed of the bike in the game which brought a significant rise in accuracy by 30% making it 90%.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F-Measure |
| Left | 0.667 | 0.75 | 0.706 |
| Right | 0.71 | 0.625 | 0.664 |
| Up | 1 | 0.625 | 0.77 |
| Down | 0.8 | 1 | 0.889 |

**Limitations & Challenges:**

We faced the following challenges while designing the game:

* As there are many classes in the code, we found it very difficult to back track turnleft() function. It is very difficult to sift through so many classes in the code.
* The speed of the bike is very high in the game. There is a lag in the capturing of sensor information while playing the game. So, we reduced the speed of the bike so that the sensor data is captured accurately.

**Future Work:**

There can be a few additions to this game like

* Multiplayer gaming facility wherein multiple players can play at once using their mobile phones.
* There can be a provision of some additional powers and points to keep the players motivated.
* There can be a jump feature to make the game more competitive.
* We can also make the game online on facebook so that people can invite their friends to participate in the game.
* We can have different arenas to play in.

**Project Video:**

[**https://www.youtube.com/watch?v=Hd9s0A5cqlA**](https://www.youtube.com/watch?v=Hd9s0A5cqlA)

**Related Work:**

There are a few more versions of GLTron developed by android game developers like Loki and others. They are all inspired by the movie Tron which led to the development of this game.

**Bibliography:**

<http://www.gltron.org/>

<http://www.gltron.org/download.php>

<https://play.google.com/store/apps/details?id=com.glTron&hl=en>

<http://en.wikipedia.org/wiki/GLtron>